Assignment

1st part of the assignment,

Test Plan

Purpose

Testing of Firely Server with respect to the specification of the FHIR RESTful API

Test Environment

- Public instance of Firely Server to test: https://server.fire.ly.
- FHIR Version to be tested: R4
- Base Endpoint: https://server.fire.ly/R4
- Postman is chosen as the tool to test the RESTful APIs.

Features

FHIR is described as a 'RESTful' specification. Resource structures and interfaces are standardized. Each "resource type" has the same set of interactions which is used to manage the resources.

Instance Level Interactions	
read	Read the current state of the resource
vread	Read the state of a specific version of the resource
update	Update an existing resource by its id (or create it if it is new)
patch	Update an existing resource by posting a set of changes to it
delete	Delete a resource
history	Retrieve the change history for a particular resource
Type Level Interactions	
create	Create a new resource with a server assigned id
search	Search the resource type based on some filter criteria
delete	Conditional Delete across a particular resource type based on some filter criteria
history	Retrieve the change history for a particular resource type
Whole System Interactions	
capabilities	Get a capability statement for the system
batch/transaction	Perform multiple operations (e.g. create, read, update, delete, patch, and/or [extended operations]) in a single interaction
delete	Conditional Delete across all resource types based on some filter criteria
history	Retrieve the change history for all resources
search	Search across all resource types based on some filter criteria

Each request has multiple combinations of the following:

Author: Aswathi Krishnan

1. Endpoints
2. Request body
3. Response status code
4. Response body
5. Business Rules
6. Resource Types

Assumptions (One of this can be true):

Tests are automated and new feature can be tested manually and then add a test for it.

- Automated tests can run which a tester have to oversee and report the findings to the test lead
- Add a new test and get it reviewed
- If Automated tests exist – I would say it is a Medium sized effort for 3 testers for round 1 of testing (assuming no issues are found)

Testers have only a documentation about these endpoints and needs to write new postman tests and then test it during releases

- This would be a new automation project which is a XL task involving 3 testers for a sprint given they work only on automating for 3 resource types.

Features need to be checked without any tests

- Exploratory testing with an existing example JSON for 3 resource types.

Based on the above 3 methods of testing

1. Existing Automated Functional Test scripts in Postman/Newman for Regression testing
2. Manual Testing with Postman for new features
3. Adhoc - Exploratory Testing

Test Data

Source:
http://hl7.org/fhir/R4/patient-example.html
http://hl7.org/fhir/R4/observation-example-f001-glucose.html
http://hl7.org/fhir/R4/allergyintolerance-example.html

Features In Scope for testing

The following resource types can be tested for all endpoints:

1. Patient
2. Observation
3. Allergy Intolerance
4. All Status codes except 422

Features Out of Scope

1. Other resource types
2. Business Rules – Out of scope
3. Tests for 422

How to make the tests repeatable on every release

1. Regression testing – should be automated as much as possible.
2. Create a test data for chosen combinations in a Json array and use it as an external file during test run. These combinations will become iterations for the request.
3. Request body can come from external files – So if there is a modification, changes can be made without getting into the test itself.
4. Make sure to delete the resources created at the end of the test
5. Error texts, codes etc can also be in a data file

Roles and Responsibilities

Resource Type patient has a lot of combinations. Tester who is given this is not given too many other tasks.
Assigned tester is accountable for the endpoints and resource types, if its manual or automated.
If Automated: the endpoints can be automated and various folders in collections can be run by the QA team so that there is no repetition.

Testers will discuss the following outcomes of the testing to the test lead.

- Test result: Passed/Failed and skipped tests
- Test Data
- Observations
- Defects

Instance Level Interactions	**Adhoc Testing**	**Resource: Patient**	**Resource: Observation**	**Resource: Allergy Intolerance**
read	Tester2 with Patient who has an observation and allergy	Tester1	Tester2	Tester3
vread		Tester1	Tester2	Tester3
update	Tester2 with Patient who has new allergy	Tester1	Tester2	Tester3
patch	Tester2 with Observation referenced to a patient	Tester1	Tester2	Tester3
delete		Tester1	Tester2	Tester3
history		Tester1	Tester2	Tester3
Type Level Interactions				
create	Tester2 with Allergy	Tester1	Tester2	Tester3
search	Tester3 with Patient	Tester1	Tester2	Tester3
delete		Tester1	Tester2	Tester3
history		Tester1	Tester2	Tester3
Whole System Interactions				
capabilities		Tester1	Tester2	Tester3
batch/transaction		Tester1	Tester2	Tester3
delete		Tester1	Tester2	Tester3
history		Tester1	Tester2	Tester3
search	Tester3 with Patient	Tester1	Tester2	Tester3

2nd part of the assignment,

The following has been designed as the Test Cases for Create endpoint for a Patient Resource

Request type	Test Case	Response status code	Validations
POST Request with Valid body	Verify that a patient can be created with Create endpoint and valid request Json body	201	ID Meta properties Schema Status check
POST Request with invalid endpoint	Verify that a patient cannot be created if the user gives an incorrect resource name in the endpoint	404	Status check Error code Resource type
POST Request with invalid data type	Verify that a patient cannot be created if the request body contains invalid data types	400	Status check Error code Resource type
POST Request with invalid data	Verify that a patient cannot be created if the request body contains invalid data values	400	Status check Error code Resource type
POST Request with id and meta properties	Verify that a patient is created with a new id even if user adds an id to the request body	201	Status check ID should not match request Meta should not match request
POST request with invalid data [violating a business rule]	Verify that a patient cannot be created if a business rule is violated in a request body	422	Status check Error code Resource type

Observation Resource

Request type	Test Case	Response status code	Validations
POST Request with Valid body and content type header	Verify that an observation can be created with Create endpoint and valid request Json body	201	ID Meta properties Schema Status check
POST Request with invalid endpoint	Verify that an observation cannot be created if the user gives an incorrect resource name in the endpoint	404	Status check Error code Resource type
POST Request with invalid data type	Verify that an observation cannot be created if the request body contains invalid data types	400	Status check Error code Resource type
POST Request with id and meta properties	Verify that a patient is created with a new id even if user adds an id to the request body	201	Status check ID should not match request Meta should not match request
POST request with invalid data	Verify that a patient cannot be created if a business rule is violated in a request body	422	Status check Error code Resource type

[violating a business rule]			

Patch Tests

Request type	Test Case	Response status code	Validations
PATCH Request with Json-patch with various operations	Verify that the changes requested in the body are made to the patient data	200	ID Meta properties Status check Changes are applied
PATCH Request with invalid endpoint	Verify that the changes requested are not made and error is displayed	404	Status check Error code Resource type
PATCH Request with invalid patient id	Verify that the changes requested are not made and error is displayed	404	Status check Error code Resource type
PATCH Request with invalid parameters	Verify that the changes requested are not made and error is displayed	5xx	Status check Error code Resource type
PATCH Request with invalid payload	Verify that the changes requested are not made and error is displayed	400	Status check Error code Resource type

3rd part of the assignment,

Public Test Automation Repository for **Create** FHIR endpoint for Patient and

Observation:
https://github.com/aswathikrish24/firely-test
Some observations for the tests:
Data values seems to work for most data types – testing them specifically was not included in the automated tests. Tried to check negative flows for these.
All except for error 422 has been automated.

Project: Firely Sever

	executed	failed
iterations	1	0
requests	15	0
test-scripts	30	0
prerequest-scripts	30	0
assertions	40	0
total run duration: 3.2s		
total data received: 23.79kB (approx)		
average response time: 130ms [min: 15ms, max: 769ms, s.d.: 175ms]		

	executed	failed
iterations	5	0
requests	5	0
test-scripts	10	0
prerequest-scripts	10	0
assertions	20	0
total run duration: 2.2s		
total data received: 3.27kB (approx)		
average response time: 353ms [min: 14ms, max: 917ms, s.d.: 395ms]		

I had fun automating it :)

Author: Aswathi Krishnan