CS6440 Practicum Project Final Submission

Group G: Dynamic FHIR IG

***Abstract*—**FHIR Implementations Guides are key to interoperable health information. Standardizing data representation and terminology is key to preserving semantic interoperability. FHIR IGs are crafted by groups of stakeholders and domain experts who must adapt their models to meet growing challenges. Example data is a key part of an FHIR IG, but the example data provided often is not comprehensive or becomes out of date. This project will explore dynamically creating example data based on an FHIR IG.

1 BACKGROUND AND SIGNIFICANCE

An implementation guide (IG) is a set of rules about how FHIR resources are used (or should be used) to solve a particular problem, with associated documentation to support and clarify the usage. For example, The US Core Implementation Guide is based on FHIR Version R4 and defines the minimum set of constraints on the FHIR resources to create the US Core Profiles. It also defines the minimum set of FHIR RESTful interactions for each of the US Core Profiles to access patient data. There are two different ways to implement US Core: Profile Only Support: and Profile Support. But for each US Core Profile, it usually only provides one or two examples that are not comprehensive or even outdated. Users need more examples to better understand the implementation guide. More importantly, Users require large datasets which follow the specific implementation guide to do academic research such as studies in population health. These datasets are also frequently used in the Health IT industry and Policy Formation. This is why creating example data based on an FHIR IG is very important.

Name	Flags	Card.	Type	Description & Constraints
Patient		0..*	Patient	Information about an individual or animal receiving health care services
identifier	Σ	1..*	Identifier	An identifier for this patient
system	Σ	1..1	uri	The namespace for the identifier value
value	Σ	1..1	string	The value that is unique within the system.
name	Σ I	1..*	HumanName	A name associated with the patient **us-core-8:** Either Patient.name.given and/or Patient.name.family SHALL be present o be present.
family	Σ I	0..1	string	Family name (often called 'Surname')
given	Σ I	0..*	string	Given names (not always 'first'). Includes middle names This repeating element order: Given Names appear in the correct order for presenting
telecom	Σ	0..*	ContactPoint	A contact detail for the individual
system	Σ I	1..1	code	phone | fax | email | pager | url | sms | other **Binding:** ContactPointSystem (required): Telecommunications form for contact point.
value	Σ	1..1	string	The actual contact point details
use	?! Σ	0..1	code	home | work | temp | old | mobile - purpose of this contact point **Binding:** ContactPointUse (required)
gender	Σ	1..1	code	male | female | other | unknown **Binding:** AdministrativeGender (required)
birthDate	Σ	0..1	date	The date of birth for the individual
address	Σ	0..*	Address	An address for the individual

Figure 1—Snapshot Table of US Core Patient Profile IG.

2 PROBLEM

Our mission is to provide high-quality, realistic but not real, patient data and associated health records covering every aspect of healthcare. So one problem is we cannot just generate random values (random string or random numbers) for each attribute, instead, we have to output realist fake values. And It's hard to find a realistic fake data library that targets HealthCare data. The second problem is we are supposed to be able to provide a large amount of data. In addition, parsing information from the implementation guide is quite challenging because we try to support any implementation guide but many of them don't follow the same format.

3 SOLUTION

With the help of Team Inferno, we found an existing tooling infrastructure which is called Synthea™ (generic realistic synthetic data generation). For each synthetic patient, Synthea data contains a complete medical history, including medications, allergies, medical encounters, and social determinants of health. Synthea™ also provides thousands of sample patients. By utilizing Synthea™, we solved most of our problems. Our task then narrowed down to build on the example data of Synthea™, use any implementation guide Structure Definition to plug in all information that is missing from the generated Synthea example data bundle.

4 COMPLEXITY OR EFFORT

We created a Spring boot application that takes the implementation guide and fetches all the attributes from each of the profiles. Also added a couple of flags that users can toggle with like allFields or must support and depending on these flags data generated will have corresponding fields which match those criteria. We used Docker to containerize our application and Swagger UI to document each service

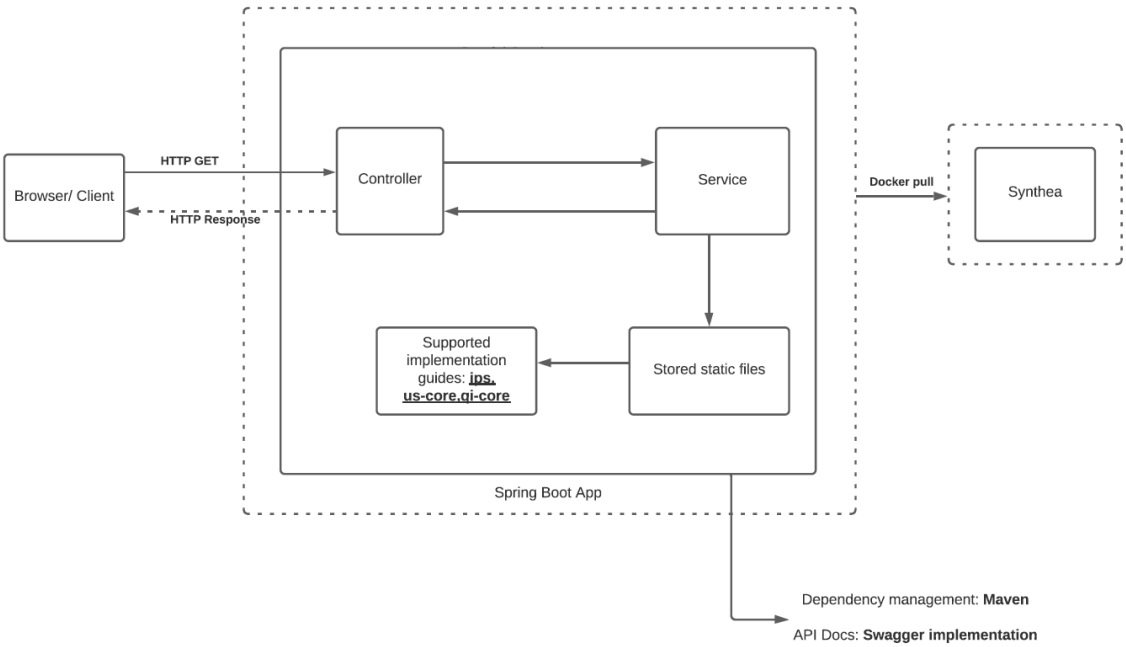

Figure 2—Architecture of the application.

We had multiple syncs up with mentors in the initial weeks to understand the project and clarify the questions regarding the same. Identified how the implementation guide is set up and the corresponding flags it has. Also, went through Synthea data and understood how the data is mapped with the profiles of implementation guides. Iterated the solution multiple times before reaching the final solution and the architecture (Figure 2) is updated to showcase the final approach which we took for this project.

5 REFERENCES

1. Synthetic patient generation. Synthea by the Standard Health Record Collaborative. (n.d.). Retrieved December 10, 2021, from https://synthetichealth.github.io/synthea/.
2. HL7.FHIR.US.CORE\index - FHIR V4.0.1. (n.d.). Retrieved December 10, 2021, from https://www.hl7.org/fhir/us/core/.