QuickBooks for Mac 2016 Test Plan

**Created by: Vincent Leung**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Author | Description |
| 3/9/16 | 1.0 | Vincent Leung | Draft |
| 3/14/16 | 1.1 | Vincent Leung | 2nd draft |
| 4/7/16 | 1.2 | Vincent Leung | Edits and corrections |
|  |  |  |  |

Table of Contents

[1. Overview 3](#_Toc448163065)

[1.1. Purpose 3](#_Toc448163066)

[1.2. Objectives 3](#_Toc448163067)

[2. Testing Summary 4](#_Toc448163068)

[2.1. Scope of Testing 4](#_Toc448163069)

[3. Testing Approach 5](#_Toc448163070)

[3.1. Functional Acceptance Testing 5](#_Toc448163071)

[3.2. Stress & Volume Testing 5](#_Toc448163072)

[3.3. Regression Testing 5](#_Toc448163073)

[3.4. Unit Testing 5](#_Toc448163074)

[3.5. Integration Testing 5](#_Toc448163075)

[3.6. Security Testing 5](#_Toc448163076)

[4. Test Strategy 6](#_Toc448163077)

[4.1. Test Type & Approach 6](#_Toc448163078)

[4.2. Build strategy 6](#_Toc448163079)

[4.3. Facilities, data, and resources provision 6](#_Toc448163080)

[4.4. Testing Tools 6](#_Toc448163081)

[4.5. Testing Metrics 7](#_Toc448163082)

[5. Functions List 8](#_Toc448163083)

[6. Test Environment Plan 9](#_Toc448163084)

[6.1. Test Environment Chart 9](#_Toc448163085)

[6.2. Test Environment Details 9](#_Toc448163086)

[6.3. Establishing Environment 10](#_Toc448163087)

[6.4. Environment Control 10](#_Toc448163088)

[7. Assumptions and Dependencies 11](#_Toc448163089)

[7.1. Assumptions 11](#_Toc448163090)

[7.2. Dependencies 11](#_Toc448163091)

[8. Entry and Exit Criteria 12](#_Toc448163092)

[9. Administrative Plan 13](#_Toc448163093)

[9.1. Test Milestones and Schedule 13](#_Toc448163094)

[9.2. Training 13](#_Toc448163095)

[9.3. Defect Management 13](#_Toc448163096)

[10. Test Deliverables 15](#_Toc448163097)

# Overview

## Purpose

This purpose of this document is to define the strategy and parameters to apply to the application under test: QuickBooks for Mac 2016. The test plan will define:

* The test scope, focus areas, and objectives
* The test strategy and types of test for this release
* The test suites and issue reports
* The application’s functions list
* The details on the test environment
* Any risks, assumptions, and test dependencies
* The entry and exit criteria
* The test schedule and major milestones
* The test deliverables

## Objectives

The focus of testing will be on the main functions of the application as well as regression testing of new features as defined by a features list or changelog of previous fixes. Coverage should include front-facing features as expected to be used by a small to medium sized business for business accounting purposes, as well as sanity testing of older features. The goal is to assess validation and performance of key areas and the overall functionality as users navigate and make use of the application.

This document provides an overview of the testing that will be performed by the tester for this project. It details the overall testing requirements and provides a summary of the testing activities. Its purpose is to document:

* What will be tested
* How testing will be performed
* Who will be performing the testing activities
* Where resources are needed
* When testing will be performed and its duration

# Testing Summary

## Scope of Testing

### In scope

Functional testing of customer-facing features will be included. General usage of the application by a typical end user falls under the testing scope. In scope components include and are not limited to:

* User sign up
* User log in
* User account management
* Bank account management
* Customer management
* Employee management
* Vendor management
* Transactions and finances
* Data and reports
* Billing and accounts payable/receivable
* Statements and invoices

### Out of scope

The following testing activities and/or focus areas are out of scope:

* Security or penetration testing
* Unit testing of code
* Localization testing
* Usability or UX issues
* Use of emulators or virtual machines
* Temporary server-side issues

# Testing Approach

## Functional Acceptance Testing

This will be the main form of testing for this cycle. Test cases will be written to represent use-case scenarios. This will be used to validate functionality for all the areas under test. User inputs, actions, and control flow will be designed to be indicative of real-world scenarios. The testing effort will be split into exploratory and script-driven approaches.

## Stress & Volume Testing

Load and performance testing should be performed on the application using open source spreadsheets and data found online to simulate real-world use-case scenarios. Large amounts of data should be inputted to test application performance across different sections of the application. Any anomalies or slowdown not expected by an end-user will be noted.

## Regression Testing

Regression testing will not be performed on the application as it is already in production. If any regression issues are present, they will be discovered during the main exploratory effort.

## Unit Testing

Unit testing will not be performed on the application as this effort is focused on black box testing.

## Integration Testing

Integration testing will indirectly be performed during the main functional testing. Different functions within the application will be tested against each other, as well as the ability to process outside data sets and applications as applicable.

## Security Testing

Security testing will not be performed on the application.

# Test Strategy

## Test Type & Approach

Overall testing approach will be a guided exploratory effort. In this way, test design and execution will take place concurrently. The tester will use the product as a real first-time end-user might, deliberately learning about and exploring the functions at the same time. As the application would already be in production, the different modules or units would already be a fully integrated system. In this way, a big bang testing strategy would be more appropriate than an incremental one.

## Build strategy

No specific builds other than the production build will be tested.

## Facilities, data, and resources provision

### Test environment

As this testing effort is focused on user stories in a production environment emulating a small business, access to complex distributed systems will not be required. See section *7. Test Environment Plan* for more details.

### Testing Requirements

Each person involved in testing will need the following access:

* A personal computer with access to the Internet
* Access to QuickBooks for Mac 2016
* Access to Microsoft Excel to raise defects

### Data Requirements

Testing of the application will be done from a fresh application install and a fresh user profile, emulating a first-time user’s path through the functions of the program. No special data requirements are needed unless otherwise specified in a test case, in which the appropriate reference files and/or data files will be attached.

### Resources & Skills

* A resource with black-box and regression testing skills, and test case design and execution skills

## Testing Tools

The following tools will be used for testing:

| Process | Tool |
| --- | --- |
| Test case creation | Microsoft Excel |
| Test case tracking | Microsoft Excel |
| Test case execution | Manual |
| Test case management | Microsoft Word |
| Defect management | JIRA |

## Testing Metrics

Metrics will be captured to better manage the QA process. This will also be insightful to the developers and other stakeholders involved, offering a snapshot of the overall health and stability of the application under test. The metrics can be used for reports during and after the various test phases to provide feedback for process and tooling improvements. Some metrics to be tracked include:

* Number of defects raised
* Percentage of issue reports against total number of tests
* Defects against each module of application
* Total number of tests with PASS or FAIL or PENDING statuses
* Work time spent on issues raised

# Functions List

This is a top-level checklist of all the major functions, sub-functions, and sub-sub-functions that will be included for testing. This will include menus, commands, actions, options, or other capabilities in the program. They are to be listed in outline form with the main goal being that the list will be a detailed map of the program’s structure and capabilities.

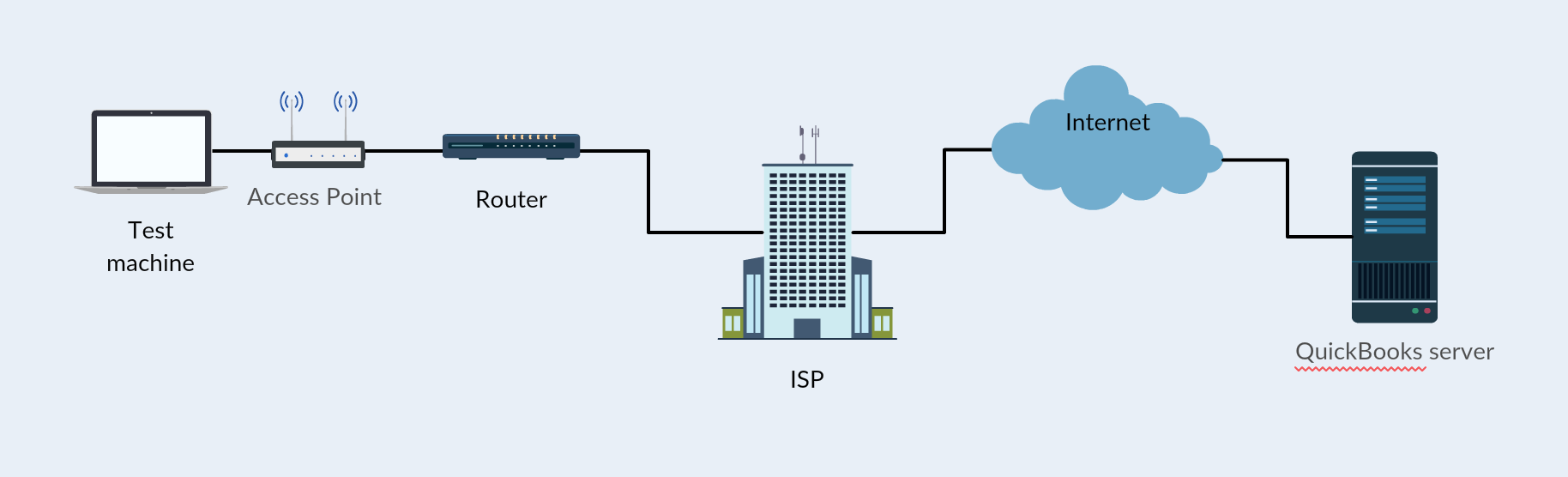
This will also serve as a visual representation of the application to organize testing efforts and the program’s function layout.

The functions list itself will be organized in a hierarchical manner as the program has a UI that is menu-driven, rather than a conceptual one based on related features. The functions list is stored in the **Function\_List.docx** document.

# Test Environment Plan

## Test Environment Chart

The following diagram illustrates the expected connection from the test machine to the QuickBooks server(s) for authentication and synchronization:

**

## Test Environment Details

### Roles and Responsibilities

|  |  |  |
| --- | --- | --- |
| Role | Staff Member | Responsibilities |
| Tester | Vincent Leung | Responsible for the overall testing effort, from test plan design and implementation, test case design and execution, and drafting of issue reports |

### Hardware and Firmware

Testing will be conducted on the tester’s personal machine updated with the latest OS X version and security updates. The machine will be an Apple MacBook running OS X 10.11.4. This machine will be the sole testing environment for the entire duration of the testing effort. No external peripherals will be connected to the machine except for the following:

* Wireless Bluetooth mouse

### Software

The following list will include the application under test as well as tools to manage testing documentation and reports. To avoid potential integration issues, no third-party monitoring or diagnostic software will be used other than the built-in system tools.

* QuickBooks for Mac 2016
* Microsoft Word 2016
* Microsoft Excel 2016

### Other Materials

The software manual will be referenced from the Support section of the Publisher’s website. Test data may include sample accounting and bookkeeping databases.

## Establishing Environment

The testing environment will be a user client connected to the Internet as per application requirements. The test machine must meet or exceed minimum system requirements.

|  |  |  |
| --- | --- | --- |
| Task | Requirements | Responsibility |
| Obtain machine with OS X 10.10 or higher | Must have access to administrative rights to install programs | Vincent Leung |
| Obtain copy of QuickBooks 2016 application | Must be production commercial copy | Vincent Leung |
| Install QuickBooks 2016 | Must have access to Internet | Vincent Leung |

## Environment Control

Control measures placed on the test environment will include the following:

* Only the tester will have access to the test machine and the application under test. This would minimize or eliminate issues pertaining to multiple users on one user profile account on the machine.
* System and application monitoring will include tools built into OS X to monitor performance issues and crash logs if necessary, using diagnostic tools such as Activity Monitor and Console.

# Assumptions and Dependencies

## Assumptions

This single test cycle was designed with a single tester executing the test cases. Any outside QA communication will be isolated to correspondence with the QuickBooks community management, who would then be able to forward any relevant communications with the development team. There will be no traditional escalation matrix or bug fixing processes as this is a private QA effort on proprietary software.

## Dependencies

Testing will be done in a production environment. Personnel dependencies include tester personnel experienced in executing and validating test cases. Hardware dependencies are that the application will be installed and run on a system that has administrator privileges to install or modify files. The test machine must have a modern version of OS X installed with enough hardware requirements to meet the following system specifications as designated by the developer:

**QuickBooks for Mac 2016**

**Minimum system requirements**

* Mac OS X v10.10 (Yosemite) and v10.11 (El Capitan)
* Intel processor, Core 2 Duo or higher
* Multiuser Server: Intel Core 2 Duo or higher processor to run server
* 2 GB RAM (4 GB RAM recommended)
* 250 MB of available disk space
* CD/DVD drive (or Internet connection) for installation
* Printer: 100% Macintosh-compatible printer, if you plan to print invoices, checks, deposit slips, lists, purchase orders, mailing labels, reports, or graphs
* Checks: Use Intuit Checks if you plan to print checks. Canadian Image Ready Cheques are not supported
* Product registration required

# Entry and Exit Criteria

Because this software product is already in production, there will be a single cycle of testing, and therefore, one set of entry and exit criteria. No written approval will be required to commence or conclude testing as per these requirements. These criteria will be defined by the following:

**Entry criteria**

* Test requirements are established
* Resources are identified
* Timeline is established with completion date

**Exit criteria**

* Verify all planned tests have been run
* Verify test coverage requirement has been met, including high risk areas
* Verify there are no critical issues that have not been raised
* Verify testing activities have been completed within the projected timeline

# Administrative Plan

## Test Milestones and Schedule

The following are the testing schedule milestones:

| Milestone | Planned End Date | Actual End Date | Resource |
| --- | --- | --- | --- |
| Phase 1 – Test plan design | 3/16/16 | 4/7/16 | Vincent Leung |
| Phase 2 – Test plan implementation | 3/25/16 | 4/7/16 | Vincent Leung |
| Phase 3 – Acceptance testing | 4/1/16 | 4/8/16 | Vincent Leung |
| Phase 4 – Test case execution | 4/5/16 | 4/15/16 | Vincent Leung |

## Training

The following training requirements have been identified to ensure testing can commence:

| Training Requirement | Staff | Date |
| --- | --- | --- |
| Tester is to familiarize oneself with the application under test by systematically researching the product manual and Help section of the application, as well as outside relevant resources or documentation | Vincent Leung | 3/11/16 |
| Quality Assurance best practices are to be applied to test plan design and execution | Vincent Leung | 3/11/16 |

## 

## Defect Management

Defects will be logged under issue reports that are to be aggregated in the Issue\_Reports.xlsx spreadsheet. This is to be a document to be shared amongst the QA team. Each issue will include the following data necessary for defect management and resolution:

* **Issue ID** – This is a label used for internal tracking purposes.
* **Issue Title** – This is a one-line title summarizing the issue and the area in which it was discovered.
* **Issue Type** – This categorizes the issue as Technical, Functional, or UI.
  + *Technical* – Issues that cause application crashes, freezing, hanging, or produces error messages or other malfunctions.
  + *Functional* – Issues that produce illogical or unexpected behavior different from its intended outcome.
  + *UI* – Issues with inconsistencies in look or layout, i.e., those that affect the graphical or presentation layer.
* **Severity** – This sets the significance and priority of the issue.
  + *Critical* - Issues that are mission-critical to the core functionality of the application and for which there are no workarounds.
  + *High* – Issues that are related to the core functionality of the application, but which would be acceptable to be fixed in subsequent builds.
  + *Medium* – Issues that affect functionality, but for which there may be workarounds that allow the user to continue in a degraded fashion.
  + *Low* – Issues that do not affect core functionality, or which may be annoyances that may or may not be fixed.
* **Frequency** – This is the regularity with which the issue happens.
  + *Every Time* – Issues that are 100% reproducible under specific conditions.
  + *Occasionally* – Issues that happen more than once, but are not 100% reproducible.
  + *Once* – Issues that cannot be reproduced or are one-off occurrences.
* **Actions Performed** – This is a list of steps that must be taken to bring about the issue raised. The steps must be clear, concise, and include all necessary information to show the issue.
* **Expected Result** – This is the intended outcome of performing the actions aforementioned.
* **Actual Results** – This is the unexpected outcome that deviates from the intended results.
* **Error Message** – This lists any error messages resulting from the reported issue.
* **Test Data** – This is any required data that must be inputted in the Actions Performed steps to bring about the issue.
* **Additional Info** – This lists any other pertinent information necessary in documenting the issue.

A matching issue report will be raised under the appropriate project in the JIRA platform. The raised issues will be assigned to the appropriate developer or individual by the developer lead or project manager. The issue reports will have one of the following statuses when entered into the system:

* **Pending** – This is the status that the issue report has by default
* **Assigned** – This status is set when the issue report is assigned to a programmer or project manager
* **As designed –** This status marks that the application is behaving as intended
* **Fixed** – This marks the issue as resolved
* **Deferred** – This status acknowledges the issue, but for management reasons will not be resolved in the current cycle of development

Those issues that are marked as **Pending** by the developer team will be added to a Known Issues list that is to be created on the Google Sheets platform. This web-based spreadsheet will be used internally as well as for dissemination to third-party testers or contractors for external testing purposes.

# Test Deliverables

The following additional documents have been referenced in the creation of this document and will also serve as a list of deliverables for the testing effort.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Document name** | **Version** | **Comments** |
| 1 | Test\_Plan.docx | 1.1 | This document itself |
| 2 | Issue\_Reports.xlsx | 1.0 | List of defect reports filed during testing |
| 3 | Test\_Cases.xlsx | 1.0 | Comprehensive list of test cases accounting for coverage of testing focus |
| 4 | Function\_List.docx | 1.0 | High-level list of application functions under test |