Team 2

SYSTEM TEST PLAN

Xero Software

Author: Zhuowen Yan

Project Manager: <Project Manager Name>

Date: March 26 2024

**Table of Contents**

[INTRODUCTION 3](#_INTRODUCTION_(Damin))

[1. TESTING SCOPE 3](#_1._TESTING_SCOPE)

[2. TESTING OBJECTIVES 3](#_2._TESTING_OBJECTIVES)

[2.1 Core Features to be Tested 4](#_2.1_Core_Features)

[2.2 Non-Functional Features to be Tested 4](#_2.2_Non-Functional_Features)

[2.3 Features not to be Tested 5](#_2.3_Features_not)

[3. TEST PROCESS DEFINITION 5](#_3._TEST_PROCESS)

[3.1 Test Process Phases 5](#_3.1_Test_Process)

[3.2 Testing Tasks and Deliverables 6](#_3.2_Deliverables)

[4. APPROACH TO SYSTEM TESTING 7](#_4._APPROACH_TO)

[4.1 Approach to Functional Testing 7](#_4.1_Approach_to)

[4.2 Approach to Non-Functional Testing 7](#_4.2_Approach_to)

[5. ENTRY/EXIT CRITERIA 7](#_5._ENTRY/EXIT_CRITERIA)

[6. SYSTEM TEST ENVIRONMENT 8](#_6._SYSTEM_TEST)

[7. ROLES AND RESPONSIBILITIES 8](#_7._ROLES_AND)

[8. TEST CYCLES AND SCHEDULE 9](#_8._TEST_CYCLES)

9. RISKS AND CONTINGENCIES 9

# INTRODUCTION (Damin)

The core aim of the System Test Plan document for the "Xero" project is to ensure that everyone involved—from stakeholders to team members—shares a unified view regarding the extent, goals, and methodology of the system testing phase. Beyond establishing a common ground, this document dives into a variety of essential topics. It outlines the specific requirements for the testing environment, sets clear criteria for both initiating and concluding the testing process, maps out a detailed testing timeline, assigns roles and responsibilities to team members, and identifies potential risks along with their respective contingency plans. This comprehensive approach not only facilitates a well-organized testing process but also enhances collaboration and understanding among all project participants.

# 1. TESTING SCOPE (Daulet)

The functional scope of system testing includes only one functional area which is Xero Web Application.

The Xero Web Application consists of the following modules – Dashboard, User Management and Authentication, Settings and Configuration, Financial Data Entry, Bank Feeds and Reconciliations, Invoicing, Accounts Receivable, Purchase Orders, Expense Claims, Bills to Pay, Accounts Payable, Accounting Reports.

Testing is focused on Invoicing and Accounts Receivable modules as it was assigned to Team 2 members.

The technical scope includes the following architectural components:

* Web Browser
* Application server
* Database server

# 2. TESTING OBJECTIVES (Jill)

The main objective of system testing is to validate the implementation of the system features for compliance with their functional and non-functional requirements. The system test cases should include negative, i.e., challenging testing conditions in order to be effective in finding software defects. This section describes the features to be tested and the features that will be out of testing scope.

The list of project documents that will be used as a basis for designing test cases includes:

* User Stories (functional requirements)
* Requirements Composition Table (supplementary requirements)

#### 2.1 Core Features to be Tested

Invoicing Module

* Create new invoices
* Create recurring invoices
* Change existing invoices
* Submit existing invoices
* Approve existing invoices
* Delete existing invoices
* Customize invoices
* Create new credit notes
* Configure invoice reminders
* View invoice draft
* View awaiting approval invoices
* View paid invoices
* View repeating invoices
* Manage customers
* Integrate online payments
* Handle taxes on invoices
* Track invoices & status updates
* Import/export invoices

Account Receivables Module

* Set up payment
* Create new Quotations
* Accept Quotations
* View Accepted Quotations
* View Invoiced Quotations
* Conversion of Quotations to Invoices
* Add Quotation Notes
* Create Statements
* Filter Statements
* Create Aging Reports
* Export Aging Reports
* Complete Bank Account Reconciliations
* Edit customer Contact Details
* Edit customer Financial Details
* Set customer credit limits
* Manage payment terms

In addition to the above core features, testing will cover crosscutting concerns applicable to the context of individual core features, see the Requirements Composition Table for reference.

#### 2.2 Non-Functional Features to be Tested

Invoicing and Account Receivables Modules:

Performance Testing

* Test the load and stress handling capabilities for simultaneous invoice processing and account receivable operations.
* Evaluate response times for creating, submitting, approving, and viewing various financial documents under different load conditions.

Security Testing

* Verify data encryption during transmission of financial data.
* Assess application vulnerabilities to SQL injection, cross-site scripting (XSS), and other security threats.
* Ensure role-based access control is effectively restricting access to sensitive financial operations and data.

Usability Testing

* Assess the user interface for ease of navigation and clarity in the invoicing and account receivables modules.
* Evaluate the intuitiveness of creating, managing, and converting invoices and quotations.

Compatibility Testing

* Ensure the application's invoicing and account receivables features work seamlessly across different browsers, devices, and operating systems.
* Test integration compatibility with popular online payment gateways and banking software.

Reliability Testing

* Validate the system's ability to perform under expected conditions for a specified period, ensuring consistent performance and reliability.
* Test backup and recovery processes for financial data integrity.

Localization and Internationalization Testing

* Ensure the application supports multiple languages and currencies for global business operations.
* Verify correct format for date, time, currency, and number fields based on user location.

#### 2.3 Features not to be Tested

Invoicing and Account Receivables Modules:

* Deep Third-Party Integration Functionality
* Specific functionalities and performance metrics of third-party payment gateways, except for basic integration and transactional capabilities.

Legacy System Migrations

* Direct data migration from outdated or unsupported legacy systems not officially partnered with or recognized by Xero.
* Extreme Edge Cases
* Highly improbable user scenarios that fall outside the normal or extended business use cases (e.g., creating invoices with intentionally malformed data beyond typical validation checks).
* Complete Network Infrastructure
* Testing the underlying network infrastructure's performance and security, assuming it's beyond the scope of the application's direct control.
* Hardware Compatibility
* Specific hardware compatibility testing, such as printers for invoice printing, assuming a standard baseline of web and printer compatibility.
* Full-scale Regulatory Compliance Audit
* While the application should support compliance with financial reporting and tax laws, a full-scale compliance audit for every possible jurisdiction exceeds the scope of standard application testing.

# 3. TEST PROCESS DEFINITION (Laxman)

#### 3.1 Test Process Phases

**Test Planning:**

Define the scope and objectives of testing for the application, outlining what needs to be tested and why. Assign roles and responsibilities to team members to ensure clear accountability.

**Test Design:**

Identify and outline various test scenarios and cases based on the application's features. Document the scenarios in detailed test case specifications, covering inputs, expected outcomes, and steps to reproduce and ensure comprehensive test coverage, including functional and non-functional aspects, and identify the necessary test data to execute the tests effectively.

**Test Preparation:**

Set up the test environment to mirror the production or staging environment. Provision the necessary test data, ensuring data integrity and compliance with privacy regulations. Furthermore, install any associated software or dependencies required for testing purposes.

**Test Execution:**

Systematically run all identified test cases to validate the functionality, performance, and usability of the application. Document any software defects encountered during testing, along with detailed steps to reproduce them and assess their severity.

**Test Reporting:**

Summarize the outcomes of test execution, highlighting passed, failed, and pending test cases. Report defect metrics to provide insights into the application's quality and evaluate if the predefined exit criteria for testing have been met. A comprehensive test completion report is compiled, incorporating test summaries, metrics, issues encountered, and recommendations for stakeholders. Seek stakeholder approval to ensure the readiness of the application.

#### 3.2 Deliverables

On this project, the test process deliverables include:

* System Test Plan document
* Test Design specifications
* Test Case specifications
* Defects report
* Test Execution Logs
* Test Summary Report

# 

# 4. APPROACH TO SYSTEM TESTING (RAHUL)

#### 4.1 Approach to Functional Testing

The System Test will be carried out based on black-box techniques. This implies that the external functional specifications or business rules will serve as the primary source for designing test conditions. Secondly, testing will be executed from the user's viewpoint, i.e., considering the system as a black box inputting data, and assessing results through the user interface.

The system features identified above can be categorized into the following types of business logic – GUI, Field Validations, Field Interdependencies, and General Business Rules. Each type can have its test logic that can be reused across the system. Test conditions can be devised using conventional techniques, such as boundary value analysis, equivalence class partitioning, decision tables, etc. The detailed test logic for each pattern of business rules will be documented in the test design specification.

#### 4.2 Approach to Non-Functional Testing

All non-functional test objectives mentioned above can be tested using the black-box approach, i.e., from the user's perspective. The volume test should be performed for a complete production scenario that covers creating invoices, sending payment reminders, and reconciling bank transactions. The portability and responsive layout tests should cover all functions (menu options) of the system and validate that each function works under the specified test conditions.

# 5. ENTRY/EXIT CRITERIA (Rishitha)

The **Test Entry criteria** is used to formally evaluate the conditions necessary to begin test execution, it includes the following conditions:

* Development tasks related to invoices and accounts receivable have been completed and integrated.
* The System Test Plan document, specifically outlining testing for invoices and accounts receivable, has been approved.
* The QA environment is set up and ready for testing invoices and accounts receivable functionalities.
* QA team members have been granted access to the QA environment.
* Test case specifications for invoices and accounts receivable have been completed, reviewed, and approved.
* The Release Notes document, including updates related to invoices and accounts receivable, has been provided to the QA team.

The **Test Exit criteria** is used to evaluate the conditions necessary to conclude that testers can stop test execution and the system is ready for the final user acceptance testing, it includes the following conditions:

* All requirements related to invoices and accounts receivable testing are covered by test cases.
* All test cases for invoices and accounts receivable have been executed.
* There are zero critical and high-severity defects open in the invoices and accounts receivable functionalities.
* Open defects of medium and low severity in invoices and accounts receivable have known workarounds or are scheduled for future fixes.
* A Test Completion Report, specifically addressing invoices and accounts receivable testing, has been produced and communicated to stakeholders.
* QA testing sign-off for invoices and accounts receivable has been provided.

# 6. SYSTEM TEST ENVIRONMENT (Shubham)

Our system testing will leverage a dedicated environment isolated from production data. Test machines with minimum 4-core CPUs, 8GB RAM, and 50GB free space (Windows 10 or macOS Monterey) will run Xero Central. QTest will manage test data sets with version control and integrate with Xero central for test execution and defect reporting. Standard network security measures will be implemented, including firewalls and access controls. Anonymized test data and secure disposal practices will ensure data privacy.

# 7. ROLES AND RESPONSIBILITIES (Sushanth)

The project roles involved in system testing include the following:

|  |  |
| --- | --- |
| Project Role | Role Responsibilities |
| QA Manager | Responsible for overseeing testing strategy, resource allocation and quality control for the application.    Facilitate communication, drive continuous improvement, and ensure adherence to high-quality standards throughout the testing process. |
| QA Analyst | Responsible for designing a test plan to outline testing objectives, resources, and timelines, as well as establishing a centralized test repository for managing test artifacts efficiently.    Develop detailed test case specifications and execute tests rigorously, reporting any identified defects to ensure the quality and reliability. |
| Business Analyst | Ensures the application meets business requirements by translating them, validating features with stakeholders, and facilitating team communication, contributing expertise in both business and technology to its development and testing. |
| Software Tester | Responsible for evaluating the application's functionality and performance through rigorous testing.    Execute test cases, identify defects, and ensure adherence to quality standards.    Collaborating closely with developers and stakeholders. |

# 8. TEST CYCLES AND SCHEDULE (Zaid)

The system testing for our project is strategically segmented into three cycles, each aimed at meticulously examining different facets of the application to ensure comprehensive coverage and validation of functionalities according to specified requirements.

#### **Cycle 1: Invoicing Module Functionality**

**Objective:** The first cycle is dedicated to thoroughly testing the core functionalities of the Invoicing Module. This includes the creation, management, and customization of invoices, as well as the handling of credit notes and invoice reminders.

**Features to be Tested:**

* Creating new and recurring invoices
* Modifying, submitting, and approving existing invoices
* Deleting invoices and customizing invoice templates
* Generating new credit notes
* Configuring invoice reminders
* Viewing drafts, awaiting approval, paid, and repeating invoices

**Schedule:** This cycle will kick off the testing phase, concluding by April 19, ensuring all invoicing capabilities are robust, user-friendly, and error-free.

#### **Cycle 2: Customer and Quotation Management**

**Objective:** This cycle focuses on the management of customer information, integration of online payment systems, and the comprehensive testing of the Quotations feature within the application. It aims to ensure seamless interactions between users and financial operations.

**Features to be Tested:**

* Managing customer data
* Integrating with online payment gateways
* Handling taxes on invoices effectively
* Tracking invoices and updating statuses
* Importing and exporting invoice data
* Setting up payment terms and creating new quotations
* Accepting, viewing, and converting quotations to invoices
* Adding notes to quotations

**Schedule:** Following Cycle 1, this phase will run until May 3, emphasizing the application’s capability to handle complex customer and financial data accurately and efficiently.

#### **Cycle 3: Advanced Financial Reporting and Customer Management**

**Objective:** The final cycle delves into advanced functionalities, including financial reporting, bank reconciliations, and enhanced customer management features. This phase is crucial for ensuring the application’s readiness for handling comprehensive financial tasks and providing exceptional user experience.

**Features to be Tested:**

* Creating and filtering financial statements
* Generating and exporting aging reports
* Completing bank account reconciliations
* Editing customer contact and financial details
* Setting customer credit limits
* Managing payment terms effectively

**Schedule:** Commencing after the completion of Cycle 2, this cycle concludes by May 10. It is pivotal for affirming the application’s reliability and accuracy in financial reporting and customer data management.

### **Testing Overview**

These structured testing cycles are meticulously designed to cover every aspect of the application, from basic invoicing to complex financial reporting and customer management functionalities. By adhering to this schedule, we ensure a thorough examination and validation of the system, aiming for a high-quality, reliable, and efficient application upon release.

# 9. RISKS AND CONTINGENCIES (Zhuowen)

* Too many acceptable defects will lower the overall quality of the delivery product.
* Changes to the original requirements or designs will affect team morale.
* Lack of personnel resources when testing is to begin could make the test team work overtime.
* Limited testing resources may result in a delay.