**APPROVALS :**

Author

Project Manager

Regulatory Manager

System Engineering manager

**Introduction:**

This document defines the test plan for the functionality of the FHIR server The purpose of this plan is to define the detailed testing process for each of the interactions in the FHIR server.

**Objectives:**

* To verify that the FHIR interactions are implemented correctly according to the specification.
* To ensure that the server is secure for usage under derived standard
* To ensure that the server interactions are reliable and perform as expected.
* To ensure that the interactions are secure
* To ensure that the interactions are compatible with other FHIR systems.

**Scope:**

**Features to be tested:**

* Instance level interactions such as read, vread, update, patch, delete, and history
* Type level interactions such as create, search, delete, and history
* Whole system interactions such as capabilities, batch/transaction, delete, history, and search
* Security features and Interoperability with other systems and applications
* Performance and scalability of the system under different load conditions
* Usability and accessibility of the user interface
* Compatibility across systems of different organizations
* Error handling and recovery mechanisms
* Compatibility with different browsers and devices

**Features not to be tested:**

* Non-functional requirements that are outside the scope of testing, such as regulatory compliance or business requirements
* Third party tools which may be used in terms to testing if any
* Features that are under development and not released
* Third-party applications or services that are not directly related to the system under test

**Test Approach:**

The testing will be performed using a combinatorial method with a mix of manual and automated techniques at multiple test levels. The testing will be conducted in the following phases:

* Unit testing: Each feature will be tested individually by the developers or the testers wherever applicable
* Integration testing: Features will be tested together to ensure they work as expected.
* System testing: The entire system will be tested as a whole to ensure it meets the requirements.
* Continuous Deployment and Integration Test: Automated testing using Jenkins and Postman tool to ensure continuous testing of the API. Streamlined deployment process using GitHub to ensure efficient and error-free deployment of the API.

**Test Environment:**

* FHIR server and client software that implement the FHIR HTTP interactions.
* Test accounts with appropriate permissions to access the FHIR server and client software with proper authentications if any
* Test data that represent various scenarios for the FHIR HTTP interactions.

**Test Pass/Fail Criteria:**

* Pass criteria: A test item will be considered as "passed" if it meets all of the specified requirements, produces the expected results, and does not result in any defects being identified. Additionally, the test item must be completed within the specified timeframe and with the required level of quality.
* Fail criteria: A test item will be considered as "failed" if it does not meet one or more of the specified requirements, produces unexpected results, or results in defects being identified. Additionally, the test item may be considered as failed if it cannot be completed within the specified timeframe or if it does not meet the required level of quality..

**Test Deliverables:**

The following deliverables will be produced:

* Test cases
* Test data
* Test scripts
* Test results
* Test summary report
* Defect report
* Traceability matrix
* Test environment configuration
* User documentation
* Release notes.

**Test Schedule:**

<Depends on the kind of project and the life cycle either Waterfall, Agile or V model >

**Risks and mitigations:**

|  |  |
| --- | --- |
| Software bugs in FHIR interactions could cause unexpected behaviour or errors. | Conduct thorough unit testing, integration testing, and system testing to identify and fix any software bugs. Defect management system to be used for management of defects |
| Team experience and knowledge on the product could be less | Provide training and knowledge transfers to the resources and maintain a document with the concepts and basics of testing API and FHIR standards |
| Hardware failures such as server crashes, network outages, or disk failures could impact the availability and performance of the FHIR interactions | Conduct regular backups of all FHIR data and implement redundancy and failover mechanisms to ensure that FHIR interactions remain available and operational in the event of a hardware failure |
| Insufficient test data may not provide enough coverage of all possible scenarios. | Collect a wide range of test data to ensure that FHIR interactions are tested under various scenarios and conditions. Test data should include realistic values and be representative of the production environment |
| Inadequate testing may not identify all potential issues or defects in the FHIR interactions | Conduct thorough testing of all FHIR interactions, including unit testing, integration testing, system testing, and acceptance testing. Consider involving end-users in the testing process to ensure that all functional requirements are met |
| Budget timeless and schedule changes with frequent requirement updated | Frequent collaboration with the system engineer or the technical architect and be aware of the upcoming changes so that the same can be accommodated earlier |
| External factors such as network latency or security breaches could impact the performance and security of the FHIR interactions | Implement security protocols to protect FHIR data and ensure that the network infrastructure is robust and reliable. Conduct regular security audits and testing to identify and address any potential security vulnerabilities |

**Test Resources:**

The following resources will be required for testing:

* RESTful API client(POSTMAN) and server under test
* Testing tools/Requirements/Test management tools/Defect management tools
* Documentation
* Requirements/User cases/User stories

**Test Cases and Test Data:**

<Refer the sheet Test cases for test case details >

<The automated tests are available in the path <Test suite path>>

**Entry and Exit Criteria:**

**Entry Criteria**:

* All required test data and test environments are available and properly configured.
* All software and hardware dependencies are available and properly configured.
* All necessary documentation, including test plans, test cases, and user manuals, are complete and available.
* All relevant stakeholders, including developers, testers, and end-users, are available and have been informed about the testing process and objectives.

**Exit Criteria**:

* All test cases have been executed and passed, and any defects identified during testing have been addressed.
* All functional and non-functional requirements have been met.
* All test results, including metrics and summary reports, have been reviewed and approved by relevant stakeholders.
* The FHIR interactions have been validated and are ready for deployment.

**Suspension and Resumption Criteria:**

**Suspension criteria:**

* Critical defect or issue is discovered that prevents testing from continuing.
* Test environment is unavailable or becomes unstable.
* Critical update in Update in requirements which is critical to the testing
* Key team members or resources become unavailable or unavailable for an extended period of time.
* Test data is unavailable or inaccurate, making it impossible to execute tests effectively.

**Resumption criteria:**

* The defect or issue preventing testing has been resolved or a workaround has been identified.
* Server has been restored or made stable again.
* Updates are addressed and the test plan has been updated accordingly.
* The key team members or resources have become available again.
* Accurate and appropriate test data is available for testing

**Defect Management Procedure:**

The tests are executed formally and the defects if any are being added to the defect management systems tool used. The defect resolution board ensures that the defects are being prioritized for fix by the development team. The defect follows the defect life cycle and the details of each phase Is documented in the tool as per the template followed by the organization

**Test Dependencies:**

Test dependencies refer to the resources and artifacts required to execute the test plan for FHIR interactions. Here are some examples of test dependencies:

* Test data: This includes data sets that are used to execute the test cases, such as sample FHIR resources or mock patient data.
* Test environment: This refers to the infrastructure and tools required to execute the test plan, including hardware, software, network connectivity, and databases.
* Test tools: These are tools used to manage, automate, or execute tests, such as testing frameworks, test automation tools, or performance testing tools.
* Test documentation: This includes the test plan, test cases, and other documentation required to plan, execute, and report on the testing process.
* Test personnel: This refers to the people involved in executing the test plan, including testers, developers, and other stakeholders.
* Test schedules: This includes the timelines, milestones, and schedules required to plan and execute the testing process.
* Test budgets: This refers to the financial resources required to execute the testing process, including costs associated with hardware, software, and personnel.

**Test Approvals:**

The following approvals will be required:

* Verification Test plan approval
* Verification Test case approval
* Verification Test results approval
* Test report approval
* Defects signoff report

**Test Execution:**

* Set up the test environment
* Execute each test case for each interaction
* Record the results in the test report template
* Analyze the test results and report any issues or defects found
* Fix any issues or defects found
* Re-run the failed test cases
* Report the final test results

**Test Reporting:**

* Test case ID and description
* Test environment used
* Test data used
* Expected results
* Actual results
* Pass/fail status
* Required evidences and screeshots
* Notes and comments
* Any defects or issues found
* Test execution date and time
* Test executed by
* Test reviewed by.

**Roles and Responsibilities:**

|  |  |  |
| --- | --- | --- |
| **Employee name** | **Employee ID** | **Roles** |
| Name 1 | <ID> | Tester |
| Name 2 | <ID> | Tester |
| Name 3 | <ID> | Tester |

**Task allocation** (Usually not part of plan But based on assignment I am adding here)

**Approach 1:**

Tester 1:

* Test Instance Level Interactions (read, vread, update, patch, delete, history)
* Test Whole System Interactions (capabilities, batch/transaction, delete, history, search)

Tester 2:

* Test Type Level Interactions (create, search, delete, history)
* Security testing

Tester 3:

* Test Data Validation and automation framework integration
* Exploratory testing and adhoc tests
* Non functional testing

Each tester should also be responsible for reporting any defects, issues, or risks they encounter during testing. Communication among the testers and with the development team is crucial for ensuring that testing proceeds smoothly and efficiently.

It's also important to ensure that each tester has access to the necessary test data, test environment, and test tools to execute their assigned tasks effectively. Any dependencies or issues should be resolved before testing begins to minimize delays or interruptions.

**Approach 2:**

* Tester 1: Test Instance Level Interactions (read, vread, update, patch, delete, history)
* Tester 2: Test Type Level Interactions (create, search, delete, history)
* Tester 3: Test Whole System Interactions (capabilities, batch/transaction, delete, history, search)

Option 2:

* Tester 1: Test Security and Authentication (OAuth2, TLS)
* Tester 2: Test Data Validation (resource validation, message validation)
* Tester 3: Test Performance and Scalability (response time, resource usage, load testing)

**Approach 3:**

* Tester 1: Test Instance Level Interactions (read, vread, update, patch, delete, history), Test Security and Authentication (OAuth2, TLS)
* Tester 2: Test Type Level Interactions (create, search, delete, history), Test Data Validation (resource validation, message validation)
* Tester 3: Test Whole System Interactions (capabilities, batch/transaction, delete, history, search), Test Performance and Scalability (response time, resource usage, load testing)

**Test Tools:**

Testing tools will be used to support the testing process. The tools to be used for testing are validated and ensured that the software is validated. These tools may include automated testing tools, performance testing tools, security testing tools, and defect tracking tools.

**Tests Reusability across releases:**

* All test cases will be reviewed and evaluated for their potential for reuse in future testing cycles.
* The test environment and test data used in the current testing cycle will be evaluated for their potential for reuse in future testing cycles.
* Using a data driven test framework for API testing and active maintenance of the framework periodically
* Formal release notes after releases for ensuring if test sets differ
* Json/xml files maintained separately and ensuring the same can be reused
* Using the tools like Jenkins to have continuous triggers of execution of tests on every code merge and commit.
* Better traceability matrix which creates a trace between the requirements and tests and keeping a record for design changes
* Better quality management system for ensuring the documents are stored and good documentation practices
* Collaboration between test and developments teams to be transparent and updated on the teams updates/modifications and architecture changes
* Test scripts, tools, and other resources used in the current testing cycle will be evaluated for their potential for reuse in future testing cycles.
* All reusable test cases, test data, and test resources will be documented and stored in a centralized repository for easy access and reuse in future testing cycles.
* Test cases that have demonstrated their effectiveness in previous testing cycles will be marked for reuse in future testing cycles.
* Test cases that are not reusable as-is may be modified and updated for reuse in future testing cycles.

**Test Metrics:**

The below test metrics are generated and reported to the stakeholders:

System test metrics: Test coverage, Test run time and Number of tests

Defect metrics: Defect density, Defect removal efficiency, meantime to failures, Defect leakage

Statistics metrics: Test creation rate, test execution rate, test pass/fail rate

**Test Training:**

Testers will be provided with the necessary training to perform their testing tasks effectively. Training may include training on the RESTful API, testing techniques, and testing tools. Testers will also be trained on the appropriate standards and the necessary knowledge on tools used for the test processes

**Change Management:**

The document is a running document and changes to the testing process will be managed using a formal change management process. Changes may include changes to requirements, changes to the testing approach, and changes to the testing tools.

**Test Documentation:**

Test documentation will be produced throughout the testing process. This documentation may include the test plan, test cases, test scripts, test results, and the final test report. The documentation will be stored and maintained for future reference

**Test Standards and Guidelines:**

Testing will be conducted in accordance with industry standards and guidelines. These standards and guidelines may include IEEE standards, ISO standards, and best practices for software testing.

**References:**

|  |  |
| --- | --- |
| IEEE Standard for Software and System Test Documentation | <Revision and version> |
| REST architectural style documentation |  |
| API specification and requirements documentation | <Revision and version> |
| Software requirement specification | <Revision and version> |
| Defect management SOP | <Revision and version> |
| Test strategy document | <Revision and version> |

**Version Control:**

|  |  |  |
| --- | --- | --- |
| **Date** | **Author** | **Comments** |
| 25 April 2023 | MANJUNATH M K | Initial Draft for Test plan of Server FHIR API interactions |
|  |  |  |