Quality Assurance Test Plan

Apache Isis

Bilal Majeed

Bilal.majeed@student.lut.fi

TABLE OF CONTENTS

Contents

Document Revisions

Date	Version	Description	Author
6/12/2023	1.0	Quality Assurance Plan	Bilal Majeed

Document Approval

Signature	Signature
Date: 6/12/2023	Date:

1 Introduction

1.1 Purpose

The purpose of the Apache ISIS Test Plan is to establish a structured approach for testing the Apache ISIS framework thoroughly. This plan aims to ensure the reliability, functionality, and performance of Apache ISIS by defining the testing objectives, strategies, and procedures.

1.2 Scope of the Document

The scope of this document encompasses all aspects related to the testing of Apache ISIS. It includes the identification of testing levels, features to be tested, and the criteria for test completion. Additionally, it outlines the roles and responsibilities of the testing team and provides a clear delineation of what falls within the purview of this test plan.

1.3 Overview

A structured and comprehensive approach to validating the functionality, reliability, and performance of the Apache ISIS framework. As an integral part of the software development lifecycle, this test plan outlines key objectives, strategies, and methodologies employed to ensure the delivery of a high-quality and dependable framework.

Throughout this document, stakeholders will find detailed information on the scope of testing, including the identification of testing levels, features under evaluation, and the criteria for determining successful test completion. The plan delineates roles and responsibilities, providing clarity on the contributions of the testing team.

With a focus on transparency, this test plan establishes a framework for communication and collaboration among project members. Key milestones and deliverables are outlined, offering stakeholders a clear roadmap for the testing process. By adhering to the principles and guidelines presented in this document, the project team aims to achieve a thoroughly tested and reliable Apache ISIS framework, meeting the expectations of quality and performance set forth by the project's objectives.

1.4 *Definitions, Acronyms, and Abbreviations*

Term	Description
UI	User Interface
UAT	User Acceptance Testing

1.5 *References*

1. **ISTQB (International Software Testing Qualifications Board). (2018). "ISTQB Foundation Level Syllabus." Retrieved from https://www.istqb.org/downloads/send/2-foundation-level-documents/23-foundation-level-syllabus-2018.html**

2. **ISO/IEC/IEEE. (2011). "Systems and Software Engineering – Software Life Cycle Processes – Requirements Engineering." ISO/IEC/IEEE 29148:2011.**

2 Scope of Testing

2.1 *Product Overview*

This involves a detailed examination of the Apache ISIS framework, ensuring the best user experience for the end users. This aims to ensure that the software operates as intended, meeting specified requirements and quality standards.

2.2 *Product Risks*

Identification and assessment of potential risks associated with Apache ISIS. This includes an analysis of challenges that may impact the reliability and performance of the software, allowing for targeted testing.

2.3 *Test Coverage*

Test Type	Applicable (Yes/No)	Description
Unit Testing	**Yes**	**Focuses on testing individual components or modules to ensure they function correctly.**
Integration Testing	**Yes**	**Verifies the interactions between different components or modules to ensure they work together seamlessly.**
System Testing	**Yes**	**Test the entire system against defined requirements.**
Performance Testing	**Yes**	**Evaluates the responsiveness, speed, and stability of Apache ISIS under various conditions, ensuring it meets performance expectations.**
User Acceptance Testing	**Yes**	**Validate the system against user expectations.**
Compatibility Testing	**Yes**	**Check compatibility on multiple devices and browsers.**
Regression Testing	**Yes**	**Re-running previously executed test cases to ensure that new code changes have not adversely affected existing functionalities.**

2.4 *Functional Requirements*

Use Case No.	Functional Requirement
1	**Object Creation and Modification**
2	**User Authorization**
3	**User Authentication**
4	**Data Validation**
5	**Error Handling**
6	**Workflow Execution**
7	**Query and Reporting**
8	**Security Features**
9	**Integration with External Systems**
10	**User Interface (UI) Functionality**

2.5 *Non-functional Requirements*

S.N.	Non Functional Requirement	Criticality
1	**Security**	High
2	**Performance**	High
3	**Reliability**	High
4	**Usability**	Medium
5	**Compatibility**	Medium

3 Test Deliverables and Schedule

Deliverable	Description	Date
Project Charter **Stakeholder Analysis**	Define the project scope, goals, and team roles.	**Week 1-2**
Detailed Requirement Specification	Gather and finalize functional and non-functional requirements.	**Week 3-4**
Test Plan Document	Develop a comprehensive test plan.	**Week 5-6**
Test Case Documents	Create detailed test cases based on requirements.	**Week 7-8**
Unit Test Results	Begin the development phase with a focus on unit testing.	**Week 9-12**
Integration Test Results	Conduct integration testing to ensure components work together.	**Week 13-16**
System Test Results	Test the entire system against defined requirements.	**Week 17-20**
Performance Test Results	Evaluate system performance under different conditions.	**Week 21-24**
UAT Test Results	Validate the system against user expectations.	**Week 25-28**
Final Test Reports **Deployment Plan**	Address any issues, finalize documentation, and deploy the system.	**Week 29-30**
Project Closure Report	Review project performance, conduct lessons learned session, and close the project.	**Week 31**

Once the project schedule is finalized, the above delivery dates will be updated.

4 Test Design

The test design phase is crucial for formulating effective test cases that cover various scenarios and ensure comprehensive coverage. The following test design strategies and methods will be employed for Apache ISIS:

- Identifying and testing representative cases from groups of equivalent input conditions.
- Designing test cases based on specific use cases to validate end-to-end functionalities.
- Encouraging testers to explore the application and identify defects in an unscripted manner.
- Prioritizing testing efforts based on identified risks to focus on high-priority areas.
- Re-running previously executed test cases to ensure that new code changes have not adversely affected existing functionalities.

These strategies and methods aim to ensure a diverse and thorough set of test cases that effectively verify the functionality, performance, and reliability of Apache ISIS.

4.1 Test Case Review Process

The test case review process is an essential step in ensuring the quality and effectiveness of the test cases. The following approach will be adopted:

1. **Peer Reviews:**
    - Test cases will be reviewed by peers to identify potential issues, ensure clarity, and validate adherence to test design strategies.
2. **Checklists and Guidelines:**
    - Utilizing predefined checklists and guidelines to systematically review test cases, covering critical aspects such as requirements coverage and test case completeness.

3. **Cross-Functional Reviews:**
    - Involving cross-functional teams, including developers and business analysts, in the review process to gain diverse perspectives and insights.
4. **Automated Tools:**
    - Leveraging automated testing tools to identify potential issues and ensure consistency in test case design.
5. **Documentation Review:**
    - Verifying that test cases are well-documented, including clear steps, expected results, and any necessary preconditions.

The test case review process is iterative, allowing for continuous improvement and refinement throughout the testing lifecycle. Feedback from reviews will be actively incorporated to enhance the overall quality of the test cases for Apache ISIS.

5 Test Execution

5.1 *Approach*

The test execution phase involves implementing the testing strategies outlined in the test plan. Each testing level follows specific approaches with defined objectives, entry criteria, and exit criteria.

5.1.1 Unit Testing

5.1.1.1 Objective

The primary objective of unit testing is to validate the correctness of individual units or components in isolation, ensuring that they function as intended.

5.1.1.2 Entry Criteria

Entry criteria for unit testing include:

- Completion of the coding phase for the unit.
- Availability of unit test cases.
- Completion of code reviews for the unit.

5.1.1.3 Exit Criteria

Exit criteria for unit testing are as follows:

- Successful execution of all unit test cases.
- All identified defects during unit testing have been addressed.
- The unit meets predefined quality standards.

Unit testing will be considered complete when the specified objectives are achieved, and the exit criteria are met. Successful completion of unit testing provides confidence in the reliability of individual units before integration testing.

5.1.2 Integration Testing

5.1.2.1 Objective

The primary objective of integration testing is to verify the correct interaction and functioning of integrated components, ensuring that they work seamlessly together as a larger unit.

5.1.2.2 Entry Criteria

Entry criteria for integration testing include:

- Successful completion of unit testing for individual components.
- Availability of integrated builds.
- Completion of code reviews for the integrated components.

5.1.2.3 Exit Criteria

Exit criteria for integration testing are as follows:

- Successful execution of all integration test cases.
- Identification and resolution of integration-related defects.
- The integrated system meets specified performance and reliability standards.

Integration testing will be considered complete when the specified objectives are achieved, and the exit criteria are met. The successful execution of integration testing provides assurance that the integrated components collaborate effectively to deliver the desired functionality and performance.

5.1.3 System Testing

5.1.3.1 Objective

The primary objective of system testing is to validate the entire system against specified requirements, ensuring that it meets the intended functionality, performance, and user experience criteria.

5.1.3.2 Entry Criteria

Entry criteria for system testing include:

- Successful completion of integration testing.
- Availability of a stable and integrated system build.
- Completion of necessary environment setup and configurations.

5.1.3.3 Exit Criteria

Exit criteria for system testing are as follows:

- Successful execution of all system test cases.
- Identification and resolution of system-level defects.
- Achievement of specified performance and reliability benchmarks.

System testing will be considered complete when the specified objectives are achieved, and the exit criteria are met. The successful execution of system testing provides confidence that the entire system, including integrated components, functions cohesively to meet user expectations and project requirements.

5.1.4 Performance Testing

5.1.4.1 Objective

The primary objective of performance testing is to evaluate the responsiveness, speed, and stability of Apache ISIS under various conditions, ensuring it meets performance expectations.

5.1.4.2 Entry Criteria

Entry criteria for performance testing include:

- Successful completion of system testing.
- Availability of a stable and integrated system build.
- Identification of critical business scenarios for performance testing.

5.1.4.3 Exit Criteria

Exit criteria for performance testing are as follows:

- Achievement of acceptable response times under various load conditions.
- Identification and resolution of performance bottlenecks.
- Confirmation that the system meets specified performance benchmarks.

Performance testing will be considered complete when the specified objectives are achieved, and the exit criteria are met. The successful execution of performance testing ensures that Apache ISIS can handle expected user loads and performs optimally under different scenarios.

5.1.5 User Acceptance Testing

5.1.5.1 Objective

The primary objective of User Acceptance Testing (UAT) is to validate that the system meets the user's expectations and business requirements.

5.1.5.2 Entry Criteria

Entry criteria for User Acceptance Testing include:

- Successful completion of system testing and any required performance testing.
- Availability of a stable and integrated system build.
- Completion of training for end-users if applicable.
- Availability of necessary user documentation.

5.1.5.3 Exit Criteria

Exit criteria for User Acceptance Testing are as follows:

- Confirmation that the system satisfies user requirements.
- Identification and resolution of any user-reported issues.
- Approval from key stakeholders for system release.

User Acceptance Testing will be considered complete when the specified objectives are achieved, and the exit criteria are met. The successful execution of UAT ensures that Apache ISIS aligns with user expectations and is ready for production deployment.

5.1.6 Regression Testing

5.1.6.1 Objective

The primary objective of Regression Testing is to ensure that new code changes do not adversely affect existing functionalities, providing confidence in the system's stability after modifications.

5.1.6.2 Entry Criteria

Entry criteria for Regression Testing include:

- Successful completion of system testing and any required performance testing.
- Availability of a stable and integrated system build.
- Identification of code changes or enhancements that require testing.

5.1.6.3 Exit Criteria

Exit criteria for Regression Testing are as follows:

- Successful execution of regression test suite.
- Identification and resolution of any regression defects.
- Verification that existing functionalities remain unaffected by new code changes.

Regression Testing will be considered complete when the specified objectives are achieved, and the exit criteria are met. The successful execution of Regression Testing ensures that the introduction of new features or bug fixes does not compromise the integrity of the existing system.

5.2 *Test Data Requirements*

Test data is a critical component of the testing process and is required to execute test cases effectively. The test data requirements for Apache ISIS are as follows:

- **Data for Positive Testing**:
    - Data that represents valid and expected inputs, ensuring that the system behaves correctly under normal conditions.
- **Data for Negative Testing**:
    - Data that represents invalid or unexpected inputs, helping identify how the system handles errors and edge cases.
- **Performance Test Data**:
    - Data sets that simulate various usage scenarios, allowing for the evaluation of system performance under different conditions.
- **User Acceptance Test Data**:
    - Realistic data that aligns with user scenarios and business requirements for User Acceptance Testing.

- **Regression Test Data**:
    - Data sets used to verify that existing functionalities remain unaffected by new code changes.

5.3 Test Reporting

Report	Objective/Content	Owner	Audience	Frequency
Test Execution Report	Summarizes the overall execution of test cases, highlighting passed, failed, and pending test cases.	Test Lead	Development Team, Testing Team, Project Managers.	At the end of each test cycle.
Test Status Report	Provides a snapshot of the current testing status, including progress, issues, and any deviations from the test plan.	Test Lead	Project Managers, Stakeholders.	Regular intervals (e.g., weekly or bi-weekly) during the test execution phase.

5.4 Defect Management

Defect management is a crucial aspect of the testing process, ensuring that issues are identified, documented, and addressed systematically. The defect management process for the Apache ISIS project is as follows:

Defect Identification:

1. **Test Execution:** Testers identify defects during the execution of test cases, including functional, performance, and integration defects.
2. **Logging Defects:** Testers log defects in the defect tracking system, providing detailed information on the issue, including steps to reproduce, severity, and priority.

Defect Analysis:

1. **Defect Triage:** The test lead, along with relevant stakeholders, conducts defect triage to assess the severity and priority of each defect.
2. **Root Cause Analysis:** The development team analyzes the root cause of defects to understand the underlying issues and prevent recurrence.

Defect Resolution:

1. **Defect Assignment:** Defects are assigned to developers based on their nature and complexity.
2. **Defect Fixing:** Developers address and fix the reported defects in the codebase.
3. **Verification:** Testers verify that the reported defects have been successfully addressed by retesting the affected functionalities.

Defect Communication:

1. **Status Updates:** Regular status updates are provided to stakeholders, including the project manager, development team, and testing team.
2. **Defect Reports:** Defect reports are generated and shared with relevant stakeholders, summarizing the current defect status, trends, and resolution progress.

Defect Closure:

1. **Validation:** Once a defect is fixed, it undergoes validation to ensure that the resolution does not introduce new issues.
2. **Closure:** Defects are marked as closed in the defect tracking system, indicating that they have been successfully addressed and verified.

Defect Metrics and Analysis:

1. **Metrics Collection:** Metrics related to defect density, closure rate, and defect aging are collected and analyzed to assess the effectiveness of the defect management process.
2. **Process Improvement:** Insights gained from defect metrics are used for continuous process improvement, enhancing the efficiency and effectiveness of defect management.

The defect management process ensures that identified issues are systematically addressed, contributing to the overall quality and reliability of the Apache ISIS software. Regular communication and analysis help streamline the process and facilitate continuous improvement.

6 Test Team

6.1 *QA Team Organization*

6.2 *Roles and Responsibilities*

Role	Name(s)	Responsibilities
Test Lead	Bilal Majeed	Leading Software Testing Phase

7 Test Environment

The test environment is a critical component of the testing process, providing the necessary infrastructure and tools to execute test cases effectively. The test environment requirements for Apache ISIS are as follows:

7.1 Software and Hardware

Details of the software and hardware needs.

No.	Machine	Purpose	Software Requirements	Hardware Configuration	Quantity
1	Development Server	Hosting Development Environment	• JDK (Java Development Kit) • Apache Maven • Integrated Development Environment (IDE) - IntelliJ or Eclipse	• Octa-core processor • 64GB RAM • 1TB SSD	2
2	Test Server	Hosting Test Environment	• JDK (Java Development Kit) • Apache Maven • Application Server - Apache Tomcat • Database Server - PostgreSQL	• Octa-core processor • 64GB RAM • 1TB SSD	2
3	Continuous Integration (CI) Server	Integration and Automated Testing	• Jenkins • Version Control System - Git	• Octa-core processor • 64GB RAM • 1TB SSD	1
4	Client Machine	End-User Testing	• Web Browser - Chrome, Firefox, Safari • Database Client - pgAdmin	• Dual-core processor • 8GB RAM • 256GB SSD	5

7.2 *Tools*

No.	Purpose	Tool
1	Defect Management	JIRA
2	For automated testing of backend and UI.	JUnit and Selenium
3	For performance and load testing.	Apache JMeter

8 Assumptions

The successful execution of the testing process for Apache ISIS relies on certain assumptions to be in place. These assumptions are crucial for maintaining consistency, efficiency, and effectiveness throughout the testing lifecycle. The key assumptions include:

Access to Required Environments:

The QA team assumes that access to the necessary test environments, including development, testing, and production environments, will be provided as needed.

Availability of Test Data:

It is assumed that the required test data, including positive and negative scenarios, will be available for testing purposes in a timely manner.

Collaborative Communication:

Effective communication and collaboration between development, testing, and other stakeholders are assumed to be maintained, facilitating quick issue resolution and decision-making.

Timely Defect Resolution:

The QA team assumes that defects identified during testing will be addressed and resolved in a timely manner by the development team.

Adherence to Test Plan:

It is assumed that the testing team and other stakeholders will adhere to the defined test plan, including schedules, milestones, and testing processes.

Availability of Necessary Tools:

The availability and proper functioning of testing tools, including JIRA for defect management and other testing tools, are assumed to be maintained throughout the testing process.

Stable Development Builds:

The QA team assumes that stable and deployable builds will be provided by the development team for testing, minimizing disruptions during the testing process.

Support from Stakeholders:

It is assumed that stakeholders, including project managers, product owners, and end-users, will provide the necessary support and engagement throughout the testing lifecycle.

Documentation Availability:

The availability of accurate and up-to-date documentation, including requirements specifications and design documents, is assumed to be maintained for reference during testing.

These assumptions provide a foundational understanding of the expectations and conditions under which the testing process for Apache ISIS will take place. Continuous communication and collaboration are essential to address any deviations or challenges that may arise during the testing lifecycle.