**Project Name**

***Test Plan***

Document Change History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version Number** | **Date** | **Contributor** | **Description** |
| V1.0 |  |  | What changes (additions and deletions) were made for this version? |
|  |  |  |  |
|  |  |  |  |

**\*\* Note to Document Author**

Blue in this document is directed at the template user to describe processes, build standards and help build the document from the template. Blue text should be removed prior to completion.

Red text in this document is used to indicate replacement of specific content. Red text should be replaced with black text prior to completion

Items in *italics* are intended to be modified, added to, etc, as they are generally examples as to the type of information expected.

**Table of Contents**

1 Introduction 3

1.1 Scope 3

1.1.1 In Scope 3

1.1.2 Out of Scope 3

1.2 Quality Objective 3

1.2.1 Primary Objective 3

1.2.2 Secondary Objective 3

1.3 Assumptions for Test Execution 4

1.4 Constraints for Test Execution 4

1.5 Definitions 5

2 Environment Needs 6

2.1 Test Environment 6

2.1.1 Hardware 6

2.1.2 Software & Services 6

3 Test Methodology 6

3.1 Purpose 6

3.1.1 Overview 6

3.1.2 Proof of Concept Testing 6

3.1.3 Unit Testing 7

3.1.4 Quality Assurance Testing 7

3.1.5 System Integration Testing 7

3.1.6 Regression Testing 7

3.1.7 User Acceptance Testing 8

3.2 Test Bug Management 8

3.3 Test Completeness 8

3.3.1 Standard Conditions: 8

3.4 Deliverables Matrix 8

3.5 Documents 9

3.5.1 Test Plan 9

3.5.2 Test Schedule 9

3.5.3 Requirements Traceability Matrix 9

4 Terms/Acronyms 10

# Introduction

This test approach document describes the appropriate strategies, process, workflows and methodologies used to plan, organize, execute and manage testing of software projects within caBIG.

## Scope

Describe the current test approach scope based on your role and project objectives.

### In Scope

*The <project name>Test Plan defines the unit, integration, system, regression, and User Acceptance testing approach. The test scope includes the following:*

* Testing of all functional, application performance, security and business requirements.
* End-to-end testing and testing of interfaces of all systems that interact with the <system name>.

### Out of Scope

The following are considered out of scope for <project name> systemTest Plan and testing scope:

* Functional requirements testing for systems outside <application name>

## Quality Objective

### Primary Objective

This section may use the test below or something very high level to describe the primary testing objective.

A primary objective of the various phases of testing is to: ***assure that the system meets the full requirements.*** At the end of the project development cycle, the user should find that the project has met or exceeded all of their expectations as detailed in the requirements.

### Secondary Objective

This section may use the test below or something very high level to describe the secondary testing objective.

The secondary objective of the various phases of testing is to: ***identify and expose all issues and associated risks, communicate all known issues to the project team, and ensure that all issues are addressed in an appropriate matter before release.*** As an objective, this requires careful and methodical testing of the application to first ensure all areas of the system are scrutinized and, consequently, all issues (bugs) found are dealt with appropriately.

## Assumptions for Test Execution

* This section will contain testing assumptions

## Constraints for Test Execution

* This section will contain testing constraints

## Definitions

Add/modify definitions to clarify IT and business terms involving quality assurance.

*Bugs: Any error or defect that cause the software/application or hardware to malfunction and fail to meet the requirements.*

**Defining urgency, impact, & priority of bug resolution**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | IMPACT | | | |
|  |  | **HIGH**  Impacts operations, affects > 50% of users, interferes with core business functions or loss or potential loss of mission critical data | **MEDIUM**  Limited time available to resolve or manual controls exist, affects < 50% of users, interferes with non-core activities or functions that do not affect the entire company, interferes with normal completion of work | **LOW**  Workaround is available, affects a single user, interferes with recreational or non-business related use |
| URGENCY | **HIGH**  Requires immediate attention, event is underway and cannot be stopped or changed AND immediate action could resolve the issue | **PRIORITY = Critical**  Resolution time = < 1 hour | **PRIORITY = High**  Resolution time = < 8 hours | **PRIORITY = Medium**  Resolution time = < 24 hours |
| **MEDIUM**  Requires attention, event is scheduled to occur but can be postponed or is far enough away to allow response without loss of productivity | **PRIORITY = High**  Resolution time = < 8 hours | **PRIORITY = Medium**  Resolution time = < 24 hours | **PRIORITY = Low**  Resolution time = < 48 hours |
| **LOW**  Issue is an inconvenience, no scheduled completion time is required and normal work can continue until responding | **PRIORITY = Medium**  Resolution time = < 24 hours | **PRIORITY = Low**  Resolution time = < 48 hours | **PRIORITY = Planning**  Resolution time = Planned |

*Enhancement:*

*1) Any alteration or modification to the existing system for better workflow and process.*

*2) An error or defect that causes the software/application or hardware to malfunction.*

*Where 1) and 2) is NOT included in the requirements can be categorized as an enhancement.*

# Environment Needs

## Test Environment

### Hardware

Include the minimum hardware requirements that will be used to test the Application.

Testing will have access control to one or more application/database servers separate from any used by non-test members of the project team. Testing will also have access control to an adequate number of variously configured PC workstations to assure testing a range from the minimum to the recommended client hardware configurations listed in the project’s Requirements, Functional Specification and Design Specification documents.

### Software & Services

In addition to the application and any other customer specified software, the following list of software should be considered a minimum:

1. *Windows Workstation*
2. *MS Office*
3. *MS Exchange*

# Test Methodology

## Purpose

### Overview

The purpose of the Test Plan is to achieve the following:

1. *Define testing strategies for each area and sub-area to include all the functional and quality (non-functional) requirements.*
2. *Divide Design Spec into testable areas and sub-areas (do not confuse with more detailed test spec). Be sure to also identify and include areas that are to be omitted (not tested) also.*
3. *Define bug-tracking procedures.*
4. *Identify testing risks.*
5. *Provide testing Schedule*.

### Proof of Concept Testing

**Definition:** Proof of concept testing provides the opportunity to try out a design in a controlled environment and to identify any potential major problems or challenges prior to conducting a pilot with users.

The following will be tested during the proof of concept testing:

1. *Prove the system is capable of XXXXX*

Development will typically create a non-functioning prototype of the UI components to evaluate the proposed design. Usability testing can be coordinated by testing, but actual testing must be performed by non-testers (as close to end-users as possible). Testing will review the findings and provide the project team with its evaluation of the impact these changes will have on the testing process and to the project as a whole.

### Unit Testing

**Definition:** Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation.

The following must be unit tested during unit testing:

1. *XXXXX stored procedure*
2. *XXXXX COBOL program*

### Quality Assurance Testing

**Definition:** A planned and systematic pattern of all actions necessary to provide adequate confidence that the product optimally fulfills customers' expectations, i.e. that it is problem-free and well able to perform the task it was designed for.

The following must be tested during QA testing:

1. *Principia file generates matching production*
2. *The XXXXX screen’s ‘Continue’ button transitions to the XXXXX screen*

### System Integration Testing

**Definition:** The process of testing to ensure that all related systems exchange data seamlessly, verifying a system’s ability to operate as expected with other systems within the same environment

The following will be tested during SIT:

1. *Data transfer from BankMate to Principia during Trade Up Load*
2. *Principia to DWH3 during XXXX process*

### Regression Testing

**Definition:** The process of testing changes to programming to ensure older programming still works with new changes.

The following will be tested during regression testing:

1. *The XXXX program (unmodified) continues to function as intended with the input from the new XXXX program.*

During the repeated cycles of identifying bugs and taking receipt of new builds (containing bug fix code changes), there are several processes which are common to this phase across all projects. These include the various types of tests: functionality, performance, stress, configuration, etc. There is also the process of communicating results from testing and ensuring that new drops/iterations contain stable fixes (regression).

### User Acceptance Testing

**Definition:** This a phase of software development in which the software is tested in the "real world" by the intended audience.

The following will be tested during UAT:

1. *XXXX file contains the expected data after XXXX process completes*
2. *Current functionality of XXXX remains in tact*

## Test Bug Management

Describe the method of managing bugs found during testing, or identify the document/system managing them.

## Test Completeness

Testing will be considered complete when the following conditions have been met:

### Standard Conditions:

1. *When approved by the Business Owner/Sponsor*
2. *Script execution of all test cases in all areas have passed.*
3. *Automated test cases have in all areas have passed.*

## Deliverables Matrix

Below is the list of artifacts that are process driven and should be produced during the testing lifecycle. Certain deliverables should be delivered as part of test validation, you may add to the below list of deliverables that support the overall objectives and to maintain the quality.

This matrix should be updated routinely throughout the project development cycle in you project specific Test Plan.

|  |
| --- |
| **Deliverable** |
| **Documents** |
| Test Approach |
| 🡪 Test Plan |
| 🡪 Test Schedule |
| **Test Case / Bug Write-Ups** |
| Test Cases / Results |
| Test Coverage Reports |
| Bug tracker reporting |
| **Reports** |
| Test results report |
| Test Final Report - Sign-Off |

## Documents

### Test Plan

The Test Plan is derived from the Test Approach, Requirements, Functional Specs, and detailed Design Specs. The Test Plan identifies the details of the test approach, identifying the associated test case areas within the specific product for this release cycle.

The purpose of the Test Plan document is to:

1. Specify the approach that Testing will use to test the product, and the deliverables (extract from the Test Approach).
2. Break the product down into distinct areas and identify features of the product that are to be tested.
3. Specify the procedures to be used for testing sign-off and product release.
4. Indicate the tools used to test the product.
5. List the resource and scheduling plans.
6. Indicate the contact persons responsible for various areas of the project.
7. Identify risks and contingency plans that may impact the testing of the product.
8. Specify bug management procedures for the project.
9. Specify criteria for acceptance of development drops to testing (of builds).

### Test Schedule

This section is not vital to the document as a whole and can be modified or deleted if needed by the author.

### Requirements Traceability Matrix

Not currently used, but suggested.

A Requirements Traceability Matrix (RTM) which is used to link the test scenarios to the requirements and use cases is a required part of the Test Plan documentation for all projects. Requirements traceability is defined as the ability to describe and follow the life of a requirement, in both a forward and backward direction (i.e. from its origins, through its development and specification, to its subsequent deployment and use, and through periods of ongoing refinement and iteration in any of these phases).

Attached is a sample basic RTM which could provide a starting point for this documentation. The important thing is to choose a template or document basis that achieves thorough traceability throughout the life of the project.



# Terms/Acronyms

The below terms are used as examples, please add/remove any terms relevant to the document.

| TERM/ACRONYM | DEFINITION |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |