User Manual
Team Analytics - Alana Manuel, Anu Pradhan, Ashtyn Warner, Kirk Jensen
Supplemental Notifiable Condition
https://github.gatech.edu/gt-hit-fall2016/Supplemental-Notifiable-Condition

Before beginning use of the application, please be sure to read the special instructions document (entitled ‘Special Instructions - Team Analytics.pdf’) and follow the instructions listed there. This will ensure that you have all of the required software to execute the program.

Introduction

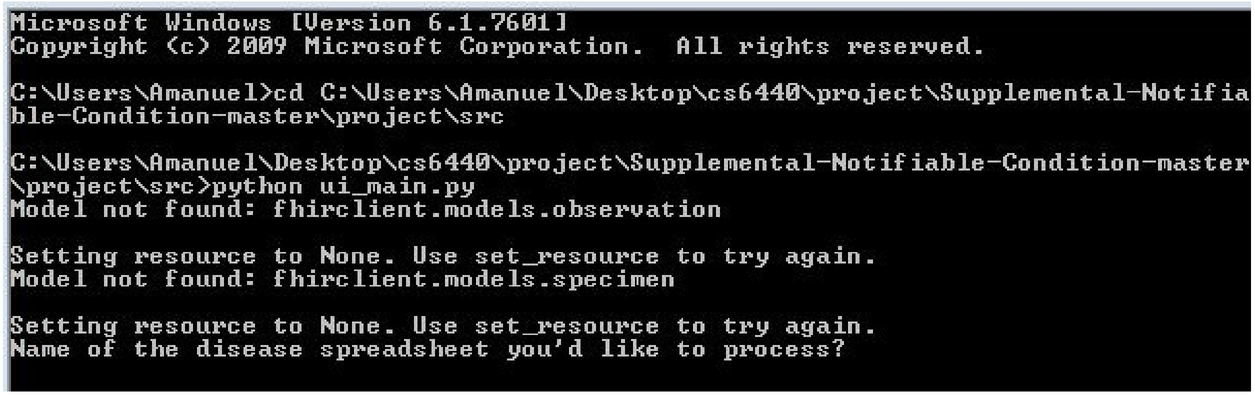
The application we developed aims to partially automate the process of mapping CDC-notifiable disease fields to FHIR Resources. The main UI for this is a Python console-based application that guides the user through this process using yes/no questions and prompts to enter specific information. This interface utilizes an object model that was developed to manage the Python objects that keep track of information and existing mappings, enable reading and writing to CSV files, enable searching, and enable output to JSON-format queries and FHIR Questionnaire entries.

The main distinction to be aware of when creating mappings is that some groups are mapped to FHIR Resources such as Observation, Medication, etc. These are considered “true” observations that have entries in an EHR. Many of the fields, however, do not correspond in a direct way to an element in an EHR, or are so specific that a patient would need to be asked directly. These types of fields should be grouped by medical concept in a Questionnaire, which is a FHIR Resource designed for precisely this type of situation.

Below you will find a walkthrough of the Main UI, followed by an overview of the underlying object model.

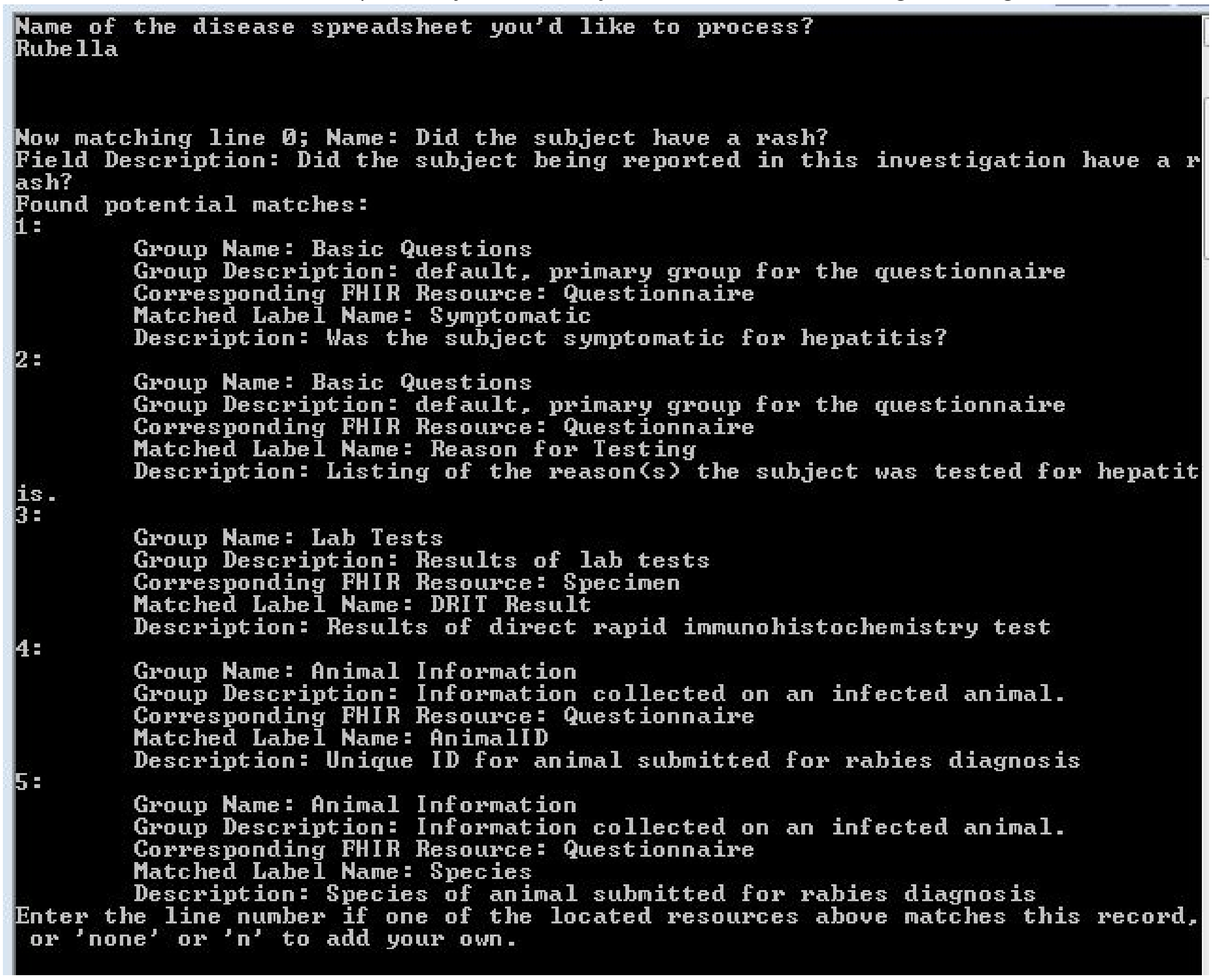
Walkthrough - Main UI

1. Begin by opening a terminal window and navigating to the application code folder.
2. Run the application by typing ‘python ui_main.py’. Once you do this you will see the following message:



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Amanuel>cd C:\Users\Amanuel\Desktop\cs6440\project\Supplemental-Notifia
ble-Condition-master\project\src

C:\Users\Amanuel\Desktop\cs6440\project\Supplemental-Notifiable-Condition-master
\project\src>python ui_main.py
Model not found: fhirclient.models.observation

Setting resource to None. Use set_resource to try again.
Model not found: fhirclient.models.specimen

Setting resource to None. Use set_resource to try again.
Name of the disease spreadsheet you'd like to process?
```

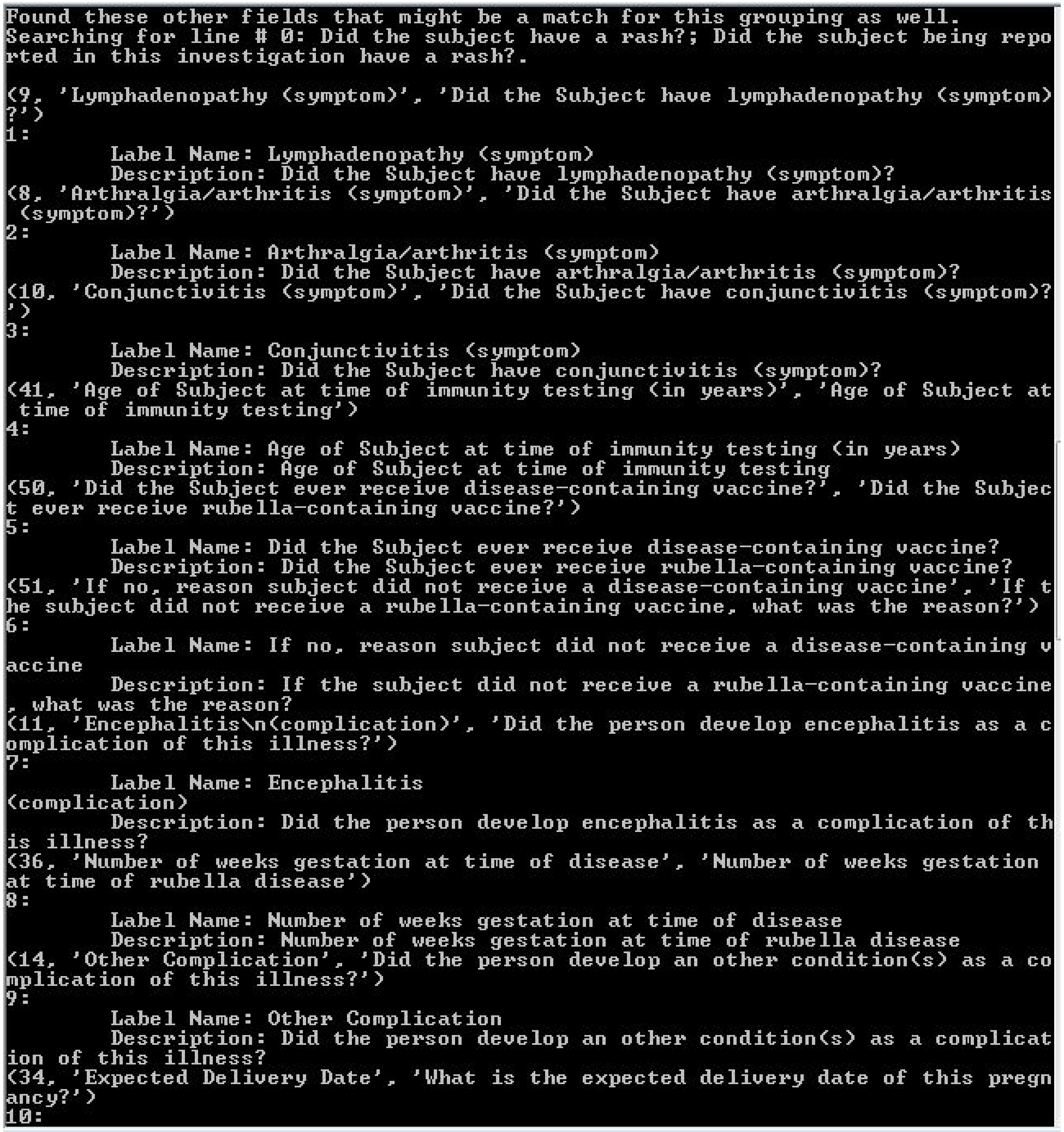
3. You then enter the name of the disease spreadsheet that you want to map. (You can check the data folder for the list of diseases.) Once you do this you will see the following message:



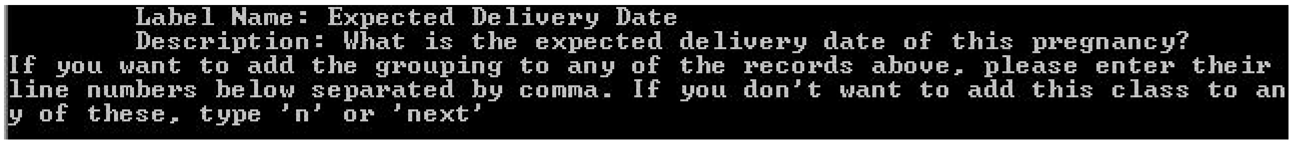
```
Name of the disease spreadsheet you'd like to process?
Rubella



Now matching line 0; Name: Did the subject have a rash?
Field Description: Did the subject being reported in this investigation have a r
ash?
Found potential matches:
1:
        Group Name: Basic Questions
        Group Description: default, primary group for the questionnaire
        Corresponding FHIR Resource: Questionnaire
        Matched Label Name: Symptomatic
        Description: Was the subject symptomatic for hepatitis?
2:
        Group Name: Basic Questions
        Group Description: default, primary group for the questionnaire
        Corresponding FHIR Resource: Questionnaire
        Matched Label Name: Reason for Testing
        Description: Listing of the reason(s) the subject was tested for hepatit
is.
3:
        Group Name: Lab Tests
        Group Description: Results of lab tests
        Corresponding FHIR Resource: Specimen
        Matched Label Name: DRIT Result
        Description: Results of direct rapid immunohistochemistry test
4:
        Group Name: Animal Information
        Group Description: Information collected on an infected animal.
        Corresponding FHIR Resource: Questionnaire
        Matched Label Name: AnimalID
        Description: Unique ID for animal submitted for rabies diagnosis
5:
        Group Name: Animal Information
        Group Description: Information collected on an infected animal.
        Corresponding FHIR Resource: Questionnaire
        Matched Label Name: Species
        Description: Species of animal submitted for rabies diagnosis
Enter the line number if one of the located resources above matches this record,
 or 'none' or 'n' to add your own.
```

4. It then runs through the list of associated fields for that disease one by one. Any field groupings from other diseases that are potential matches for the field in the current disease are listed. If one of those groups is appropriate, just type the associated number and hit enter. Otherwise, just press n.

5. For this particular example, I chose group 1. The following screen is shown when I do this:

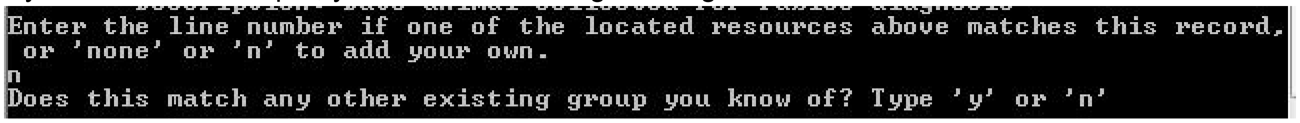


```
Found these other fields that might be a match for this grouping as well.
Searching for line # 0: Did the subject have a rash?; Did the subject being repo
rted in this investigation have a rash?.

(9, 'Lymphadenopathy (symptom)', 'Did the Subject have lymphadenopathy (symptom)
?')
1:
        Label Name: Lymphadenopathy (symptom)
        Description: Did the Subject have lymphadenopathy (symptom)?
(8, 'Arthralgia/arthritis (symptom)', 'Did the Subject have arthralgia/arthritis
 (symptom)?')
2:
        Label Name: Arthralgia/arthritis (symptom)
        Description: Did the Subject have arthralgia/arthritis (symptom)?
(10, 'Conjunctivitis (symptom)', 'Did the Subject have conjunctivitis (symptom)?
')
3:
        Label Name: Conjunctivitis (symptom)
        Description: Did the Subject have conjunctivitis (symptom)?
(41, 'Age of Subject at time of immunity testing (in years)', 'Age of Subject at
 time of immunity testing')
4:
        Label Name: Age of Subject at time of immunity testing (in years)
        Description: Age of Subject at time of immunity testing
(50, 'Did the Subject ever receive disease-containing vaccine?', 'Did the Subjec
t ever receive rubella-containing vaccine?')
5:
        Label Name: Did the Subject ever receive disease-containing vaccine?
        Description: Did the Subject ever receive rubella-containing vaccine?
(51, 'If no, reason subject did not receive a disease-containing vaccine', 'If t
he subject did not receive a rubella-containing vaccine, what was the reason?')
6:
        Label Name: If no, reason subject did not receive a disease-containing v
accine
        Description: If the subject did not receive a rubella-containing vaccine
, what was the reason?
(11, 'Encephalitis\n(complication)', 'Did the person develop encephalitis as a c
omplication of this illness?')
7:
        Label Name: Encephalitis
(complication)
        Description: Did the person develop encephalitis as a complication of th
is illness?
(36, 'Number of weeks gestation at time of disease', 'Number of weeks gestation
at time of rubella disease')
8:
        Label Name: Number of weeks gestation at time of disease
        Description: Number of weeks gestation at time of rubella disease
(14, 'Other Complication', 'Did the person develop an other condition(s) as a co
mplication of this illness?')
9:
        Label Name: Other Complication
        Description: Did the person develop an other condition(s) as a complicat
ion of this illness?
(34, 'Expected Delivery Date', 'What is the expected delivery date of this pregn
ancy?')
10:
```



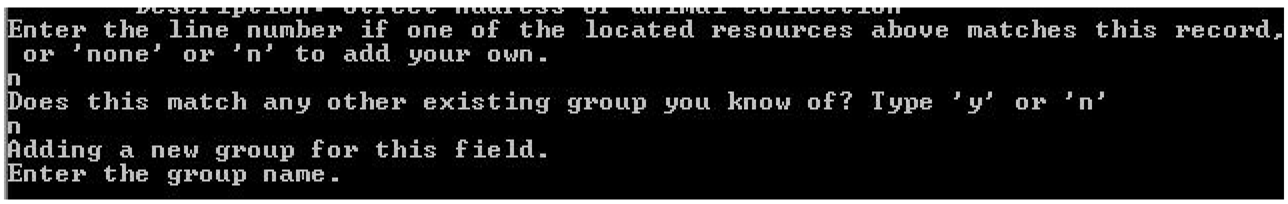
```
        Label Name: Expected Delivery Date
        Description: What is the expected delivery date of this pregnancy?
If you want to add the grouping to any of the records above, please enter their
line numbers below separated by comma. If you don't want to add this class to an
y of these, type 'n' or 'next'
```

6. You can then choose other fields associated with the disease that also belong in the same grouping. You can separate them with commas.
7. If you chose ‘n’ at step 4, you see the following message:



```
Enter the line number if one of the located resources above matches this record,
 or 'none' or 'n' to add your own.
n
Does this match any other existing group you know of? Type 'y' or 'n'
```

If from doing previous mappings, you know of an existing group that this belongs in, press ‘y’. You will then be given the option to enter the name of the group or search existing groups for a match.
If you don’t know of any other existing group, type ‘n’. You will then see the following screen:



```
Enter the line number if one of the located resources above matches this record,
 or 'none' or 'n' to add your own.
n
Does this match any other existing group you know of? Type 'y' or 'n'
n
Adding a new group for this field.
Enter the group name.
```

Once you enter the group name, you will be prompted to enter the description and appropriate FHIR resource. You will then be given similar options as in step 5 for what other fields you want to be in this new grouping.

Overview - Object Model

The object model contains several Python classes that form a hierarchical structure to efficiently manage mapping operations, searches, and JSON output. It's possible to manually accomplish everything discussed in the previous walkthrough by running main.py from an IPython console, and manually executing the various methods available. Since only Python developers would be comfortable doing this, the interface was developed. Each element of the object model is discussed below.

1. Data Manager
    a. location: manager.py
    b. class name: DataManager
    c. description:
        i. Data is managed using the DataManager class (automatically instantiated in main.py) so the user can access the objects for any disease, so that mapped fields can be cross-referenced (searched), and so that data can be opened or saved with a single function (constructor calls load_data(), and to_csv() saves all mappings and other progress).
    d. roles:
        i. Loads CSV files from the data folder, and saves any changes made to the mappings in the Group columns
        ii. Maintains a dictionary of {'Disease Name': Disease object} pairs.
        iii. Queries Disease members for mapping information to perform searches
    e. relationship to member objects:
        i. DataManager {1…*} Disease
2. Disease
    a. location: disease.py
    b. class name: Disease
    c. roles:
        i. Main point of entry for mapping and unmapping lines to FHIR resources.
        ii. Maintains a Pandas DataFrame containing the contents of the CSV file
        iii. Maintains a Questionnaire object for creating groupings, and mapping and unmapping lines to/from Questionnaire groups.
        iv. Maintains a list of FHIR Resource Groups (QueryGroup class) and can map and unmap lines to/from any group.
    d. relationship to member objects:
        i. Disease {1…*} QueryGroup
        ii. Disease {1...1} Questionnaire
3. Query Group
    a. location: querygroup.py

    b. class name: QueryGroup
    c. roles:
        i. Maintains a single group of lines, and information about the FHIR Resource being mapped to.
        ii. Maintains a list of FHIR query criteria
        iii. Generates a JSON string necessary for generating a FHIR query
        iv. Given a fhirclient server and patient ID, executes a FHIR query
4. Questionnaire
    a. location: questionnaire.py
    b. class name: Questionnaire
    c. roles
        i. Maintains groups of lines that map to a single logical concept, such as patient history or travel
        ii. Generates a JSON string that can be submitted as an entry in a FHIR server
    d. relationship to member objects:
        i. Questionnaire {1...*} Group
5. Questionnaire Group
    a. location: questionnaire.py
    b. class name: Group
    c. roles
        i. Contains description of a medical concept, and a list of lines that have been mapped to that concept.

Design Decisions

Data is imported from CSV files that are converted from the supplied Excel spreadsheets. Mapping information is stored back in those same CSV files so that people can work in parallel and save/retrieve each others' progress on github.

The data is parsed into Disease object, which exposes all the necessary functions for mapping and management. This includes functions for mapping and unmapping lines, adding new groupings for related lines, and saving mappings back to the CSV files in new columns "Group" "Group Description" "Resource". Having all necessary functionality exposed from a single object simplified the workflow.

Two very helpful search functions were added. The first, Disease.search_unmapped_lines(#) takes a line number, and does a word-search for the top n=10 unmapped lines in the current disease to facilitate easier grouping. This will print out the line numbers, labels, and descriptions for the top matches. The user can then decide if any are a good map to the current line they are examining.

The second search, Disease.search_mapped_lines(#) takes a line number (for the current disease), and looks through the descriptions of all mapped fields for similar fields. The purpose of this is to maximize commonality across all diseases that have been mapped to FHIR resources and Questionnaires. This outputs the top n=5 lines with label, description, group name, group description, and fhir resource. With this information, the user can then add a new group with the same or similar information, map the line used in the search to that group, and then use the previous method search_unmapped_lines to find other lines in the current disease that should be included in that group. Rinse, repeat.

The output for the Questionnaire is a JSON structure so that it can be easily ingested into a FHIR server. The output of other mappings to FHIR Resources is also JSON (use the as_json methods at the object level desired), which can be converted in a straightforward way to a FHIR query. This was done in the form of a python function that takes as input a fhirclient server object and performs a query, but since we don't know how the mappings will be used, we thought that outputting a JSON structure that could be easily repurposed would be the most future-proof output.

Future Work

One of the things that should be improved for a production environment is adding FHIR codable concepts and valid URI’s to the Questionnaire in order to generate fully compatible JSON output. This would be simple to add but wasn’t part of the mapping process.

Another way to improve the process would be to reduce the number of fields mapped to the Questionnaire. The questionnaire contains all questions that either can’t be found in a FHIR Resource, or would require a complex query joining multiple resources together. This type of query could be designed manually after groupings are made. To integrate this functionality would be a complex feature that would greatly increase the complexity of the interface and underlying logic, so wasn’t feasible to develop.