Design Rationale

The third assignment for this unit was designed and implemented in a way that supports easy **maintainability, extensibility** of the application. We wanted to ensure that should another observation be added to the system, the system would be able to easily track the new patient observation, with only very small changes. In our previous assignment we not only implemented the **Observer pattern** but also implemented it with the **MVC architecture**. This design decision from the previous assignment has helped us with assignment 3, so we didn't need to change that. However, we needed to refactor our code to now be able to support multiple observations as opposed to only tracking cholesterol.

We have also an ENUM called ObservationTypes which helps to ensure that the Observation text that is used by both view and model package is obtained from a central location. This helps avoid confusion/ mistakes while trying to reference a specific Observation text in the code. Furthermore, if there are any additional changes that need to be made to the text, it can be done easily to the ENUMs without having to change the references in the code that uses them.

With new understanding and requirements for the system, the model package has been refactored. Previously, FHIRObservation had value, units, system and timeRecorded as its fields as that was what an Observation was in our context. However, we now need to support having multiple values, units, systems and timeRecorded for a FHIRObservation such as the FHIRBloodPressure which has 2 different sets of values, units, systems and timeRecorded; 1 for systolic and 1 for diastolic blood pressure Observation. Hence, we did a **move field refactoring** from FHIRObservation to the newly created Observation class and at the same time **replaced data value with object**. To support the new requirements, **the rename method** of refactoring has also been heavily used.

On the functionality side of things, the FHIRObservation and its subclasses were added with the retrieveDataFromServerAndInit method and refactored the original method of that name to be retrieveDataFromServerAndInitWithLatest. The new method allows for getting the *ith most recent FHIRObservation* and thus when invoked repeatedly, allows us to get the top N most recent FHIRObservations or specifically for this assignment, the top 5 most recent FHIRBloodPressure of a FHIRPatient which contains the Systolic Blood Pressure readings/observations. However, for **extensibility** purposes, we have designed it such that we can get an arbitrary number of the top most recent Observations.

The PatientWrapper class is a Wrapper for Patient that works as a ‘connector/bridge’ to the View by **encapsulating** and hiding the complexity of the model package, especially the FHIRPatient class and its fields. The PatientWrapper class contains the method described above which repeatedly invokes the method retrieveDataFromServerAndInit to get the N most recent Observation(s) as well as a method to simply get the most recent Observation. It hides the complexity by allowing the service class i.e. the View to simply query using the Observation’s name which is the ObservationTypes ENUM’s toString().

The view seamlessly works with the model package. We developed the view package in such a way that if, in the future, we had to support multiple observations, apart from just cholesterol and blood pressure, the table in the view is automatically able to track the observations without any changes to it. It is to be noted that the class that was previously called PatientCholesterolSelectionGUI has now been renamed to PatientObservationMonitorGUI because the class is now able to monitor other observations as opposed to only monitoring patients’ cholesterol. In this class, we have used a list of

checkboxes to track what patient observations are tracked by the practitioner. By iterating through each patient being currently monitored and thereafter going through each observation checkbox selection based on what the practitioner has selected, we have been able to grab all the observation values for a patient using the wrapper data obtained from the PatientWrapper class in the model package. This will generate a row of patient observations data and will be then be updated in the table. Additionally, in this assignment, we had to create different graphs for different observations, but they were designed in such a way that should there be an additional observation that we need to plot either using the line chart or bar chart, we can simply create an instance of them without having to change anything in the chart classes, which significantly helps with **reusability** and **extensibility**. These two graphs extend the GraphAbstractionWindow. Essentially, we have used **Pull Up Field** and **Extract Superclass** in this case to have common code and attributes that are shared across the classes that inherit this class. The same can be noticed between Abstract class PatientSelectionGUI and PatientObservationsMonitorGUI.

Advantages of Pull Up Fields:

- Avoids duplication and redundancy of attributes in child classes
- Helps to avoid relocation of methods that are redundant by delegating it to a superclass from a subclass [1]

Advantages of Extract Superclass:

- Similar to Pull Up fields, we now have all the common fields and methods in the super class which eliminates code redundancy making it easier to maintain the code [2]

Advantages of Wrapper:

- Hides and encapsulates the complexity of the client [3]

Disadvantages of Wrapper:

- Introduces a tight coupling between the client and the Wrapper [3]

You can view our UML Class Diagram on LucidChart (if the attached copy is not clear) using this link: https://app.lucidchart.com/invitations/accept/479e3d70-82eb-483a-8536-039dad49a215

References:

[1]"Pull Up Field", *Refactoring.guru*, 2020. [Online]. Available: https://refactoring.guru/pull-up-field. [Accessed: 18- Jun- 2020].
[2]"Extract Superclass", *Refactoring.guru*, 2020. [Online]. Available: https://refactoring.guru/extract-superclass. [Accessed: 18- Jun- 2020].
[3]"Monash University - Sign In", *Lms.monash.edu*, 2020. [Online]. Available: https://lms.monash.edu/pluginfile.php/9940603/mod_resource/content/11/Lecture_slides/Week_11/FIT3077_ArchitecturalPatterns2.pdf. [Accessed: 18- Jun- 2020].

Assignment 3
Done by Zhi Hao Tan and Sriram Viswanathan