Standard Operating Procedure (SOP)
BRICS – Master Test Plan (MTP)

Document Information

Document Owners: Matthew McAuliffe, Yang Fann, and Dominic Nathan

Organizations: NINDS DIR ITBP, CIT OIR ISL BIRSS, CNRM

Approval / Distribution Process

The SOP approval/distribution process is as follows:

1. The SOP author sends the SOP SharePoint link to their peers/subject matter experts (SMEs) for review.
2. After editing, the SOP author decides whether the SOP is ready for approval. If the SOP is ready, the author adds the SOP to the ITBP Manager meeting agenda.
3. At the ITBP Managers meeting or via email, NINDS Management formally approves/disapproves the SOP.
4. The SOP Author sends the SOP link to all people on the Distribution List.

NINDS Approval

This Standard Operating Procedure (SOP) is approved for distribution and implementation as of the Director ITBP approval date listed below. NINDS ITBP management is authorized to conduct periodic audits to ensure compliance with this procedure. Requests for corrections or changes to any part of this procedure must be submitted to the Document Owner to review. Exceptions to any procedure must be approved by the ITBP Management and documented.

Approved By:

Name	Title	Organization	Approval Date
Yang Fann	IT Director	NINDS DIR ITBP	03/28/19
Matthew McAuliffe	BIRSS Chief	CIT OIR ISL BIRSS	03/28/19
Dominic Nathan	Informatics Core Director	CNRM	03/28/19
Mark Edwards	IT Manager	NINDS DIR ITBP	03/28/19

Name	Title	Organization	Approval Date
Willy Calderon	ISSO	NINDS DIR ITBP	03/28/19

Peer Reviewers

This Standard Operating Procedure was reviewed by the peers (i.e., subject matter experts) listed below. The procedure will be reviewed by the peer reviewers at least annually.

Reviewed By:

Name	Title	Organization	Date
Tsega Gabremichael	Team Lead	CIT OIR ISL BIRSS	03/27/19
Leonie Misquitta	Sr Scientific Advisor	CIT OIR ISL BIRSS	03/27/19
Dominic Nathan	Informatics Core Director	CNRM	03/27/19

Distribution List

This Standard Operating Procedure impacts the individuals on this Distribution List. The SOP author should notify everyone on this list about changes to this SOP *within one week* of NINDS approval.

Distributed To:

Name / Department / Group / Team
Yang Fann
Matthew McAuliffe
Dominic Nathan
Willy Calderon

Table of Contents

1 PURPOSE, BACKGROUND AND SCOPE

1.1 Purpose

This document is a high-level overview defining the testing strategy for the BRICS release of the groups listed below:

- NINDS DIR Clinical Informatics Development Team
- CIT OIR ISL BIRSS Development Team

This explains the testing approach for unit, iteration, integration and system which cover different types of testing like functional, regression and user acceptance testing.

Its objective is to communicate project-wide quality standards and procedures. It portrays a snapshot of the project as of the end of the planning phase. This document will address the different standards that will apply to the end-to-end testing of the specified application. Technical teams will utilize testing criteria under the white box, black box, and system-testing paradigm. This paradigm will include, but is not limited to, the testing criteria, methods, and test cases of the overall design.

The intended audience of the master test plan includes business owners, critical partners, application and infrastructure resources assisting with testing, and development team members responsible for the creation of testing deliverables and execution of test scripts. The detailed test plan is meant to provide guidance in testing the overall look, feel, and functionality of the BRICS and its associated systems such as CiSTAR, CASA, and ProFoRMS.

1.2 Background

BRICS is a collaborative and extensible web-based system to support the collection of research studies and clinical trials, using a set of modular components that cover all stages of the research life cycle. And because BRICS is un-branded and un-associated with a disease or organization, it can be custom-tailored for any partner program.

Building on concepts originally developed for the National Database for Autism Research (NDAR), BRICS began as a joint effort between the Department of Defense (DOD), Department of Health and Human Services (HHS)/National Institutes of Health (NIH): National Institute of Neurological Disorders and Stroke (NINDS), and the Center for Information Technology (CIT). Since then, it has continued to evolve over a rich 10-year history, through collaborations with partners like the Parkinson's Disease Biomarkers Program (PDBP) and the Federal Interagency Traumatic Brain Injury Research (FITBIR).

BRICS offers researchers a secure platform and a suite of tools to promote standardization, communication, and collaboration across the research

community and a data repository to hold genetic, phenotypic, clinical, and medical imaging data. These plug-and-play modules can be shared across disease categories or deployed and branded independently, depending on the needs of your program. Together, they provide a combination of web-based functionality and downloadable tools that support data definition, data contribution, and data access throughout the research life cycle:


Data Mapping Tool: The Data Mapping Tool allows users to map their data to the BRICS data dictionary's common data elements. In addition, it builds a BRICS compliant file that can be validated and uploaded into BRICS based repository.


Data Dictionary: Define and Validate Data: The Data Dictionary provides functionality for creating, managing, and searching data dictionary components (data elements and form structures), as well as services for validating research data against the standardized common data elements (CDEs).


Data Repository: Study Management and Data Submission: The Data Repository is the central hub of the BRICS system, providing functionality for defining and managing study information, and for contributing, uploading, and storing the research data associated with each study.


Meta Study: A Meta Study contains findings from other studies that can be aggregated by researchers to conduct additional analysis. The information within the Meta Study can be referenced in publications.


GUID: Global Unique Identifier: The GUID is a Global Unique Identifier for each study participant that allows researchers to aggregate and share a participant’s data without exposing personally identifiable information (PII). The GUID is made up of random alpha-numeric characters and is not generated from PII/PHI.


ProFoRMS: Protocol Management and Data Capture:

With this clinical trial/research module, researchers can define electronic case report forms, schedule and collect clinical data, and then export, analyze, and report on the data. This module has been developed in collaboration with CNRM Informatics Core and is based on NICHD's Clinical Trail Database (CTDB).


Query and Export Data: By combining the power of the GUID and the use of a standard vocabulary via CDEs, the query tool provides a powerful means to sift through volumes of aggregated research data across studies.


MIPAV Imaging Data Submission Tool: The MIPAV (Medical Image Processing, Analysis, and Visualization) application enables quantitative analysis and viewing of medical images, such as PET, MRI, CT, or microscopy. A MIPAV plugin is used to package and submit image data of many formats (i.e. DICOM, NIFTI, Analyze, AFNI and many others) into BRICS.


Account Management: This is the application for creating, approving, and managing user accounts, including management of access controls, roles, permissions groups, and authorization to other BRICS modules.

1.3 Scope

Testing for the BRICS release will take place in DCB pre-production environments. JIRA will be used for creation/execution of the test cases.

The testing scope of this effort includes unit, white box, black box, functional, system, 508 (if applicable), data migration (if applicable), performance (if applicable), smoke, security, and user acceptance testing. In addition, regression testing will be performed towards the end of each sprint in order to make sure that the base functionality is preserved.

1.3.1 Unit Testing

Unit testing for the BRICS release will be performed to determine that the solution is functioning correctly at the method/function level for simple functions such as the creation, updating, restoration, and deletion of data entities. The unit tests are meant to achieve the following objectives:

Unit Testing is done at the source or code level for language-specific programming errors such as bad syntax, logic errors, or to test functions or code modules. The unit test cases shall be designed to test the validity of the program's correctness.

- Ensure expected positive results/outputs of the simplest methods and functions
- Ensure expected negative results/outputs of the simplest methods and functions
- Validate exception handling

1.3.2 White Box Testing

In white box testing, the user interface is bypassed. Inputs and outputs are tested directly at the code level and the results are compared against specifications. This form of testing ignores the function of the program under test and will focus only on its code and the structure of that code. Test case designers shall generate cases that not only cause each condition to take on all possible values at least once, but that cause each such condition to be executed at least once.

1.3.3 Black Box Testing

Black box testing typically involves providing expected data for every field to verify that it functions as an end-user would expect. We have decided to perform Equivalence Partitioning and Boundary Value Analysis testing on our application.

a) Equivalence Partitioning:

In considering the inputs for our equivalence testing, the following types will be used:

- Legal input values – Test values within boundaries of the specification equivalence classes. This shall be input data the program expects and is programmed to transform into usable values.
- Illegal input values – Test equivalence classes outside the boundaries of the specification. This shall be input data the program may be presented, but that will not produce any meaningful output.

The equivalence partitioning technique is a test case selection technique in which the test designer examines the input space defined for the unit under test and seeks to find sets of input that are, or should be, processed identically.

b) Boundary Value Analysis:

Boundary value analysis is a technique for test data selection. A test engineer chooses values that lie along data extremes. Boundary values include maximum, minimum, just inside boundaries, just outside boundaries, typical values, and error values. The expectation is that, if a system works correctly for these extreme or special values, then it will work correctly for all values in between. An effective way to test code is to exercise it at its natural boundaries

1.3.4 Functional Testing

Functional testing for the BRICS release will be performed to determine that the solution is functioning correctly, as per the use cases, requirements traceability matrix, and the system requirements and design specifications documents.

1.3.5 System Testing

The objective of system testing is to ensure that the new system performs as expected with regards to functionality. This includes testing how the system works with various other internal/external systems or services with which the system interfaces.

1.3.6 Section 508 Compliance Testing

The purpose of 508 testing is to evaluate websites' approach to displaying text, non-text elements, frames/iframes, colors, data tables, images and other front-end content to ensure that it performs as expected with regards to Section 508. The test scripts are designed by the tester based on the Section 508 guidelines applicable to the type of media being developed (e.g., website, operating system, multimedia).

1.3.7 Performance Testing

The main objective of load and performance testing is to record page load times for all of the pages in the system.

1.3.8 Regression Testing

Regression testing objectives will be to make sure that the previously coded requirements have not been adversely affected by recent changes to the code.

Regression testing is prepared based on the initial round of testing. Once the changes are incorporated after the initial testing, the regression testing

is performed to confirm that the application has not regressed to a less functional state that in the previous build. Selected Test scripts from each type of testing like functionality, ad hoc, usability etc. are included in test plan for regression testing. The test plan also includes the expected results for regression test scripts. After the testing is done the results are compared with the initial test results. The test results are used to prepare the final regression test report.

1.3.9 Smoke Testing

The purpose of the smoke test is to ensure that an application or system is stable enough to enter testing in the currently active test phase. It is a non-exhaustive set of tests to ensure that most major functions are working before initiating the more exhaustive set of tests. It is also a method of testing the production environment without impacting production data.

1.3.10 Security Testing & Evaluation

The purpose of security testing & evaluation is to ensure common understanding of security requirements and ensure that they are assessed throughout the development process. Basic security tests to ensure that core functionality and logical controls adhere to NIH and National Institute of Standards and Technology (NIST), such as number of logins attempts and preventing Structured Query Language (SQL) injections in open form fields will be the responsibility of the test team. Identification of physical controls and completion of the system security plan (SSP) will be led by security liaisons within the Information Security Program (ISP).

1.3.11 User Acceptance Testing

The objective of user acceptance testing is to ensure that the stakeholders can formally perform comprehensive functional tests on the system prior to closing out a sprint as well as accepting delivery or ownership of the system. This part of the testing process helps ensure quality assurance and formal acknowledgment by the operations team that the system meets their expectations and fulfills the originally agreed upon requirements

1.3.12 Automation Testing

The objective of implementing automation testing for the BRICS release is mainly focused on the following items:

a) Covering Regression scenario which will add value to regression testing

b) Reduction of manual testing effort

The test scenario-based approach is used to identify the candidate for automation. Script updates and Script maintenance will be performed during each release to ensure the stability of the test automation regression suite.

2 RESOURCE REQUIREMENTS

This section summarizes the environment, tools, personnel resources, and skills required to perform the testing identified within the scope of this document.

2.1 Testing Environment

The following table provides a mapping of test types to the environments in which the testing will be performed. The BRICS Test Environment consists of the following servers:

Table 1 – Environments in which Testing will Occur

Test Type	Environment(s)
Unit Testing	▪ Development Environment
Integration Testing	▪ Development Environment
System Testing	▪ STAGE Environment
508 Testing	▪ STAGE Environment
Performance Testing	▪ STAGE Environment
Regression Testing	▪ STAGE Environment
Security Testing & Evaluation	▪ Demo Environment ▪ Production Environment
Smoke Testing	▪ Demo Environment ▪ Production Environment
User Acceptance Testing	▪ STAGE Environment ▪ DEMO Environment
Automation Testing	▪ STAGE Environment

2.2 Testing Tools

The testing tools utilized by the team are listed below.

Table 2 - Test Tools

Tool	Description	Access
JIRA	Test case management	• JIRA Site: https://dcb-jira.cit.nih.gov/secure/BrowseProjects.jspa#all • Log in with your credentials
Selenium	Test automation	• Local laptops
Unified Functional Tool (UFT)	Test automation	• NIH Laptops

2.3 Test Role, Skills and Responsibilities

The table below identifies the key staff, required skills, and responsibilities.

Table 3 - Test Roles and Skills

Role/Function	Necessary Skills (Experience, Tool Sets, Etc.)	Responsibilities
Testing Lead	▪ Experience leading a testing team ▪ Experience with Section 508 / accessibility ▪ Experience with JIRA, HP UFT (Unified Functional Testing) and/or Selenium ▪ Experience working in agile/iterative development environments	▪ Define the overall testing strategy, designing test suites, and supporting overall application compliance. Planning, managing, maintaining, coordinating, and executing the test scripts for complex, multi-tier applications ▪ Identify test data ▪ Execute test conditions and mark-off results ▪ Prepare software error reports ▪ Administrate error measurement system ▪ Ensure test systems outages/problems are reported immediately and followed up ▪ Ensure entrance criteria are achieved prior to system test start ▪ Ensure exit criteria are achieved prior to system test signoff ▪ Ensure that automated test script has been created and test cases have been identified for automation

Role/Function	Necessary Skills (Experience, Tool Sets, Etc.)	Responsibilities
Configuration Manager	▪ Experience setting up and running the software configuration management process using tools such as Ant or Maven ▪ Experience setting up and configuring a continuous integration process ▪ Experience writing, executing, and managing project build, continuous integration, deployment scripts, and processes	▪ Oversee and manage the build and continuous integration process, which ensures that all checked in code is incorporated into the build, unit tests are run as part of the build, and that each build can be uniquely identified ▪ Manage, create, and edit deployment and build scripts ▪ Maintain change history across deployment builds
Testers	▪ Experience working as part of a testing team ▪ Experience with JIRA, HP UFT (Unified Functional Testing), and Selenium ▪ Experience working in agile/iterative development environments ▪ Experience performing all the testing types outlined under section 1.3	▪ Designing automated test scripts, writing test scripts, and supporting test execution ▪ Ensure exit criteria are achieved prior to system test signoff ▪ Regularly review testing progress with test controller ▪ Raise and manage issues/risks relating to project or outside test teams' control ▪ Review and sign off test approach, plans and schedule

Role/Function	Necessary Skills (Experience, Tool Sets, Etc.)	Responsibilities
Architect	▪ System design and architecture experience with major Java technologies ▪ Experience working in agile/iterative development environments	Define coding standards, story exit/completion criteria, and enforce TDD best practices across the development team. Setup/configure all environments.
Development Leads	▪ System design and architecture experience with major Java technologies ▪ Experience leading small tracks of work within a development team ▪ Experience working in agile/iterative development environments	Enforce coding standards, story exit/completion, and unit test best practices. Support testers in the execution of test activities as needed. Review unit tests to ensure they are comprehensive. Assign defects to developers.
Developers	▪ Experience working in agile/iterative development environments	Follow coding standards, support testers in the execution of test activities as needed, write unit tests, and follow best practices.

3 ASSUMPTIONS, CONSTRAINTS, DEPENDENCIES AND RISKS

3.1 Assumptions

1) All development and testing personnel have knowledge and experience with all forms of testing outlined in this test plan.
2) Testing tools outlined in section 2.2 will be provided and hosted by NIH's Center for Information Technology (CIT).
3) All testers and developers will have access to the testing tools outlined in section 2.2.
4) Test and development leads will have the knowledge and ability to administer the testing tools outlined in section 2.2.
5) JIRA will be used to store test cases and test case results.
6) JIRA will be used to manage the defect lifecycle.
7) All functional requirements/user stories will be validated by the business owners during the use case development process.
8) Once base-lined, functional requirements/user stories will be documented in the JIRA which will be the source of requirements against which test cases will be created.
9) Each User story will have acceptance criteria that address business requirements.
10) Test cases will reference the acceptance criteria and functional requirements that they address.

3.2 Constraints

- At this point in time, there are no performance threshold or objectives requirements against which test cases can be created and executed.

3.3 Dependencies

These are the test related dependencies that are being monitored as part of this effort are documented in the table below.

Table 4 - Dependencies

Dependency	Impact if Not Provided
Availability of STAGE Environment	No BRICS functionalities can be tested
Availability of DEMO	Testing cannot be performed in an environment that does not require Open VPN
Availability of performance objectives	The team will be unable to identify reasonable performance goals against which to test
Availability of performance metrics for the current system	The team will be unable to identify reasonable performance goals against which to test

3.4 Risks

The test related risks that are being monitored as part of this effort are documented in the table below.

Table 5 - Testing Risks

Testing Risk	Contingency/Mitigation
Lack of personnel resources when testing is to begin	▪ Work with the testing team and make sure work is assigned evenly and nothing is left out. ▪ Communicate the timeline in every team meeting and look for the resources available for testing. ▪ Expedite the process of hiring a new and skilled personnel.

Testing Risk	Contingency/Mitigation
Lack of availability of required hardware, software, data or tools	▪ Coordinate with the Business Owner to make sure the real (production) data is available when it's required. ▪ Coordinate with the architect and system personnel to make sure all the required hardware (e.g., environments) is in sync and running. ▪ Coordinate with the development team and Business Owner to make sure all the software is up and running.
Late delivery of the software, hardware or tools	▪ Coordinate with the Business Owner and all responsible personnel for delivering Software, Hardware and tools. ▪ Check-up and follow up with the status on a regular basis.
Changes to the original requirements or designs	▪ Communicate with the requirements team as soon as the documents (e.g., Use Cases, Wireframes, SRDS) are ready for peer review. ▪ Clarify the enhancement and suggestion during the UAT.
Complexities involved in testing the applications	▪ Analyze the testing effort with the team and try to point out the complex testing scenarios at early stage.

4 TESTING PROCESS

4.1 Methodology

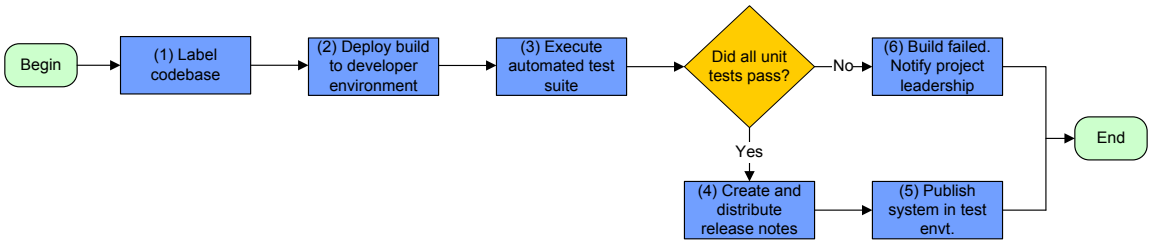
As part of the test process, the DCB project will make use of the agile/iterative approach during system development. The project aims to capture all defects early in the development process when they are easier and less costly to fix by writing the test scripts first and only allowing development of new functionality to proceed when all test scripts have passed. This testing is executed iteratively, leading to an outcome that is evolutionary and informed by action, rather than planned. As part of this approach, a process known as Continuous

Integration (CI) is employed in which all system code that is committed to the versioning repository is compiled and tested every time a change is encountered. By utilizing this automated continuous process and given a comprehensive test set, it is known immediately if any problem to the system has been introduced so that it can be addressed and fixed immediately.

Also, as functionality matures, automated test cases will be developed using Selenium. Automated test cases will help add efficiency to regression testing for releases.

4.2 Test Progression/Order

The following diagram outlines the testing migration process that is followed in moving the developed system build from the development environment to the test environment.



Once the code has been moved to the Stage environment, tests will be executed in the following order:

- Smoke Testing
- System Testing
- Functional Testing
- Section 508 Compliance Testing (if needed)
- Regression Testing
- Performance Testing (if needed)
- User Acceptance Testing
- Smoke Testing (in Demo and Production only)

Prior to any deployment, there will be a full sprint of testing which will include all the items above along with full regression testing.

The following represents the overall flow of the testing process:

1) Identify the requirements to be tested. All test cases shall be derived using the Used Stories and specification documents.

2) Identify which test(s) will be used to test each module.
3) Review the test data and test cases to ensure that the unit has been thoroughly verified and that the test data and test cases are adequate to verify proper operation of the unit.
4) Identify the expected results for each test.
5) Document the test cases, test data, and expected results.
6) Perform the test(s).
7) Document the test data and test cases used during the testing process. This information will be stored in JIRA using the Zephyr plug in.
8) Unsuccessful testing requires a Bug Report Form to be generated. This document shall describe the user story, the problem encountered, it's possible cause, and the sequence of events that led to the problem. It shall be used as a basis for later technical analysis.
9) Test documents and reports shall be submitted. Any specifications to be reviewed, revised, or updated shall be handled immediately.

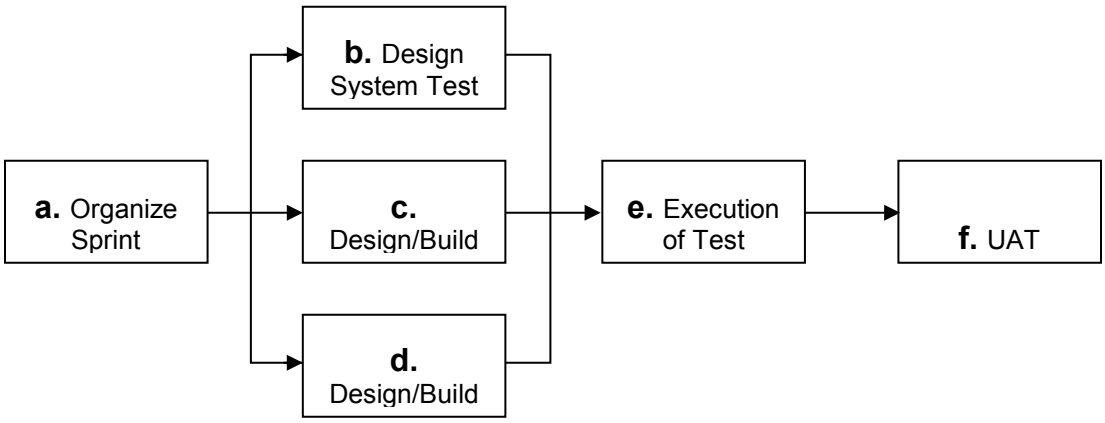


Figure 1: Test Process Flow

The diagram above outlines the Test Process approach that will be followed. A description of the steps is provided below:

a) **Organize Project** involves creating a System Test Plan, Schedule & Test Approach, and assigning responsibilities.

b) **Design/Build System Test** involves identifying Test Cycles, Test Cases, Entrance & Exit Criteria, Expected Results, etc. In general, test conditions/expected results will be identified by the Test Team in conjunction with the Development Team. The Test Team will then identify Test Cases and the Data required. The Test conditions are derived from the Program Specifications Document.

c) **Design/Build Test Procedures** includes setting up procedures such as Error Management systems and Status reporting.

d) **Build Test Environment** includes requesting/building hardware, software and data set-ups.

e) **Execute System Tests –** The tests identified in the Design/Build Test Procedures will be executed. All the 10 types of testing mentioned in Section 1.3 will be executed here. All results will be documented, and Bug Report Forms filled out and given to the Development Team as necessary.

f) **UAT** - UAT happens when all pre-defined exit criteria have been achieved.

4.3 Test Milestones

Please refer to the project schedule for testing milestones. It is in the same folder as this Master Test Plan, under the folder “Referenced Artifacts”.

4.4 Test Data

Test data will be manually generated to mirror the data entities, objects, fields, and characteristics of real data. This data is created by the QA team during functional testing and by the Operations team during the UAT period.

Within each sprint, test data has been created to support the following data entities.

4.5 Recording Results

Test results will be recorded in the appropriate column of the test case within JIRA. Each step in the test script has an expected result. The testers will record whether the expected result was achieved for each step. If the expected result is achieved, the result will be marked as “Passed”. If the expected result was not achieved, then the step will be marked as “Failed”. The testers will fully document the deviation from the expected result and log this as a defect in JIRA. If the test script can continue after the failed step, then the testing will continue. If the test script cannot continue, then this will be logged, and the priority of the defect will be escalated accordingly. The individual who opened the defect will be tracked; they will be the individual required to verify that the defect has been resolved.

Defect reports will be generated from JIRA as required.

4.6 Analyzing Results

Test reports summarizing the number, category, and priority of defects will be generated on a regular basis during the test cycle. Trend analysis will be used to assess defect injection and close rates. Effectiveness of testing will be assessed based on the number of unknown defects identified during user testing; test

scripts and sample data will be updated in order to increase the comprehensiveness of testing activities.

Testing for a sprint will be considered complete when all steps of each test script can be executed in their entirety such that all acceptance criteria and business requirements associated with the stories in the sprint have been met. All high priority defects must be resolved prior to sprint UAT; however, all defects identified do not have to be resolved for the sprint to close. (See section 6.2 for more details about the exit criteria).

5 TESTING PLAN

The following section describes how testing objectives will be met. Each of the following subsections outlines a type of testing that will be integrated into on-going testing activities at the end of each sprint, during regular “Hardening Sprints” (e.g., sprints dedicated to testing), and prior to deployment. Each test type may not be implemented within each sprint, in which case this has been noted in the description.

5.1 Unit Testing

This testing involves running test scripts against discrete modules in the system to ensure that the modules perform as expected and that changes to the system do not impact other system modules. These tests typically involve testing each of the operations of the module in all the expected combinations. They also involve testing exception cases to ensure they either pass or fail cleanly with minimal impact to the system.

Unit tests will be written by the individual developer(s) along with the development activities. All unit test cases will be compiled into a test suite and the test suite will be executed along with each build. The unit test cases will be part of the code repository. The development build and automated build will execute these test cases on each build deployment. The unit tests results will be reviewed, and a build will only proceed to the next set of test activities if all unit tests execute successfully.

5.1.1 White Box Testing

In white box testing will be done by the developer. Inputs and outputs are tested directly at the code level and the results are compared against specifications. This form of testing ignores the function of the program under test and will focus only on its code and the structure of that code. Test case designers shall generate cases that not only cause each condition to take on all possible values at least once, but that cause each such condition to be executed at least once. This will be implemented as unit tests.

5.1.2 Black Box Unit Testing

Black box unit testing is equivalent to what is called API testing or integration testing. Essentially, the test does not care how the API/integration point is implemented, it is only there to make sure that given a set of parameters, the output of the API/integration point is correct. Such test can be used on the following examples: making sure query tool is able to get permission from account modules, making sure instances are able to query centralized GUID server, and making sure tri-planar is able to get data from BRICS.

5.2 Functional Testing

During functional testing, the testing team will validate that the application, as built, performs the functions defined by the business requirements and functional specifications. The testing team will coordinate and facilitate/track the execution of functional testing activities. Functional testing focuses on individual actions in the system as opposed to the integration/combination of multiple actions. Test cases will be written in enough detail to test each individual action that the user can perform in the system. They will identify expected actions and testers will be required to make note of any deviation from the expected action as part of the defect description. This includes a description of steps taken and test data used. Functional testing also includes testing boundary conditions, ensuring that only valid data is allowed in the system.

Where possible, automation utilizing the tools outlined in section 2.2 will be leveraged to reduce the level of effort involved with functional testing. These tools can be used to automatically execute each step in the test cases and compare the results on the screens with the expected results in a manner like manual testing.

The testing team will be involved in the following activities:

- Exercise key business flows in creation of scenarios for different functionalities.
- Creation of fully automated testing scripts for regression testing.
- Creation of manual testing scripts where automation is not possible or not necessary.
- Compare the results of user actions to expected results in a variety of free-form business scenarios.

The following main functionalities will be tested as part of the Smoke Testing/Regression Testing at a minimum. Where appropriate, test cases will be written to test these functions for primary data entities and inputs in the system.

Table 6 - Summary of Key Functions to be Tested

Module	Functionality
Account Management	**As an End User:** 1. Validate system login authentication using various combination of username/password 2. Verify user can request a new account 3. Verify user can recover username and password 4. Verify user can modify existing profile via 'Edit My Profile' 5. Verify user can change password 6. Verify user can request additional privileges, including request to join an account group 7. Verify user can upload documentation to existing profile 8. Verify 'Administrative File Templates' are accessible to user **As an Account Admin:** 1. Verify admin can search an account by typing keyword in search box 2. Verify admin can filter by all statuses (All, Active, Requests, Pending, Denied, Inactive) 3. Verify Table pagination and Columns sorting are working for all filtering statuses (All, Active, Requests, Pending, Denied, Inactive) 4. Verify admin can approve an account either approving all requested privileges/with few privileges 5. Verify admin can reject any requested privileges 6. Verify admin can reject an account 7. Verify admin can deactivate an active account (Make sure deactivated account shall not be able to login to portal) 8. Verify admin can reactivate an inactive account (make sure this user can log into the portal) 9. Verify while admin reactivating an inactive user system validate username for uniqueness (if a user exists with same username then system shall force user to reactivate that user with a different username.) 10. Verify admin can create a user with all privileges 11. Verify admin create an account group by including existing users

Module	Functionality
	12. Verify admin can approve a user request to join an account group 13. Verify admin can modify user profile of a user 14. Verify admin can modify an existing account group
GUID	**As an End User:** 1. Verify user can launch GUID Client using various java version and security settings 2. Verify user can download GUID Batch Template 3. Verify user can access GUID Manual 4. Verify user can 'Generate PseudoGUID' 5. Verify user can convert PseudoGUID to Valid GUID (Check Copy PseudoGUID and PII Buttons are functioning) 6. Verify user can generate GUID with required/optional PII data (please repeat the same PII data to make sure system still generate same GUID for same PII data) 7. Verify user can validate GUID/PseudoGUID existence 8. Verify user can process batch file without error checking 9. Verify user can process batch file with error checking 10. Verify that clicking on a GUID link displayed the dataset information where that GUID been used 11. Verify user receive error in case where user try to generate a GUID against an existing GUID i.e. System shall check Hash Code 1, Hash Code 2, Hash Code 3 **As an Admin:** 1.Verify admin can access all the existing GUID in the system 2.Verify that column sorting and pagination is working for 'View All GUIDs' table
Data Dictionary	**Data Element** **As an End User:** 1. Verify user can create an element with all required and optional data (via three step process) 2. Verify user can search that data element by keyword term 3. Verify user can search that data element using 'Advanced Filter' option

Module	Functionality
	4. Verify user can edit an existing draft data element by modifying all the data fields (except define the data page) 5. Verify user can access data element import template 6. Verify user can import data element using all required and optional data; System shall allow user to import an element with multiple diseases (domain/sub domain/classification) 7. Verify user can overwrite an existing draft data element by modifying all data fields 8. Verify user can overwrite an existing published data element by modifying all fields except the following fields: Variable Name, Element Type, Data Type, Size, Input Restrictions, Permissible Values, Unit of Measure 9. Verify user can create an element using three step process inside a FS (either building a new FS or editing an existing draft FS) 10. Verify user can search an element for following statuses (draft (owned by this user), Awaiting Publication, Published) 11. Verify user can request publication for draft data element (status changed to awaiting publication, cancel to request this status will revert to draft status) 12. Verify user can download data elements details via xml, CSV for different search option (keyword/Advanced filter) 13. Verify user can delete a draft data element (owned by this user and not associated to a FS) **Admin:** 1. Verify admin can search data element using keyword/Advanced Filter Option 2. Verify admin can download data elements details via XML/CSV/XSD for any search combination 3. Verify admin can filter by following statuses (all, draft, awaiting publication, publication) 4. Verify admin can approve data element publication 5. Verify admin can delete a draft data element (As long as it is not associated to a FS) **Form Structure:** **User:**

Module	Functionality
	1. Verify user can filter Form Structure's by following statuses Ownership and Status 2. Verify user can create a FS by attaching existing DE or creating new data elements to Main or additional groups 3. Verify user can assign grant permission to individual user or account group 4. Verify user can modify draft FS by modifying all the required/non-required fields 5. Verify user can request to publish a FS (Cancel this request will revert this FS to Draft) 6. Verify user can create a draft copy of FS from a published/shared draft FS 7. Verify user can view and download Data Element details report for associated map elements 8. Verify user can export Form Structure using following link: XML, CSV, CSV with sample data 9. Verify user can delete a draft FS that user has access to that FS (owned or admin permission) **Admin:** 1. Verify admin can filter by following statuses: Ownership and Status (User shall be able to sort or paginate tables for any filtering statuses) 2. Verify admin can approve or reject a FS publication request 3. Verify approving a FS publication will also published draft elements that associated to that FS 4. Verify admin can convert a draft FS to Shared draft or vice versa (Shared Draft FS shall available to PS) 5. Verify admin can import a FS via xml tool 6. Verify admin can delete only draft FS
Data Repository	**Study:** **As an End User:** 1. Verify user can view study by using Keyword, Ownership, and Advanced Search option 2. Verify column sorting and pagination is working for any filtering combination

Module	Functionality
	3. Verify user can submit a study request by populating all the required or non-required fields 4. Verify user can edit an existing study i.e. Add/Upload documentation to study 5. Verify user can 'Manage Dataset' for an existing study i.e. 'Request to Share', 'Request to Delete', 'Request to Archive' 6. Verify user can approve permission to an existing study i.e. permission to individual user or account group **As an Admin:** 1. Verify admin can filter studies by following statuses: all, requested, public, private and rejected 2. Verify admin can approve or reject a study request (upon approval study become public) 3. Verify when admin create a study by default it become a public study 4. Verify via admin functionality admin can mark a study either public or private 5. Verify admin can delete a study by marking a study as 'Private' 6. Verify admin can access 'Repository List', Clicking on Datastore ID or Form Structure link shall display the relevant page for that Form Structure 7. Verify admin can approve or reject any dataset status change request **Validation Tool:** 1. Verify user can access VT Tool using various java version and security settings 2. Verify user can only validate a draft or shared draft FS, but cannot create submission package unless FS is published 3. Validate at least one FS (that has various element groups main, additional) using all supported data type and one element per data type to ensure that VT Tool emits error and warning for any sort of invalid data used in csv data file 4. Verify user can export result details in working directory 5. Verify when user click on 'Build Submission Package' button expected submission ticket created in user working directory **Upload Tool:**

Module	Functionality
	1. Verify user can launch Upload Tool client using various java version and security settings 2. Verify user can access all studies that user has access to make submission to data repository 3. Verify user successfully make a submission to repository by selecting study, submission ticket and providing a dataset name that is unique to a study **Download Tool:** 1. Verify user can launch Download Tool Client using various java version and security settings 2. Verify user can add data to download queue from data repository 3.Verify data added from query result page is only available via download tool UI, not in user download queue 4. Verify once user added data from query result to download queue, the 'download' link become disable 5. Verify user can delete added data from download queue either single entry or multiple entry 6. Verify user can view all the data added either from repository or query tool in download tool UI 7. Verify user can download selected data file (csv) or associated files (file, image) in a desired directory in user machine 8. Verify 'Refresh' button allow user to view latest data added to user download queue 9. Verify the following download actions are working as expected: 'Start Download(s)','Stop Download(s)','Delete Download(s)' 10. Verify download progress bar is synchronize with actual download 11. Verify downloaded data that is added from query result page for a single form that has no repeatable group, ensure that downloaded data is same as displayed in query tool and data originally submitted to repository 12. Verify downloaded data that is added from query result page for a single form that has data in main group and repeatable group, ensure that downloaded data is same as displayed in query tool and data originally submitted to repository 13. Repeat step 11 and 12 by applying either single or multiple filter including filter on instance data, ensure that for each case downloaded data is same as displayed in query tool and data originally submitted to repository

Module	Functionality
	14. Download data using each scenario described in 'Join on GUID' section for query tool, modify query using either single or multiple filter or narrow down query result by applying instance data for each selected filter, ensure that for each case downloaded data is same as displayed in query tool and data originally submitted to repository **MIPAV Tool:** 1. Verify user can launch MIPAV Tool by clicking the Launch MIPAV Tool by clicking the Launch MIPAV TOOL button or the link to MIPAV Tool. 2. Verify that the Load CSV File and Add Form Structure buttons are enabled. 3. Click on Add Form Structure tab at the bottom of Image Submission Package Creation Tool page. 4. Verify all the PUBLISHED and Imaging Form Structures are displayed in the Form Structure table. 5. Select a Form Structure and click the button Add and verify that the DEs are displayed.
Meta Study	**As an End User:** 1. Verify user can create a meta study by populating minimum required data fields 2. Verify when a new meta study saved in the system its publication status displayed as Draft 3. Verify user can request publication, edit permissions, delete, and edit all aspects of a draft meta study 4. Verify during edit workflow user can upload documentation, link PubMed, and linked a saved query that user has access 5. Verify user can search and view an existing meta study based on permission either as owner or inherited privileges 6. Verify user can edit a meta study based on permission of that meta study **As an Admin:** 1. Verify admin user can create a meta study by populating minimum required data fields 2. Verify admin user can publish any Meta Study that has draft or awaiting publication statuses 3. Verify admin user can unpublished any published Meta Study

Module	Functionality
	4. Verify when admin revert a published meta study to unpublished then status changed to draft 5. Verify admin user can edit permissions, delete, and edit all aspects of a draft meta study 6. Verify during edit workflow admin user can upload documentation, link PubMed, and linked a saved query
Query Tool	**As an End User:** 1. Verify QT shall only display those Study that user has access to that study/Form; otherwise system shall display message to user 2. Verify user can filter by studies/Forms using respective tabs from Query Tool main page 3. Verify user can execute meta search using text search or Facets for different study/form combination 4. Verify relevant 'Studies' or 'Forms' displayed to UI based on search criteria 5. Verify from 'Study' search result user either can add all forms belongs to that study or an individual form to data cart 6. Verify from 'Form' search result user either can add all studies where that Form belongs to or an individual study to data cart 7. Verify user can 'Filter Studies by Data Element(s)' or 'Filter Forms by Data Element(s)' 8. Verify user can add one or multiple data elements as filter by selecting check box 9. Verify user can search an element by 'Variable Name' or 'Title' from 'Add as Filter' window 10. Verify user can narrow the search result by applying filter for permissible values or free-form text search 11. Verify user can remove an added filter, upon removal search result shall update accordingly 12. Verify user can navigate to 'Refine Selected Data' page by accessing '1 Form in 1 Study' link from 'Your Data' section 13. Verify user can view data in data table view for a single form (i.e. total # of row counts, expand and collapse group, # of item to display per page, column sorting, pagination and data displayed under respective column as they are submitted to repository) 14. Verify user can view Form Structure details by clicking on 'Version' or 'i' tool tip

Module	Functionality
	15. Verify user can apply single or multiple filter (main or additional group) by clicking on 'Select Criteria' tab, ensure that for each filtering criteria data displayed in data table view and # of row counts are synchronized 16. Verify that user can narrow the filtering criteria by applying filter on instance data used in dataset, ensure that # of row counts and data displayed in data table are synchronized (Note: If user apply filter where that filter do not contain any data then result page shall display 0 rows of data) 17. Repeat test case # 13,14,15,16 for multiple forms with following scenarios: Form has only Main Group Form has 1 main group and 3 additional groups (exactly, At least and Up to), each group contain same repeated data elements Form has 1 main group and 3 additional groups (exactly, At least and Up to), each group contain random data elements Form has only Additional Groups, No main element group
ProFoRMS	**Site Administration:** **As a Proforms Admin:** 1. Verify user can login and navigate to Site Administration section 2. Verify user can Search and Manage user accounts 3. Verify user can Add/Edit Roles & Privileges 4. Verify user can View URLs or add/edit an URLs **As an End User:** Verify that Site Administration module is not available for non-admin user. (Includes all user roles: Principal Investigator, Associate Investigator, Research Associate, Data Manager, Data Entry, Clinical Research Associate) **Manage Study> Study Information** **Study Information:** 1. Verify user can Update A Study 2. Verify user can Create A Study 3. Verify user can modify and save Study Details

Module	Functionality
	4. Verify user can modify and save Data Repository Study Linking 5. Verify user can modify and save Study Sites 6. Verify user can Delete Study **Assign Roles:** 1. Verify user can Assign Roles to Users 2. Verify user can Assign Study Sites to Users **Create Visit Type:** 1. Verify user can Create Visit Type 2. Verify user can search and Add Standardized Published eForms to Visit Type 3. Verify user can mark eForms as Required 4. Verify user can mark eForms as Self Reporting 5. Verify user can Add before and after scheduled dates to Self Reporting eForms 6. Verify user can reorder eForm display **Manage Visit Types:** 1. Verify user can search Visit Types 2. Verify user can View Audit for Visit Types 3. Verify user can Update Visit Types 4. Verify user can Delete Visit Types **Documents/Contacts/E-Binder:** 1. Verify user can Add/edit/view/delete new or existing documents. 2. Verify user can Add/edit/view/delete new or existing contacts. 3. Verify user can Add/edit/view/delete new or existing E-Binder folders/files in a study **Test it with the following End User Roles:** **Clinical Research Associate User - (CRA)** 1. Verify user only able to Select A Study the user is assigned to 2. Verify user is NOT able to make any changes to Study 3. Verify user can View Audits for existing visit types/forms included in a Study

Module	Functionality
	4. Verify user can View existing E-Binder folders/files in a Study **Data Entry User - (DE)** 1. Verify user only able to Select A Study the user is assigned to 2. Verify user is NOT able to make any changes to Study 3. Verify user can View existing visit types/forms included in a Study **Data Manager User - (DM)** 1. Verify user only able to Select A Study the user is assigned to 2. Verify user is NOT able to make any changes to Study 3. Verify user can View Audits for existing visit types/forms included in a Study 4. Verify user can Add/edit/view/delete new or existing E-Binder folders/files in a Study **Research Associate User - (RA)** 1. Verify user only able to Select A Study the user is assigned to 2. Verify user is NOT able to make any changes to Study 3. Verify user can Create/edit/view-audit/delete new or existing visit types 4. Verify user can View existing E-Binder folders/files in a Study **Associate Investigator - (AI) / Principal Investigator - (PI) User** 1. Verify user only able to Select A Study the user is assigned to 2. Verify user can modify and save Study Information/link to Repository Study 3. Verify user can Delete Study 4. Verify user can Assign Roles to Users (excluding own user role) 5. Verify user can Create/edit/view-audit/delete new or existing visit types 6. Verify user can Add/edit/view/delete new or existing documents. 7. Verify user can Add/edit/view/delete new or existing contacts. 8. Verify user can Add/edit/view/delete new or existing E-Binder folders/files in a Study **Manage Subject:** **As a ProFoRMS Admin:**

Module	Functionality
	1. Validate system Login authentication using various combination of username/password 2. Verify user can Add Subjects with GUID or Pseudo-GUID to a study 3. Verify user can Validate GUID or Pseudo-GUID and Add Subject to a study 4. Verify user can Search/view/edit/delete subject information 5. Verify user can Add/view/edit/delete Visit to a Subject 6. Verify user can Add/view/edit/delete Attachments and Attachment Categories to a Subject 7. Verify user can Add/view/edit/delete monitor queries to a Subject **Test with the Following End User Roles:** **Clinical Research Associate User - (CRA)** 1. Verify user can Search/View Subject Demographics (includes view attachments, view audits) 2. Verify user can View/add/Edit/Delete/ Manage Monitor Queries **Data Entry User - (DE)** 1. Verify user can Search/View Subject Demographics (includes View Attachments) 2. Verify user can Schedule subject Visits in a Study (includes View/add/edit/delete visits) **Data Manager User - (DM)** 1. Verify user can Add Subjects with GUID or Pseudo-GUID to a study 2. Verify user can Validate GUID or Pseudo-GUID and Add Subject to a study 3. Verify user can Search/view/edit/delete subject information 4. Verify user can Search/View Subject Demographics (includes view attachments, view audits) 5. Verify user can Manage Attachments (includes Add/edit/delete subject attachments) 6. Verify user can Schedule subject Visits in a Study (includes View/add/edit/delete visits) **Research Associate User - (RA)** 1. Verify user can Search/View Subject Demographics (includes View Attachments and View Audit)

Module	Functionality
	2. Verify user can Schedule subject Visits in a Study (includes View/add/edit/delete visits) **Principal Investigator - (PI) User** 1. Verify user can Add Subjects with GUID or Pseudo-GUID to a study 2. Verify user can Validate GUID or Pseudo-GUID and Add Subject to a study 3. Verify user can Search/view/edit/delete subject information 4. Verify user can Search/View Subject Demographics (includes view attachments, view audits) 5. Verify user can Manage Attachments (includes Add/edit/delete subject attachments) 6. Verify user can Schedule subject Visits in a Study (includes View/add/edit/delete visits) **Collect Data** **As a ProFoRMS Admin:** 1. Verify user can Search and view Subject, by subject form or by non-subject forms ready for collection by Visit date, Study Subject ID, Form Status and Form Type 2. Verify user can Collect Data on specific GUID and/or Form 3. Verify user can Save, Save/Exit, Lock form(s) 4. Verify user can view My Collections page 5. Verify user can View Entry and View Audit of collected forms 6. Verify user can Edit the data for collected form 7. Verify user can Reassign the data for collected form *(note: Reassigning Locked form is not allowed)* 8. Verify user is able to Delete the data for collected form 9. Verify user is able to Export the data for collected form(s) *(note: Exporting collections for different forms is not allowed)* 10. Verify user is able to Resolve Discrepancies for double data entry forms **Clinical Research Associate User - (CRA)** 1. Verify user can Search and view Subject, by subject form or by non-subject forms ready for collection by Visit date, Study Subject ID, Form Status and Form Type 2. Verify user can view My Collections page 3. Verify user can View Entry and View Audit of collected forms

Module	Functionality
	Data Entry User - (DE) 1. Verify user can Search and view Subject, by subject form or by non-subject forms ready for collection by Visit date, Study Subject ID, Form Status and Form Type 2. Verify user can Collect Data on specific GUID and/or Form 3. Verify user can Save, Save/Exit, Lock form(s) 4. Verify user can view My Collections page 5. Verify user can View Entry and View Audit of collected forms 6. Verify user can Edit the in-progress data for collection if the user is an owner or assigned user **Data Manager User - (DM)** 1. Verify user can Search and view Subject, by subject form or by non-subject forms ready for collection by Visit date, Study Subject ID, Form Status and Form Type 2. Verify user can Collect Data on specific GUID and/or Form 3. Verify user can view My Collections page 4. Verify user can View Entry and View Audit of collected forms 5. Verify user can Resolve Discrepancies for double data entry forms **Research Associate User - (RA)** 1. Verify user can Search and view Subject, by subject form or by non-subject forms ready for collection by Visit date, Study Subject ID, Form Status and Form Type 2. Verify user can Collect Data on specific GUID and/or Form 3. Verify user can view My Collections page 4. Verify user can View Entry ONLY of collected forms 5. Verify user can Resolve Discrepancies for double data entry forms **Principal Investigator User - (PI)** 1. Verify user can Search and view Subject, by subject form or by non-subject forms ready for collection by Visit date, Study Subject ID, Form Status and Form Type 2. Verify user can Collect Data on specific GUID and/or Form 3. Verify user can Save, Save/Exit, Lock form(s) 4. Verify user can view My Collections page

Module	Functionality
	5. Verify user is able to View Entry and View Audit of collected forms 6. Verify user can Edit the data for collected form 7. Verify user can Reassign the data for collected form *(note: Reassigning Locked form is not allowed)* 8. Verify user can Delete the data for collected form 9. Verify user can Resolve Discrepancies for double data entry forms **Reports:** **ProFoRMS Admin and End Users:** 1. 1. Verify user can view Study Reports and Completed Visit reports for entire system 2. 2. Verify user can view ALL Reports for selected Study

In addition, we will test:

- The interfaces to ensure they are functioning as desired (i.e. check if each interface is behaving as expected, specifically verifying the appropriate action is associated with each mouse click event).
- The interaction between the GUI and the backend repository. In this case the data will be inserted and check if they are processed in the backend and give the expected output.

5.2.1 Positive Testing

Positive Testing is carried with an idea of checking whether the application works as per requirements or not. In other words, making sure of whether the system does what it should really is intended to do.

5.2.2 Negative Testing

Negative Testing is trying to make sure that the application does what it should not and does not what it should do.

5.3 System Testing

System testing involves running end-to-end scenarios and business use cases against the entire application, ensuring that the application as a whole performs as expected with regards to functionality. As opposed to functional testing, this method does assess how individual functions work together to support broader business requirements. The test scripts are designed by the tester based on the business logic and scenarios defined by the business requirements and functional specifications. They include both normal and exception cases. These

scenarios will then be executed by a group of testers to constitute the end-to-end testing for the project. All code, content, and data must be present for end-to-end testing to take place.

In addition to business logic and scenarios as defined by the business requirements, focus areas of system testing include:

- Verifying that the user interface (e.g., look and feel of links, buttons, colors, etc.) of the system is consistent across screens
- Verifying that the system performs as expected across all supported operating systems, environments, and internet browsers
- Verifying the data flow from the ProFoRMS module to the Data Repository module happens when MIRTH is run
- Verifying that the data flow from Data Repository to the Query Tool happens when the nightly process RDFGen is run

5.4 Section 508 Compliance Testing

The Axe tool from Deque (https://www.deque.com/axe/) will be used to test that 508 Compliance for Web Based Internet Information and Applications (1194.22) are being met

While not utilized at the end of every sprint, these tools will be utilized whenever significant changes are made on our website. Where possible, accessibility tools will be used to assess whether compliance is met.

The following standards are excerpted from Section 508 of the Rehabilitation Act, §1194.22. The pass/fail criteria in this document represent an interpretation of Section 508 web standards.

Table 7 - Summary of Section 508 Standards

508 STANDARD	PASS	FAIL
(a) A text equivalent for every non-text element shall be provided (e.g., via "alt", "longdesc", or in element content).	Every image, applet, embedded media, plug-in, etc. that conveys content has equivalent alternative text (alt, longdesc, or in the element context).	A non-text element has no alt or text description or the description is not equivalent, or is not described in the adjacent text.
	The alternative text succinctly describes the content conveyed by the	Alternative texts are verbose ("picture of...", "image of...", etc.), vague,

508 STANDARD	PASS	FAIL
	element, without being too verbose (for simple objects) or too vague (for complex objects).	misleading, inaccurate, or redundant to the context (e.g. the alt text is the same as adjacent text).
	Complex graphics (graphs, charts, etc.) are accompanied by equivalent text, either through a description in the body of the page, a link to a description on a separate page, and/or the longdesc attribute. [See Note 1]	Complex graphics have no alternative text or the alternative does not fully convey the content of the graphic.
	Images that have a function (images within links, image buttons, and image map areas) have alternative text which describes the associated function.	Alternative texts for linked images, image buttons, or hot spots are not descriptive of the function.
	Decorative graphics are CSS background images or have null/empty alt values (alt=""). Images with text alternatives in element content are given empty alt text to avoid redundancy.	Decorative graphics have alternatives of "spacer", "decorative graphic," or other extraneous text. Graphics have alt text that is redundant with adjacent text.
	Transcripts are provided for audio content.	Audio does not have transcripts.
508 STANDARD	**PASS**	**FAIL**

508 STANDARD	PASS	FAIL
b) Web pages shall be designed so that all information conveyed with color is also available without color, for example from context or markup.	Color is not used solely to convey important information.	Color is the sole means of conveying information.
	Enough contrast is provided.	Contrast is poor.

508 STANDARD	PASS	FAIL
(c) Row and column headers shall be identified for data tables.	Data tables have column and/or row headers appropriately identified (using the <th> element).	Data tables have no header rows or columns.
	Tables used strictly for *layout purposes* do NOT use the <th> element.	Tables used for layout have headers identified when there are no true headers.

508 STANDARD	PASS	FAIL
(d) Markup shall be used to associate data cells and header cells for data tables that have two or more logical levels of row or column headers.	Data table cells are associated with the appropriate headers using the scope or id/headers attributes.	Data table cells are not associated with column and/or row headers or they are associated incorrectly.

508 STANDARD	PASS	FAIL
(e) Frames shall be titled with text that facilitates frame identification and navigation.	Each frame is given a title that describes the frame's purpose or content.	Frames have no title or a title that is not descriptive of the frame's purpose or content.

508 STANDARD	PASS	FAIL
(f) When a web page requires that an applet, plug-in or other application be present on the client system to interpret page content, the page must provide a link to a plug-in or applet that complies with §1194.21(a) through (l).	A link is provided to a page where the plug-in can be downloaded.	No link is provided to a page where the plug-in can be downloaded.
	All applets, scripts and plug-ins (including PDF and PowerPoint files, etc.) and the content within them are accessible to assistive technologies, or else an alternative means of accessing equivalent content is provided.	Inaccessible plug-ins, scripts, and other applications are used without providing an accessible alternative.

508 STANDARD	PASS	FAIL
(g) When electronic forms are designed to be completed on-line, the form shall allow people using assistive technology to access the information, field elements, and functionality required for completion and submission of the form, including all directions and cues.	<input>, <textarea>, and <select> elements have descriptive labels.	There is no association between the form element and its label.
	Scripting of form elements does not interfere with assistive technologies or keyboard.	Scripting makes parts of the form unavailable to assistive technologies or keyboard users.

508 STANDARD	PASS	FAIL
(h) A method shall be provided that permits users to skip repetitive navigation links.	A link is provided to skip over navigational menus or other lengthy lists of links. A good heading	There is no way to skip over repetitive lists of links.

508 STANDARD	PASS	FAIL
	structure also facilitates navigation.	

508 STANDARD
(a) When software is designed to run on a system that has a keyboard, product functions shall be executable from a keyboard where the function itself or the result of performing a function can be discerned textually.
(b) Applications shall not disrupt or disable activated features of other products that are identified as accessibility features, where those features are developed and documented according to industry standards. Applications also shall not disrupt or disable activated features of any operating system that are identified as accessibility features where the application programming interface for those accessibility features has been documented by the manufacturer of the operating system and is available to the product developer.
(c) A well-defined on-screen indication of the current focus shall be provided that moves among interactive interface elements as the input focus changes. The focus shall be programmatically exposed so that assistive technology can track focus and focus changes.
(d) Textual information shall be provided through operating system functions for displaying text. The minimum information that shall be made available is text content, text input caret location, and text attributes.
(e) Color coding shall not be used as the only means of conveying information, indicating an action, prompting a response, or distinguishing a visual element.
(f) When electronic forms are used, the form shall allow people using assistive technology to access the information, field elements, and functionality required for completion and submission of the form, including all directions and cues.

5.5 Regression Testing

As part of the regression testing approach, there will be a set of test cases from the previous sprint which will be used in the regression testing bed (the first sprint will not have a set of test cases in this testing bed). Once the testing and fixes

are complete for the current sprint, regression testing is performed to confirm that the application has not regressed to a less functional state that in the previous build.

Selected test scripts from each type of testing (e.g., unit, functionality, system, etc.) are included in the test plan for regression testing. The test plan also includes the expected results for all test scripts in the regression testing bed. After the regression testing is complete, the results are compared with the initial test results. The comparison of these test results is used to prepare the final regression test report.

Manual testing that has been automated in Selenium will be executed in full with each test cycle.

5.6 Smoke Testing

Prior to a system deployment to production, a round of smoke testing is performed in pre-production environment to ensure that major functional areas are operational, and that navigation is working as designed (e.g., tab, button and pages). This is a basic activity which involves navigating through the site, checking links, and generally ensuring that all aspects of the system that can be tested without saving new data elements are assessed. If any major issues are identified, the development team is notified to make changes prior to the testing team re-engaging to move forward with the remaining test activities.

After each deployment into production, smoke testing is performed to ensure the system is working as designed without impacting the integrity of the data in the production environment. The activities are like those described above.

5.7 Security Testing

This involves testing the security of both the application and underlying infrastructure (firewalls, web-server, database, etc.), attempting both legal and illegal operations to ensure that the four aspects of security are being handled correctly:

- **Privacy/Encryption** – Is the data readable by an outside party?
- **Authentication** – The process of identifying, and individual, usually based on user name and Password?
- **Authorization** – The process of granting or denying access based on the user identity?
- **System** – Are the underlying network, hardware, and software infrastructures safe from penetration?

Assessment of these areas will include testing done by the development team under the guidance of the security critical partners as well as security scans

performed by CIT on the production environments prior to deployment(s). This section will be updated as more information is provided regarding the security testing required to support an authorization to operate and eventual deployment.

5.8 Automation Testing

The Automation Testing involves the following activities to be carried out

1) Test Scenario derivation – Considering Regression scope from stories
2) Test Script development
3) Dry Run Execution/Script updates (need to perform after code freeze)
4) Test Script Execution with every new build

5.9 Performance Testing

Performance testing will be focused on measuring the average page load times for the various pages within the application. Random sets of test cases will be executed to test the performance of the system. Typically, this level of testing will be performed using a timer.

This testing is performed only when a new capability has been introduced which would enhance the performance of the existing module (for example Query Tool).

5.9.1 Load Testing

Load testing will be performed to assess the system's performance under varying loads. Evaluation areas include, but will not be limited to:

- Average page load times as the number of concurrent users increases from the minimum to the average number of users
- Average time to complete business operations performed on data sets of varying sizes
    - Targeted business operations will be determined with the business owners

5.9.2 Stress Testing

Spike testing will be performed to assess the system's upper limits. Evaluation areas include, but will not be limited to:

- Average page load times when the maximum number of users are logged in
- Average time to complete operations (e.g., search and reporting) on large data sets
    - The large data sets on which users will frequently perform

operations will be determined with the business owners

5.9.3 Spike Testing

Load testing will be performed to assess the systems performance as a result of sudden increases in load. Evaluation areas include, but will not be limited to:

- Average time it takes for pages to load when the number of concurrent users suddenly increases
- Average time it takes to complete operations when data intensive operations occur simultaneously (e.g. large number of queries being run at once)

5.10 User Acceptance Testing

User Acceptance testing involves demonstrating to the operations team members that the application performs in accordance with the statement of work, and that any outstanding defects or issues are acceptable to the client and/or the agreed level of compliance outlined in the statement of work / contract. Typically, this involves bringing in Ops team representing different instances of the DCB BRICS team. These users execute formal test scripts and ad-hoc testing to uncover unexpected issues that were not apparent to the DCB QA/ development team. The results of this test event will be captured, categorized, and prioritized for review with the business owners. High priority defects will be incorporated into the scope for the following sprint and re-tested in the subsequent user acceptance testing event.

6 TEST EXECUTION

6.1 Test Identification

Please refer to the JIRA link to review all the Test Cases created/executed for the BRICS release: https://dcb-jira.cit.nih.gov/issues/?filter=13800

6.2 Entry and Exit Criteria

6.2.1 Entry Criteria

The following criteria must be met before moving into the test phase for any sprint:

- The code delivered by developers will produce functionality that meets the acceptance criteria outlined in the stories

- The requirements for the upcoming sprint, as captured in the use cases, have been reviewed and approved by the business owners
- Manual test scripts are documented and available in JIRA
- Automated test scripts, as needed, are documented and available
- UAT test scenarios, as needed, are documented
- Unit tests have been written and integrated into the continuous build process
- Regression test scripts have been selected
- The STAGE environment is in stable condition
- Developers have conducted unit testing and have deployed the applications to the test environment including database changes
- Test data has been created that represents expected, boundary, and unexpected data scenarios
- Performance objective and threshold values have been identified for any performance test activities
- The Graphical User Interface and the back-end must be fully functional.
- All Black Box testing must be complete and exposed bugs must be corrected.
- All White Box testing must be complete and exposed bugs must be corrected.
- Function Validation Testing is the accepted method of testing for all the BRICS module.

6.2.2 Exit Criteria

The following test criteria must be met before exiting the test phase for any sprint:

- All test scripts have been executed and defects recorded with enough details to support fixing and re-testing
- All Severity 1 and Severity 2 defects have been fixed, re-tested, and closed
    - Business owner approval will be provided before any Severity 1 or Severity 2 defects can be deferred to the next release

- UAT has concluded and the business owner(s) have provided approval to move forward to the production environment.

Prior to deployment to the production environment, the goal is to have 95% of defects resolved. 100% of severity 1 and agreed upon Severity 2 defects will be resolved.

- All the deliverables are completed and signed-off by the stakeholders. Master Test Plan, Test Cases, Test and Defect Report.
- Any application product like Test Cases or Defects marked as NA (Not Applicable) should be reviewed and approved by the stakeholders
- All the defects which are found during the testing phase should be closed or deferred by taking approval from all the stakeholders.

7 RECORDS MANAGEMENT

All data and/or records generated during this procedure are stored in the NINDS SharePoint-based Document Library.

8 REVIEW/REVISION HISTORY

Date	Author	Description of Change
03/08/2019	Gladys Wang	Added Appendix A, B, C and reformatted
04/10/2019	Leonie Misquitta	Revised document title
04/22/2019	Tsega Gabremichael	Added Table 6 & 7

APPENDIX A: RTM

The Requirements Traceability Matrix (RTM) in the table below was prepared before a software release.

#	Key	Summary	Linked Test Cases/Comments		
1	PS-4420	Remove/hide the 'Biorepository Subject ID' field and user inputs in ProFoRMS	PS-4477		
2	PS-4378	"See also" - need to increase the size of the field in the database and the text box that displayed in DD	PS-4389		
3	PS-4331	Add to GUID tool Country of Birth	PS-4487	PS-4488	PS-4489
4	PS-4268	As user, system should let me refresh my session in Data Dictionary	PS-4336		
5	PS-4267	As an Operation User, data error report email should be get updated with no of request for archived, deleted data set and requested study.	PS-4327		
6	PS-4266	As a user, system should allow me to change subject label GUID to Subject ID or vice versa in Manage Protocol Section of ProForms	PS-4335		
7	PS-4179	SVN to GIT Migration	Not a Functional Test Item hence NO Test Cases		
8	PS-4135	As a admin user, I need my previous admin noted migrated into the new admin note functionality box.	PS-4308		
9	PS-4031	As cdRNS user, I should able to see cdNRS specific Form Structures	PS-4479		
10	PS-4028	As FITBIR Public site data dictionary - default DE view to Awaiting Publication and Published DEs	PS-4307		
11	PS-4027	As a user, I should know which environment my email came from	PS-4338		
12	PS-4009	As a user, I should know whether my email is for BioSample or BioFind	Duplicate of PS-4027 so was closed		

13	PS-3830	Data Repository - Research Management Table Sort Recommendation	PS-4326	PS-4325	
14	PS-3829	Need to update the primary BRICS public site	Duplicate of CRIT-9102 so was closed		
15	PS-3828	As Account Admin, I should able to approve or rejected privileges individually.	PS-4013		
16	PS-3827	As a Account Admin/Reviewer, I should able to identify user, whose privileges been expired.	PS-4328		
17	PS-3821	As a user, I should see consistent in data set statuses across table	PS-4340	PS-4341	
18	PS-3659	Supporting Documentation | The "File" field information appears under the Column Name.	PS-4314	PS-4315	
19	PS-3441	As a user, I want files I'm attempting to upload to be maintained when there is a validation error in related fields when trying to upload the file (PF, Act, DD))	PS-4334		
20	PS-3309	As a user, I should have the Ability to make an optional/recommended question required on the eform level	PS-4343		
21	PS-3226	As an admin or end user, I want the data dictionary search to support special characters *,? and "" in order to increase the likelihood of returning a result the user is searching for.	PS-4483	PS-4484	
22	PS-3214	Implementation of converting byte array to hex is flawed	Not a Functional Test Item hence NO Test Cases		
23	CRIT-10029	Update GUID Manual file	CRIT-10043		
24	CRIT-9903	Data Validation Calculation Rules - an error message should occur if a total score or a subscore is given while one of the needed data inputs is missing	Spreadsheets with datasets attached to the User Story		
25	CRIT-9874	Data validation calculation rules - DHI	Spreadsheets with datasets attached to the User Story		

26	CRIT-9844	Data validation calculation rules - Headache Impact Tool (HIT-6)	Spreadsheets with datasets attached to the User Story		
27	CRIT-9774	Data Validation Calculation Rules - FIM Instrument	Spreadsheets with datasets attached to the User Story		
28	CRIT-9581	As a developer, I should be able to migrate GUID information from one MongoDB instance to another	Not a Functional Test Item hence NO Test Cases		
29	CRIT-9549	As a user, I should be able to generate GUIDs in a batch process	CRIT-9729	CRIT-9730	
30	CRIT-9535	As a user, my PII should be sent to the server as a Hashcode	CRIT-10048		
31	CRIT-9534	As a user, I should be able to interface with the GUID server	CRIT-9767		
32	CRIT-9533	As a user, I should be able to see the interface of the GUID client	CRIT-9721		
33	CRIT-9532	As a user, I should not be able to enter invalid PII	CRIT-8983	CRIT-9787	CRIT-9788
34	CRIT-9531	As a user, I should be able to validate that a GUID exists	CRIT-9785	CRIT-9786	
35	CRIT-9530	As a user, I should be able to convert a pseudoGUID to a GUID	CRIT-9770	CRIT-9771	
36	CRIT-9520	For large result sets in QT, adding to download queue crashes the server	CRIT-9796		
37	CRIT-9511	3.6 Code Merge for NINDS/NIA/GRDR instances + add mongoDB conn info	Not a Functional Test Item hence NO Test Cases		
38	CRIT-9356	NINDS/NIA/GRDR Code Merge	Not a Functional Test Item hence NO Test Cases		
39	CRIT-9259	Data validation-calculation rules - SCAT-3	Spreadsheets with datasets attached to the User Story		
40	CRIT-9223	GUID Javascript Client Design	Not a Functional Test Item hence NO Test Cases		
41	CRIT-8571	Design Basic GUID JS Framework	Not a Functional Test item hence NO test cases		

42	CRIT-7779	As a user, I should be able to access the GUID user guide	CRIT-8990		
43	CRIT-7655	As a user, I should be able to generate a GUID through the new JS Client	CRIT-9768	CRIT-9769	
44	CRIT-7640	As an Admin, I should see all legacy & new GUIDs that have been created by my Entity	CRIT-8993	CRIT-8995	
45	CRIT-7613	As a NINDS user, I should be able to access legacy NINDS GUIDs	CRIT-8993		
46	CRIT-7612	As an NIA user, I should be able to access legacy NIA GUIDs	CRIT-8993		

APPENDIX B: TEST CASES

The details for the linked Test Cases for a software release are summarized in the table below.

#	Key	Summary	Status
1	PS-4489	GUID: Verify Batch File uploads creates GUIDs against newly added Countries as Place of Birth.	Pass
2	PS-4488	GUID: Verify new Place of Birth are added to Create New GUID in COB & GIID Countries list.	Pass
3	PS-4487	GUID: Verify new Place of Birth are added for Convert Pseudo GUID to GUID COB & GIID Countries list.	Pass
4	PS-4484	TC: Dictionary: Verify the user can perform Searches using special characters "" , ? , and * .	Pass
5	PS-4483	TC: Dictionary "Advanced Search Capability" light box sumarizes the Advance search capabilities.	Pass
6	PS-4479	TC: Public Site user are not displayed Form Structure with "Standard NINDS CDE" standardization and "Traumatic Brain Injury" disease by default.	Pass
7	PS-4477	TC: Remove/hide the 'Biorepository Subject ID' field and user inputs in ProFoRMS	Pass
8	PS-4392	Public Site: Middle Initial is added to PI Name on Study Profile to add "F Name, M Initial, L Name in PUBMed Advance Search	Pass

9	PS-4389	TC: Dictionary - Data Element "See Also" field allows the 1000 Alpha Numeric Characters	Pass
10	PS-4388	TC: Dictionary - Any user can perform Boolean operator AND, OR, NOT on Form Structure.	Pass
11	PS-4374	TC: ProFORMs: Field label "Visity Type" is updated to "Type"	Pass
12	PS-4363	TC: Accounts Management: Admin user can generate and download DAR Reports.	Pass
13	PS-4362	TC: Accounts Management: Admin user can generate and download DSR Reports.	Pass
14	PS-4359	TC: Form Structure - System should gracefully handle the form structure publication related issue	Pass
15	PS-4343	TC: As a user, I should have the Ability to make an optional/recommended question required on the eform level	Pass
16	PS-4341	TC: Repository Study: Verify the Manage Datasets page "Status" definition text has been updated.	Pass
17	PS-4340	TC: Repository Study: Datasets with status "Private - Requested Archive" are displayed in both Manage Studies and Manage Datasets tables	Pass
18	PS-4338	TC: System Generated emails displays the environment extension after the email subject.	Pass
19	PS-4336	TC:As user, system should let me refresh my session in Data Dictionary	Pass
20	PS-4335	TC: System shall allow PDBP Clinical Coordinator to should allow me to change subject label GUID to Subject ID or vice versa in Manage Protocol Section of ProForms	Pass
21	PS-4334	TC:As a user, I want files I'm attempting to upload to be maintained when there is a validation error in related fields when trying to upload the file (PF,MS,DR, DD))	Pass
22	PS-4328	TC: Account Management: users with expired privileges are displayed on "Account Renewal Dashboard" table filtered as "Active - Expired"	Pass
23	PS-4327	TC: Data Repository: Operation User, data error report email should be get updated with no of request for archived, deleted data set and requested study.	Pass
24	PS-4326	TC: Data Repository: Primary Principal Investigator is displayed on top followed by Principal Investigator by default on Study Research Management Table.	Pass
25	PS-4325	Meta Study: Primary Principal Investigator is displayed on top followed by Principal Investigator by default on Research Management Table.	Pass

26	PS-4315	TC: Meta Study: Verify attributes Title and Documentation has been added to Add Documentation table and required Field "Title" added to Add File/URL lightbox.	Pass
27	PS-4314	TC: Repository Study: Verify attributes Title and Documentation has been added to Add Documentation table and required Field "Title" added to Add File/URL lightbox.	Pass
28	PS-4308	TC: Account Management: Administrative Notes text box has been removed and notes have migrated to "Account Administrative Notes" table	Pass
29	PS-4307	TC: Public Site: Verify the Status navigation only displayed checked DEs "Awaiting Publication" and "Published" by default.	Pass
30	CRIT-10048	Test Case - As a user, my PII should be sent to the server as a Hash code	Pass
31	CRIT-10043	TC: GUID- Verify the GUID Updated GUID Java Script Manual is uploaded on GUID JS page.	Pass
32	CRIT-9802	TC: Public Site: Verify ORC ID is moved below the PI Name on Study Profile page.	Pass
33	CRIT-9796	TC: System shall allow user to download any data displayed in Query Tool data table regardless the size of the package	Pass
34	CRIT-9787	TC: GUID Interface: Verify GUID Matching Alert for Hash code 2 from front end	Pass
35	CRIT-9786	TC: GUID Interface: Search if New and Legacy GUID Exists in the System	Pass
36	CRIT-9785	TC: GUID Interface: Search if New and Legacy Pseudo GUID Exists in the System	Pass
37	CRIT-9770	TC: GUID Javascript: Convert Pseudo GUID to GUID using existing PII information of legacy GUID from the same instance	Pass
38	CRIT-9769	TC: GUID Javascript: Create new NIH Prefix GUID using the Required and Optional Fields	Pass
39	CRIT-9768	TC: GUID Javascript: Create new NIH Prefix GUID using only the Required Fields	Pass
40	CRIT-9767	GUID Javascript: Verify the GUID Java Script interface is created on "Create GUIDs" page.	Pass
41	CRIT-9753	CLONE - Test Environment Specific - Clicking on MIPAV User Guide link throws yellow triangle	Pass
42	CRIT-9730	TC: Web GUID Interface: Create Multiple GUIDs (Without Error Checking)	Pass

43	CRIT-9729	TC: Web GUID Interface: Create Multiple GUIDs (With Error Checking)	Pass
44	CRIT-9721	TC: GUID Interface: GUID web interface is added on "Create GUIDs" page.	Pass
45	CRIT-8983	GUID Client: Verify GUID Matching Alert for Hash code 2 from front end	Pass

APPENDIX C: PERFORMANCE STATISTICS

The statistics for loading a single form has been collected in the table below.

form_1	1_Stage_time	4ppl_Stage_SameT	1_PROD time	Stage_Prod_Variance (sec)
Single Form Load				
BiosampleCatalogV5	9s	20s	6s	3s
FamilyHistory	5s	8s	3.5s	1.5s
Demographics	3s	4s	3s	0s
LabTestTracking	4s	4s	3s	1s
MDS_UPDRS	9s	17s	8s	1s
NeurologicalExam	7.5s	12s	8s	0.5s
MoCA	3.5s	4s	3s	0.5s
VitalSigns	3s	4s	3s	0s
PriorAndConcomitantMeds	3s	3.5s	3s	0s

The statistics for 3 form joints were displayed in the table below.

form_1	form_2	form_3	1_Stage_time	4ppl_Stage_SameTime	1_PROD time	Stage_Prod_Variance (sec)
BiosampleCatalogV5	LabTestTracking	MDS_UPDRS	2.07m	4.91 m	58 s	1m 9s
BiosampleCatalogV5	MDS_UPDRS	NeurologicalExam	3.01m	6.5m	1.57m	19s

APPENDIX D: KEY TERMS

The following table provides definitions and explanations for terms and acronyms relevant to the content presented within this document.

Term	Definition
CCB	Change Control Board
CCR	Configuration Change Request
CM	Change Management
CR	Change Request
O&M	Operations and Maintenance
PMP	Project Management Plan
PPA	Project Process Agreement
QA	Quality Assurance
RTM	Requirements Traceability Matrix
SOW	Statement of Work
SRDS	System Requirements Design Specification
TL	Track Lead