2023-09 dx-PQ IG: Data Exchange Industry – Pharmaceutical Quality

Preparation

For these exercises you will need a REST client, ideally Postman - that is what demos will use.
Unless you are more experienced, please use that. The downloadable version is better than web-based version.
You will need a JSON and/or XML editor e.g. Visual Studio Code, or Oxygen (free trial).
Notepad is not really good enough, but may be usable as a last resort.

Overview

This set of exercises walks you through creating a set of connected FHIR resources, on a FHIR server, that describe a drug product, its constituents and manufacturer, and then records an analytical test on it and gives a conclusion. We do this by hand, but the same steps could be done by software, exporting data from different back-end systems, SQL tables etc. The data is then in sharable FHIR repository, and could be downloaded by a 3rd party who could then do the reverse and incorporate that into their back-end systems. This is working interoperability of structured CMC data.

After some introductory steps (1-3), learning to read and write data, each step starts with some existing data, from a file, that you customize and load to the server, connecting to the data from the previous step. Starting with a new MedicinalProductDefinition - the “header” of the drug product - in step 2 and 3, we add manufacturer in step 4 and 5, then add an ingredient in step 6. With that product description complete, we create a batch of it in step 7, add a test in step 8 and write a concluding report on the batch in step 9. Step 10 is more advanced and lets you define a test plan, using PlanDefinition and ObservationDefinition.

Note that that there is no expectation of finishing all the steps in any particular time frame.

Step 1 - Read a simple resource - MedicinalProductDefinition (MPD)

There is one on the test server here:
https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/demo-product-id-step-1

How can that be viewed?

FHIR data servers are webservers – but instead of serving HTML pages they serve XML or JSON data.
But they are still webservers, using HTTP, and you usually look at them with an ordinary web browser (e.g. Chrome).
If you visit the link above with our ordinary browser (click it, or cut and paste it to the address bar) you will see some JSON data.
(Note – some browsers will offer to download the data, instead of displaying it directly. You can save it as a file and open it in something like notepad.)
If you click the link on that page that says HTML XML, you will see some XML.
What you are seeing is FHIR data from the server, in one of two forms. Some servers default to JSON, some to XML, but you can usually work with either.

Technical note:
You can now guess that, because you can see the data in two formats (XML or JSON), the data is probably not stored in both formats. That would be very inefficient. So for one format at least it is probably being converted to your desired format on demand.
You can start to see that the storage of data and the FHIR view of it are somewhat separated. The FHIR data may be stored on the server as FHIR (maybe in JSON) and then converted to XML when you ask, or vice versa. Or a more realistic scenario is that it is stored as SQL data (perhaps in your companies CMC data base), and is converted to FHIR JSON or XML when you ask. (Advanced readers may recognise this as a "façade" – a translating FHIR layer built on top of a non-FHIR data source, allowing access to data in a FHIR format without duplicating any data.)

Notice that already we are seeing two aspects of FHIR. The data itself (XML and/or JSON) and the way to access it – the web address of this data item.
FHIR is data definitions (models, called resources) and access API for reading and writing it.

Let's look at that URL itself, in more detail:
https://dx-pq.lantanagroup.com/fhir/*MedicinalProductDefinition*/demo-product-id-step-1

1 The bold part is the web address of the FHIR server (which is a web enabled database).
2 The italic part is the name of one type of FHIR resource *MedicinalProductDefinition*
3 The last part is the FHIR record id of this particular piece of data (about a product called Stelbat Tablets). Data in FHIR is called resources.

Part 3 changes for every record – every resource has a different id.
If you omit this id on the URL it means "all of that type of resource" (the whole collection).
Try going to https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition and you will see some resources (there are 2 at the time of writing).

Part 2 changes for every type of record, every resource type.
Try going to https://dx-pq.lantanagroup.com/fhir/Organization to see a different type of resource.

Part 1 is the whole server. Try looking at http://hapi.fhir.org/baseR5/MedicinalProductDefinition to see some data on the public HAPI server (which is not related to today's workshop – but works more or less identically).

Now have another look at this resource data itself – the same page of data at https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/demo-product-id-step-1 (in XML or JSON as you prefer), and get a feel for it.
It is in a format that is documented here:
http://hl7.org/fhir/medicinalproductdefinition.html
And lower in the same page is an overview diagram (see UML tab)
http://hl7.org/fhir/medicinalproductdefinition.html#tt-uml
For an overview of how this FHIR resource relates to others see the Module Page
http://hl7.org/fhir/medication-definition-module.html

The front page of FHIR is here: http://hl7.org/fhir/ and has some introductions for executives, developers etc on that page.

There is more detail about using this resource type with CMC data in the Accumulus implementation guide for FHIR here:
https://build.fhir.org/ig/HL7/uv-dx-pq/
and specifically this page

https://build.fhir.org/ig/HL7/uv-dx-pq/StructureDefinition-MedicinalProductDefinition-drug-product-dxpq.html
A FHIR implementation guide takes the core of FHIR (that everyone uses) and makes some extra rules about how to use FHIR in a specific domain, use case, or community.

Back with our resource, notice that it has an "id" at the top "demo-product-id-step-1". Note that this always matches the end part of the URL.
In FHIR, a resource's id (its identity) is synonymous with its location on the server – its URL. URLs and FHIR ids are two sides of the same coin.

Read down the data and at the end notice that it is linked to another resource "Organization/manufacturer".

That is another resource type ("Organization") and another id ("manufacturer").
How would you view that data?
Knowing about the structure of FHIR URLs we can see that that resource lives and, and can be viewed at: https://dx-pq.lantanagroup.com/fhir/Organization/manufacturer
Go there and see it.
See http://hl7.org/fhir/organization.html#tt-uml for details about that resource type. It is included in the DX-PQ IG here https://build.fhir.org/ig/HL7/uv-dx-pq/StructureDefinition-Organization-dxpq.html.

We now know that FHIR data consists of multiple resources, with their own ids, and their own URLs, which are linked to each other. It's actually like an intranet of linked pages of data. But not one of HTML pages - one of XML or JSON data pages.

This is powerful, but can be hard to view – getting an overview of the linked data would be nice.

A site that can do this is https://vhewer.azurewebsites.net/display-fhir
Visit that page in your browser and then enter the address of our demo product in the "FHIR Resource URL:" box e.g. put in the full FHIR URL https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/demo-product-id-step-1. Then press Display FHIR. This takes 10 seconds or so, and then shows a view of the product including Organization/manufacturer, which it is linked to.
If you click on the bold word "Organization" it will take you to page for that resource on its own. Use the back button to go back.
If you click "show debug" at the top right you see some behind the scenes information, such as the id (hidden otherwise), the URLs that are used to find the different parts of thus set of data and some direct links to documentation.

This site works as a viewer on top of other FHIR databases. We can use this site to display our own data as we build it up today.

Now look at a different product id on our same server: demo-product-id (same as before but without "-step-1" on the end).
The full URL of that will be https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/demo-product-id
View this with a browser. It looks almost identical to the other one (note the different id property). Now view it using https://vhewer.azurewebsites.net/display-fhir.
This has a little more data linked to it. An ingredient and a substance within that, as well as a second organization linked to that substance. This is five resources telling the story of one product, each resource having some aspect of the full picture. Note that these can be assembled step by step – and parts can be shared. The same manufacturer would be used by many products, and perhaps even the same substance in some cases (especially for excipients).

Question – if the resource looks almost the same in XML/JSON (as the one from step 1), why does it look different in the viewer?

Recap - Your actions for Step 1 (for this you only need a web browser)

- Visit https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/demo-product-id-step-1, using a browser and see both JSON and XML versions
- Examine the XML or JSON for yourself.
- Look at the FHIR definition of this resource.
- Now look at id: demo-product-id
- Open that resource in https://vhewer.azurewebsites.net/display-fhir and see the organization and the ingredient and substance. Try debug more. Explore the links.

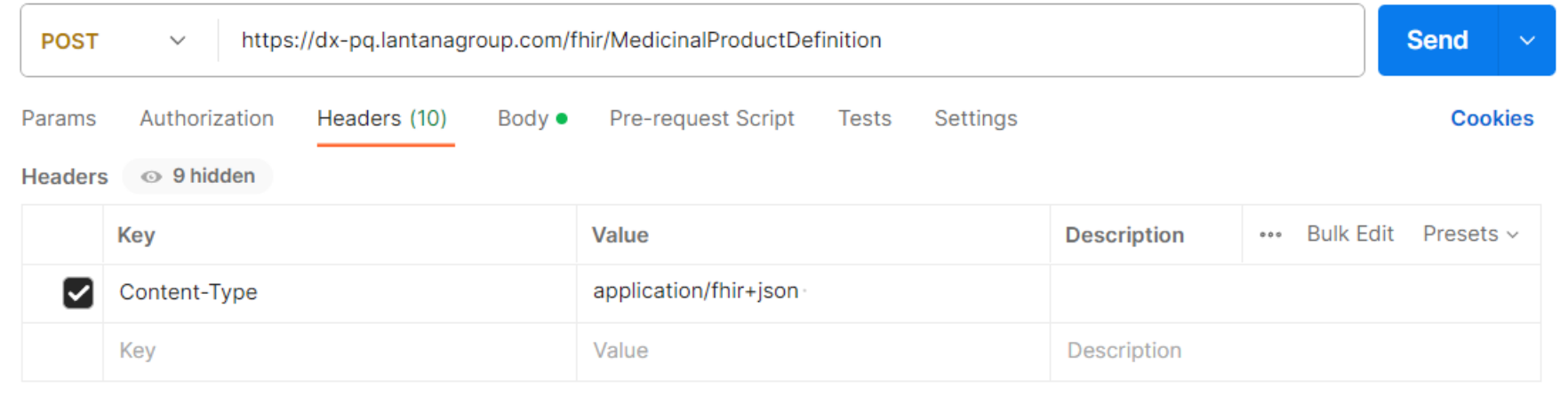
In Step 2 we are going to see how to build a structure like this.

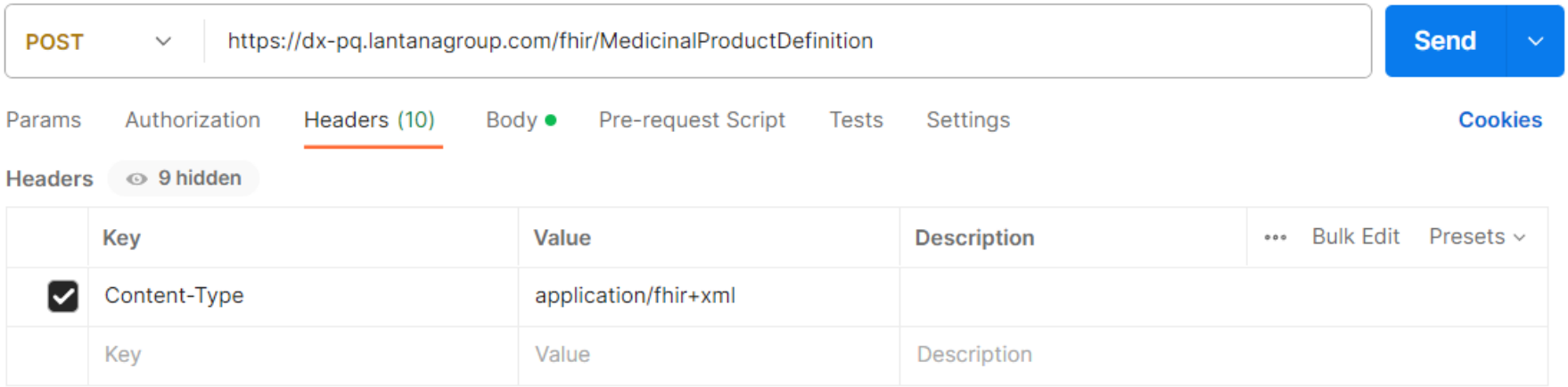
Step 2 - Make your own product

We are going to add our own product to the server.
You need one to start with:
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/MedicinalProductdefinition-Starting-Point.xml
or if you prefer JSON:
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/MedicinalProductdefinition-Starting-Point.json

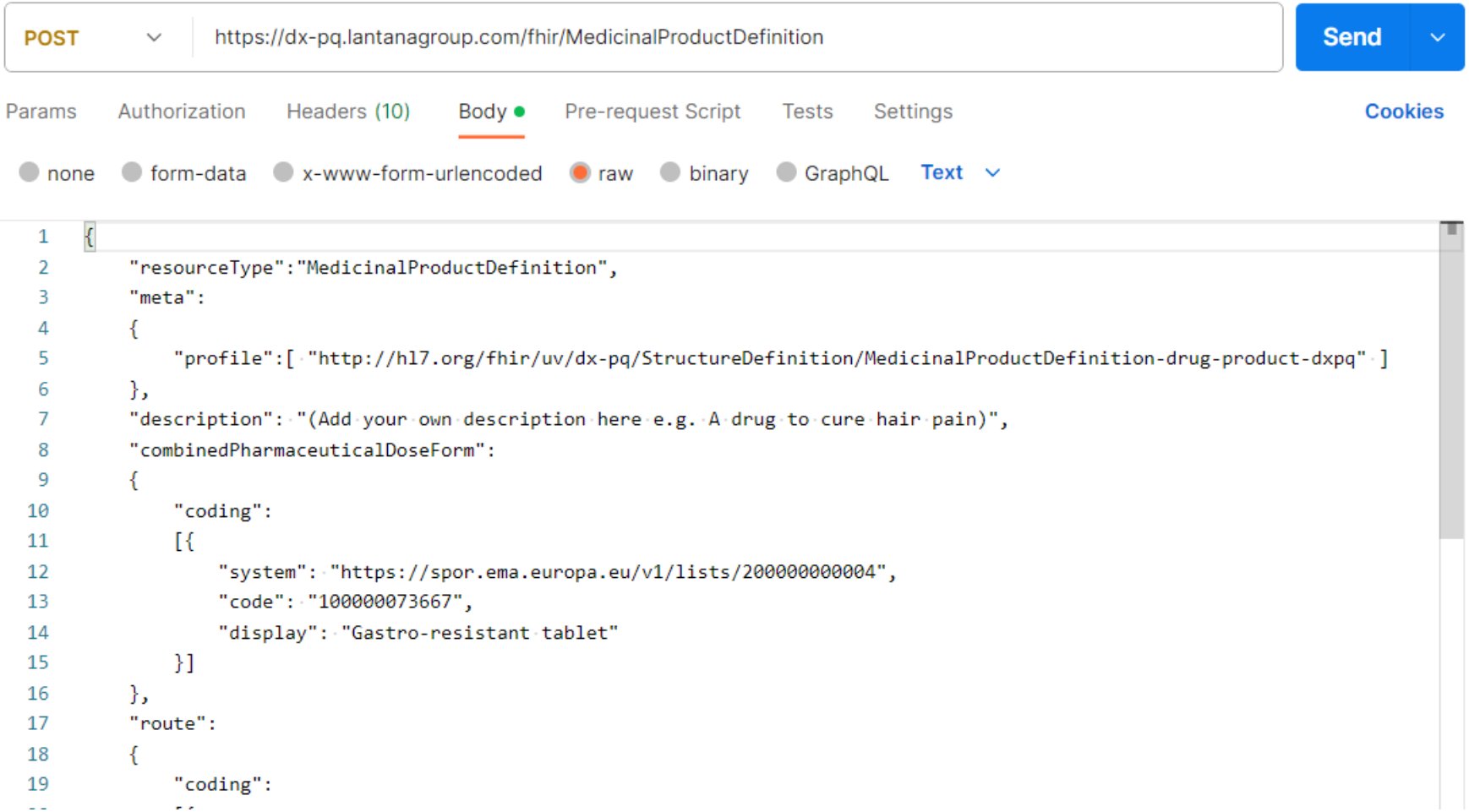
Open this file on your editor.
Add some test data to make this unique to you. There are notes in brackets where you can add a description and a drug name.
Anything different will do, so that you can recognise your product on the server.
You can change any other data that you like (but need to keep the overall structure conforming to FHIR).
Note that unlike the examples on the server these examples don't have an id.
That is because FHIR ids are assigned by the server when the data is written to the server.
Also at this point there is no manufacturer. We will add that later.

Use Postman to send (POST) this data to the server
The URL is https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition
You will need to set the content-type http header to application/fhir+xml or application/fhir+json accordingly.





And the JSON or XML goes in the body:



POST that and you should get a “201 created”. An Id will be assigned and the new version of the resource, with id added, will be returned in the body. The Id will also be indicated in the returned http headers.
The server will tell you (with a 400) if there was a problem.

It is possible to validate FHIR XML or JSON with schemas, but it is beyond the scope of today. But the server does a basic validation that the FHIR is correct, and will reject bad data.

It is also possible to check how your data looks before it gets to the server by pasting it at
http://vhewer.azurewebsites.net/display-fhir-paste

Now you have the id of the data that you uploaded, GET the data back in a different format.

If you loaded JSON, try to GET the data back as XML or vice versa. Use GET with the URL plus your id.
https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/{your-generated-id}
and set the accept header to application/fhir+xml or application/fhir+json.

Tip – duplicate the Postman tab, when doing a GET, so you can go back to your previous one if you need to.

Once you have successfully got your data back, you can post the address to the chat, with your name, and what you did - so that we can see your progress and have a look at your data
e.g. Alex J created https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/12345

You can also display the data using
http://vhewer.azurewebsites.net/display-fhir
and check it is as expected.

Step 3 - Edit your data

Using the same data as you previously loaded, make a change to it – to the name or the description perhaps - and PUT it back on the server, updating it.

Unless you add the existing id to your data (or you use the copy that came back from the server after the POST, or from your GET), the server will not allow you to update it. It must have the id.

PUT uses the same URL as GET e.g.
https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/{your-generated-id}

Put your id back into the chat e.g.
Alex J updated https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/12345

Step 4 - Add an Organization

You can either find an Organization on the server, or get in file from the folder
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/Organization-Starting-Point.xml
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/Organization-Starting-Point.json

Change the data to something specific to you. Change the name and/or address.
Then POST it to the server.
Note the Id that is assigned.

Post to chat that you done this step
e.g. Alex J created organization https://dx-pq.lantanagroup.com/fhir/Organizaion/456

Step 5 - Link the Organization to the MPD

Now to link this as a manufacturer of your product.

You need to update your product (same one from the last step) and add a manufacturer reference.
You can see the XML or JSON for this by looking at
https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/demo-product-id
Look at that using Postman (GET), or a browser. Add the “operation” element to your own product (using an editor, then a PUT).
Use the references above to check the FHIR definition if you need to.
Use the Id of your own recently added Organization in the reference e.g. “Organization/{your-id}”, rather than the Organization/manufacturer one.

Once this is done, check how your product looks using http://vhewer.azurewebsites.net/display-fhir. If things worked correctly it should display the product and the linked organization.
Post to chat that you done this step
e.g. Alex J linked org to https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/12345

Step 6 - Add an Ingredient

The Ingredient is what makes up the drug (with a certain strength). It can be just the name of a substance, or can use a code, or it can reference a SubstanceDefinition resource.
For simplicity at this step we will just use the name of a substance (but the optional extra step 6.5 will add a SubstanceDefinition).

So, first add an Ingredient, using:
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/Ingredient-Starting-Point.xml
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/Ingredient-Starting-Point.json

Note that the Ingredient must link to your product (MedicinalProductDefinition), by its Id.
Choose a name for your substance. Then add the Ingredient to the server, as with previous resources.
You can display the link with http://vhewer.azurewebsites.net/display-fhir.

Post to chat that you done this step
e.g. Alex J linked ingredient to https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/12345

Step 6.5 - Create and link a SubstanceDefinition to the Ingredient (Advanced/Optional)

This means adding a SubstanceDefinition resource, and then editing the Ingredient so that the code/concept is a reference, instead of just text.

You can look at https://dx-pq.lantanagroup.com/fhir/Ingredient/productIngredient to see how this is done

Use these files or one borrow from existing resources on the server.
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/SubstanceDefinition-starting-point.xml
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/SubstanceDefinition-starting-point.json

Note also that this SubstanceDefinition file has a manufacturer (e.g. in JSON, "reference": "Organization/{id-of-of-your-organization}")
Fill in the id of your Organization – or this will fail when you POST it. Even better, add another Organization, with a different company name, which makes the substance (rather than the product as a whole) and use that id.

Post to chat that you done this step
e.g. Alex J linked SubDef to https://dx-pq.lantanagroup.com/fhir/MedicinalProductDefinition/12345

Step 7 - Add a test Batch

The main product is now basically complete, as a MedicinalProductDefinition and associated resources. This product definition can be used repeatedly, connected to various protocols, manufacturing plans, analysis procedures and so on. A MedicinalProductDefinition represents *type* information about the product. It doesn't represent any particular actual physical *instance* of it. It is not something you could see, on a given date, and run tests on. For that we need a Medication resource. This represents a batch of the product, and we can make lots of them, to represent different physical batches with properties that we can test and report.

Add a Medication resource and use an extension to link it to the MedicinalProductDefinition.

Use these files
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/Medication-Starting-Point.xml
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/Medication-Starting-Point.json

You can add a batch number of your choice (batch.lotNumber)

You can display the Medication with http://vhewer.azurewebsites.net/display-fhir, and it should show a clickable link to the MedicinalProductDefinition.

Step 8 - Add a test, which is an Observation

An Observation is a test that has been done on a batch (a Medication). The observation has the Medication resource as its subject.

Use these files:
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/Observation-Starting-Point.xml

https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/Observation-Starting-Point.json

Displaying the Observation should show the subject, and the name of the related product.

Step 9 - Add a DiagnosticReport

The final part of the solution is to add a report of the observation, on the batch. This is like a document that wraps up a set of individual tests (observations) in reportable format, with a conclusion.

Use these:
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/DiagnosticReport-Starting-Point.xml
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/DiagnosticReport-Starting-Point.json

It is possible to display the DiagnosticReport with http://vhewer.azurewebsites.net/display-fhir and see the report, observation, and batch all together. There is a link to the product, which has the ingredients and all the other data.

Step 10 – Add a protocol (Advanced)

The next logical part is to create the testing protocol in FHIR. This could be for a Batch Analysis, or a Stability Report. This is the set of required activities for what needs to be done, rather than the actual tests and conclusion themselves.

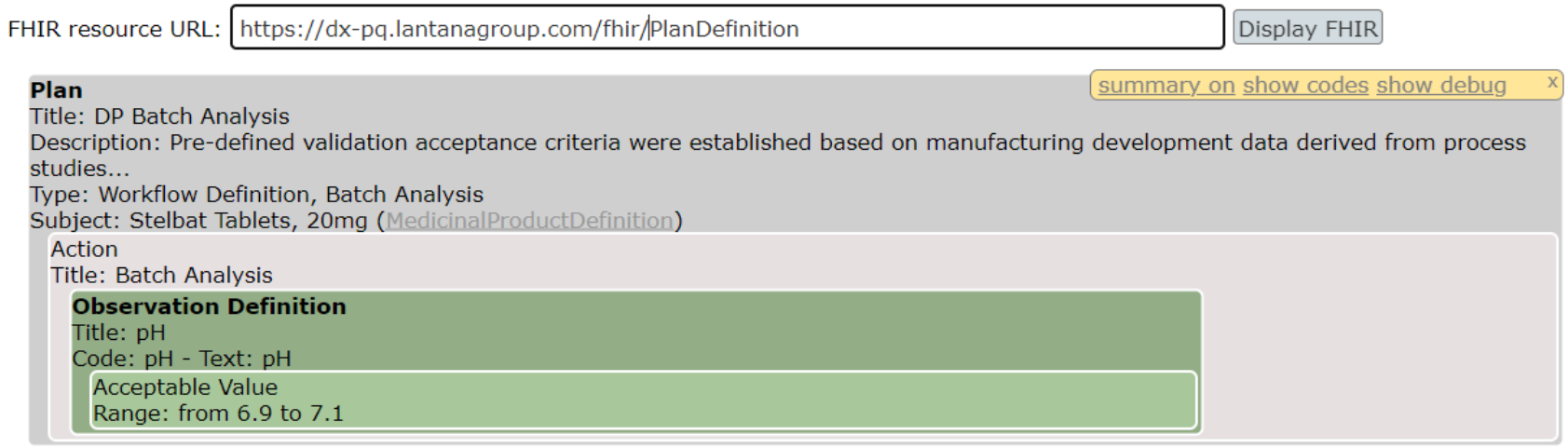
This needs to use the FHIR PlanDefinition resource, for the pattern of steps – what is tested and when.

The individual tests are described with ObservationDefinitions. They describe what the test is, the method, and the expected values (pass/fail etc).

Use these 4 files:
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/ObservationDefinition-Starting-Point.xml
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/ObservationDefinition-Starting-Point.json
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/PlanDefinition-Starting-Point.xml
https://github.com/HL7/uv-dx-pq/blob/master/additional_data/connectathon_2023-09/sample_data/PlanDefinition-Starting-Point.json

You will want to add the ObservationDefinition first. Then add the PlanDefinition and link it to the MedicinalProductDefinition as its subject and to the ObservationDefinition as one of its actions.

The result may look like this:



The Observation added at Step 8 can be set to “instantiate” the corresponding ObservationDefinition. You can edit that Observation to add an “instantiatesReference” to the ObservationDefinition e.g. in XML

```
<Observation>
  <id value="example-observation-id"/>
  <instantiatesReference>
    <reference value="ObservationDefinition/example-id-of-observation-definition"/>
  </instantiatesReference>
```

Your observation will now show a traceable link back to its definition (rationale).