Test Execution and Results Plan Version 1.0

<Project or Solution Name>

U.S. Department of Housing and Urban Development

<Month, Year>

Solution Information

	Information
Solution Name	*<Solution Name>*
Solution Acronym	*<Solution Acronym>*
Project Number / Task Number	*<From New Core Accounting System>*
Document Owner	*<Owner Name>*
Primary Segment Sponsor	*<Primary Segment Sponsor Name>*
Version/Release Number	*<Version/Release Number>*

Document History

<Provide information on how the development and distribution of the Test Execution and Results Plan document is controlled and tracked. Use the table below to provide the version number, date, author, and a brief description of the reason for creating the revised version.>

Version No.	Date	Author	Revision Description

*[This document is a template of a **Test Execution & Results Plan** document for a project. The template includes instructions to the author, boilerplate text, and fields that should be replaced with the values specific to the project.*

- *Blue italicized text enclosed in square brackets ([text]) provides instructions to the document author, or describes the intent, assumptions and context for content included in this document.*
- *Blue italicized text enclosed in angle brackets (<text>) indicates a field that should be replaced with information specific to a particular project.*
- *Text and tables in black are provided as boilerplate examples of wording and formats that may be used or modified as appropriate to a specific project. These are offered only as suggestions to assist in developing project documents; they are not mandatory formats.*

When using this template for your project document, it is recommended that you follow these steps:

1. *Replace all text enclosed in angle brackets (e.g., <Project Name>) with the correct field values. These angle brackets appear in both the body of the document and in headers and footers. To customize fields in Microsoft Word (which display a gray background when selected):*
    a. *Select File>Properties>Summary and fill in the Title field with the Document Name and the Subject field with the Project Name.*
    b. *Select File>Properties>Custom and fill in the Last Modified, Status, and Version fields with the appropriate information for this document.*
    c. *After you click OK to close the dialog box, update the fields throughout the document with these values by selecting Edit>Select All (or Ctrl-A) and pressing F9. Alternatively, you can update an individual field by clicking on it and pressing F9. This must be done separately for Headers and Footers.*
2. *Modify boilerplate text as appropriate to the specific project.*
3. *To add any new sections to the document, ensure that the appropriate header and body text styles are maintained. Styles used for the Section Headings are Heading 1, Heading 2 and Heading 3. Style used for boilerplate text is Body Text.*
4. *To update the Table of Contents, right-click and select "Update field" and choose the option- "Update entire table"*
5. *Before submission of the first draft of this document, delete this "Notes to the Author" page and all instructions to the author, which appear throughout the document as blue italicized text enclosed in square brackets.*

Contents

Exhibit List

1. Introduction

The Test Execution and Results Plan defines the testing approach to the *<insert Project/Solution>*. It focuses on describing the test strategy, processes, and activities that will incrementally deliver capabilities. Major changes to the test strategy will drive updates to this document.

The following documents should be referenced for additional information on the program:

- **Release Backlog (RBK)** – Prioritized list of the requirements (user stories) that are initially planned to develop the capabilities detailed in the Capabilities and Constraints document.
- **Project Oversight Plan (POP)** –The POP includes details such as the artifacts and reviews required, status updates and reporting, as well as release schedule.
- **Capabilities and Constraints (CAC)** – Set of capabilities and constraints of the features that will be delivered at the release level.
- **Team Process Agreement (TPA)** and **Definition of Done (DoD)** – Agreement among team members regarding the team structure, responsibilities, and practices, as well as additional activities such as distribution of work, and version control of source code and other artifacts. The DoD documents the criteria by which a user story, or piece of functionality, will be considered "done" during a sprint.
- **Critical Task Schedule** – For each release, the critical tasks and dependencies for delivery of committed functionality.

1.1. Purpose

This document provides testing approach information for the *<insert Project/Solution>* including test types, objectives, roles and responsibilities, environments, test tools, testing process, issue resolution processing, and test reporting, and references to test-related artifacts.

- The tests covered in this document include:
- Unit Test (UT)
- System Integration Test
- Section 508 Compliance Test (508)
- User Acceptance Test (UAT)
- Independent Verification and Validation (IV&V)

2. Requirements and Capabilities

The capabilities detailed in the Capabilities and Constraints (CAC) document for a release represent the high level requirements. These are expressed as user stories, which provide a business-oriented description of a feature the system should offer. User stories are expected to be modified, deleted, or added during the development and testing of each release cycle. Each user story will have a test case/test script mapped to the requirements. Acceptance criteria for each user story are the basis for test criteria for the test teams. User Stories and acceptance criteria are managed in the Backlog tool (e.g., JIRA). Manual test cases will be addressed in the Test Execution Plan for the Release.

3. Test Assumptions, Constraints and Risk

The following subsections reflect the assumptions, constraints and risks related to the testing of the solution. Unless approved by the Product Owner, these should not change throughout the development of

the system. Specific, release-level assumptions and constraints are addressed in the Test Execution Plan for the Release.

3.1. Test Assumptions

The following assumptions represent situations that are beyond the control of the project team, but the outcomes of which would influence the successful testing of the solution: <List the test assumptions for the system/solution. Include assumptions related to items such as equipment availability, resource skills, code availability, and data availability. Include testing programs/devices and programming aids as well as any components must run in parallel with the new solution to achieve functionality.>

- *<Assumption>*
- *<Assumption>*
- *<Assumption>*

3.2. Test Constraints

This section describes the constraints that are outside the control of the project team: <List the constraints that affect the testing. Include constraints related to items such as availability of equipment, trained staff, and compressed project schedule.>

- *<Constraint>*
- *< Constraint>*
- *< Constraint>*

3.3. Project-Level Testing Risks

<Identify the risks associated with the testing activities. Include a work around or mitigation plan for each of the identified risks and document the risks within the Risk Management Log.>

4. Test Approach

<Outline the overall approach for testing the release features. Once the release satisfies all of the unit/system/integration and user acceptance testing then it is ready for deployment activities, which involve submitting it for production to the HUD Test Center. The HUD Test Center can be considered the front-end of the deployment process. In addition to providing release documentation, the purpose and value of the Test Center's testing is to assure that:

- *The deployment will work in HUD-standard environment/configuration;*
- *The production installation instructions are complete and correct;*
- *The files to be deployed are available and correct.>*

4.1. Test Overview

User stories will be generated for each sprint. Each user story must be complete as prescribed by the agile approach, meaning each story will contain a testable requirement and the acceptance criteria. The agile teams develop and test requirements throughout the sprint until story-specific acceptance criteria has been satisfied. To consider a story as delivered, the product owner must explain the Definition of Done (DoD). For example: each user story will have unit tests that have no less than 90% coverage and corresponding Selenium tests that run through the happy path. This indicates that the developers must

coordinate with the automated test group to ensure tests can be written for their code so that a story may be completed within the sprint. At the UAT session, the precondition for demonstrating new functionality to the Product Owner will be a test report indicating that the code meets the DoD, either in SonarQube or some other reporting tool. For any sprint, the capabilities delivered in a predetermined interval will include the completion of testing activities prior to deployment. In addition, updates to documentation based on the changes will be completed during the sprint.

4.2. Test Types, environments, and Purpose

Testing will occur in the Development (DEV), Development/Test (DEV/TEST), Testing (TEST) and Staging (STAGE) environments. The differences in Environments are listed below:

- **DEV Environment** - Developers use their local machine to test the first cut of code integrated into the full application without the integration to interfaces.
- **DEV/TEST Environment–** This is the first stage of the Continuous Integration (CI) pipeline.
- **TEST Environment** - Internal HUD Azure subscription with access to other HUD systems. Includes most of the pieces necessary to have the application stable and perform like the final product with simulated integration.
- **STAGE Environment** - Production-like environment including most of the hardware and live integration with interfaces.

4.3. Sprint Cycle Testing Types and Objectives

Exhibit 1 details the various test types conducted throughout each sprint cycle, to include the purpose and/or objectives, the responsible personnel/organization(s) and related artifacts.

Exhibit 1: Test Types

Test Type	Environment	Purpose	Responsible Party	Artifacts
Unit Test	DEV DEV/TEST	• Performed by each developer to verify individual units of source code work as designed. • Unit Test is run locally on the VM workspace and on the CI Pipeline for every new commit to the source code. • Involves some level of automated regression testing to make sure code changes for new or changed functionality have not broken existing functionality that is not changing.	Developer	• Unit Test Cases /Scripts • Unit Test Results
System Integration Test	DEV/TEST TEST	• Integrated testing of stories based on story-specific acceptance criteria. • Performed by the System Integration Tester to verify the code works as designed in the TEST environment • Used to verify acceptance criteria of each user story as well as for all stories in each Sprint.	System & Integration Test Tester	• System Test Cases/Scripts • System Test Results

Test Type	Environment	Purpose	Responsible Party	Artifacts
		• Includes the testing of interfaces, along with negative and boundary tests. • Involves automated and manual regression testing to verify that new functionality has not broken any existing functionality of the modified code • System Integration tests will be automated to the extent possible, although some functionality will continue to require manual testing. • Apply negative testing where ever applicable to ensure code functions within the expected limits		
Section 508 Compliance Test	DEV/TEST STAGE	• Independent verification that the solution complies with Section 508 accessibility regulations. • To be conducted every sprint cycle	Developer System Integration Tester	• Section 508 Compliance Test Cases/Scripts • Test Results
User Acceptance Test	STAGE	Validates that the system meets user requirements and reports defects and issues prior to the production release.	Business Users	• Test Cases/Scripts • Test Results that identifies system change requests (SCR) or requirement numbers traceable to the test case, pass or fail status, and any comments regarding the test results.
Independent Verification & Validation	STAGE	An independent third party checks that the solution/service meets specifications and that it fulfills its intended purpose.	IV&V Testers	• IV&V assessment reports • final IV&V Test report

4.4. Other Test Activities

Exhibit 2 illustrates additional test types, activities, and events that are dependent upon and/or performed throughout each sprint cycle. The table also specifies the artifacts created during the conduct of each test type. Unless noted otherwise, the artifacts listed in the table are internal team products that are created to facilitate the development and test processes. Final test execution results are made available via formal documentation delivered prior to the Release Readiness Review (RRR).

Exhibit 2: Other Test Activities

Other Test Activities			
Test Types	**Purpose | Objectives**	**Responsible Organization(s)**	**Artifacts**
Failover and Recovery Testing	• Testing to verify how well the system (including redundant and backup systems) confronts and successfully recovers from crashes, hardware failures, or other catastrophic problems. • Testing is performed as part of the build and deploy process prior to delivery into production.	CI Team	Build and Deploy Results
Rollback Testing	• Rollback refers to the process of "unwinding" changes applied to an environment or code in order to revert to a previous state. • Testing rollback policies and procedures to verify the ability to return the production environment to a previously "known" working state during code deployments.	CI Team	Server and Load balancer Logs
Security Test	• Testing to verify that system security controls are implemented correctly, working as intended, and producing the desired outcome. • Security scans are performed during each sprint, and prior to deployment into production.	IT Security	Vulnerability Scan Results

4.5. Test Tools

<Update the content in Exhibit 3 to report the Test Tools used throughout the development of this solution.> Exhibit 3 reports the test tools used for this solution.

Exhibit 3: Test Tools

Test Tool	Purpose
Mockito 1.9.5	*Simulates external input*
Selenium 2.52.0	*Automated UI testing*
Junit 4.11	*Automated Unit Testing*
Yasca	*Automated Security Static Code Analysis*
Jacoco	*Measures Code Coverage*
PMD *Checkstyle* *Findbugs*	*Static Code Analysis*
TBD	*Performance / Load Test*
Jenkins	*Continuous Integration Server*
<insert tool>	*508 Compliance Testing*

4.6. Test Roles and Responsibilities

Exhibit 4 displays the roles and responsibilities for the testing of *<insert Project/Solution>.*

<Update the following table to reflect the roles required to successfully perform the tests. Include any special requirements, such as multiple-shift operation or key personnel.>

Exhibit 4: Roles and Responsibilities

Role	Responsibility	Responsible Party
Developer	• *Make code modifications for user story* • *Ensure test data needed to test user story is in the Development back end system or request test data via the technical SME* • *Inform team lead when Development environment deployment is needed* • *Unit test code modifications to validate that acceptance criteria are met in the DEV environment* • *Label and check-in code modifications for user story. Documents and code will be version controlled.* • *Inform Test Lead that user story code is complete and ready for deployment* • *Plan and coordinate with stakeholders to ensure proper Performance/Load testing.*	*<insert name(s)>*
System Integration Tester	• *Write test scripts including steps for each user story to verify via the acceptance criteria (manual testing only)* • *Write test scripts in the testing tool* • *Ensure test data needed to test user story is available or request test data via the developer and/or technical SME* • *Write regression test scripts to verify other functionality has not been broken* • *Execute test scripts iteratively in the TEST environment until it passes or user story is not supported in Sprint* • *Raise any issues to developer and other team members* • *Record test results including test data and screen shots (if needed)* • *Complete each test script with Pass or Fail status and work with developers on ensuring failed scripts are fixed during the script.* • *Inform team when tests are completed*	*<insert name(s)>*
508 Tester	• *Document 508-compliance test scripts (for manual tests) in the testing tool (TBD), including test steps for user stories that involve a User Interface change* • *Execute test scripts in TEST environment* • *Raise any issues to developer and other team members* • *Record test results including test data and screen shots* • *Complete each test script with Pass or Fail status and work with developers on ensuring failed scripts are fixed during the script.* • *Inform team when tests are completed.*	*<insert name(s)>*
IV&V Tester	• *Write test scripts for each user story to verify via the acceptance criteria* • *Write test scripts that will test the end to end functionality and interfaces* • *Ensure test data needed to test user story is in the STAGE back end system or request test data via the developer and/or technical SME* • *Write regression test scripts to verify other functionality has not been broken* • *Execute test scripts in the STAGE environment* • *Raise any issues to developer and other team members*	*<insert name(s)>*

Role	Responsibility	Responsible Party
	• *Record test results including test data and screen shots (if needed)* • *Complete each test script with Pass or Fail status and work with developers on ensuring failed scripts are fixed during the script.* • *Inform team when tests are completed*	
Business Owner SME	• *Verify user story statement* • *Verify RFS modifications meet the intended business goal as stated in the user story* • *Identify user stories failures (those that did not meet business goal)* • *Classify user story failures as those that need modifications and/or those that need new user story*	*<insert name(s)>*
Product Owner	• *Facilitate decisions for user story division into sprints or releases* • *Review all test results* • *Review all test cases marked as "N/A" or "not run" and decides whether to let it stand as is based on criteria recorded in the test results document and move forward* • *Decide when sprint is "done" and the project moves forward*	*<insert name(s)>*

5. Quality Assurance

5.1. Quality Assurance Overview

The largest degree of the quality assurance (QA) effort is internal to the agile teams, and includes the development, review, and continuous updating of artifacts throughout each of the sprint cycles. Quality assurance reviews are guided by specific criteria that focus on the content and clarity of particular artifacts. In addition, substantive feedback is solicited and collected during each Sprint Review, the result of which are prioritized and addressed prior to completion of the release, per Product Owner instructions.

5.2. Issue Resolution Process

It is common to discover issues with the code prior to final story acceptance. As issues are identified, they should be triaged to determine the nature of the problem. Issues related to an environment should be addressed by the Enterprise Architecture Team. Issues related to the user stories under development, or those impacting higher-level features (capabilities), must be resolved by the agile teams prior to completion of the current sprint.

On the occasion that the issue cannot be resolved prior to the end of the current sprint, a new task may be created (at the discretion of the Product Owner) to continue resolving the issue during the following sprint. Should the issue remain unresolved prior to the end of the final sprint in the release, the Product Owner will make a determination whether or not to accept the code. In some cases, the issue may be reprioritized and resolved in a future release.

Appendix A: References

<Insert the name, version number, description, and physical location of any documents referenced in this document. Add rows to the table as necessary.>

Document Name	Description	Location
<Document name and version number>	*<Document description>*	*<URL to where document is located>*
<Product Backlog>		
<Team Process Agreement / Definition of Done>		
<Test Execution Plans (for each Release)>		
<Capabilities and Constraints>		
<Project Oversight Plan>		
<Critical Task Schedule>		

Appendix B: Key Terms

<Insert the terms and definitions for terms and acronyms relevant to the content presented within this document.>

Term	Definition
[Insert Term]	*<Provide definition of term and acronyms used in this document.>*