SOFTWARE ENGINEERING

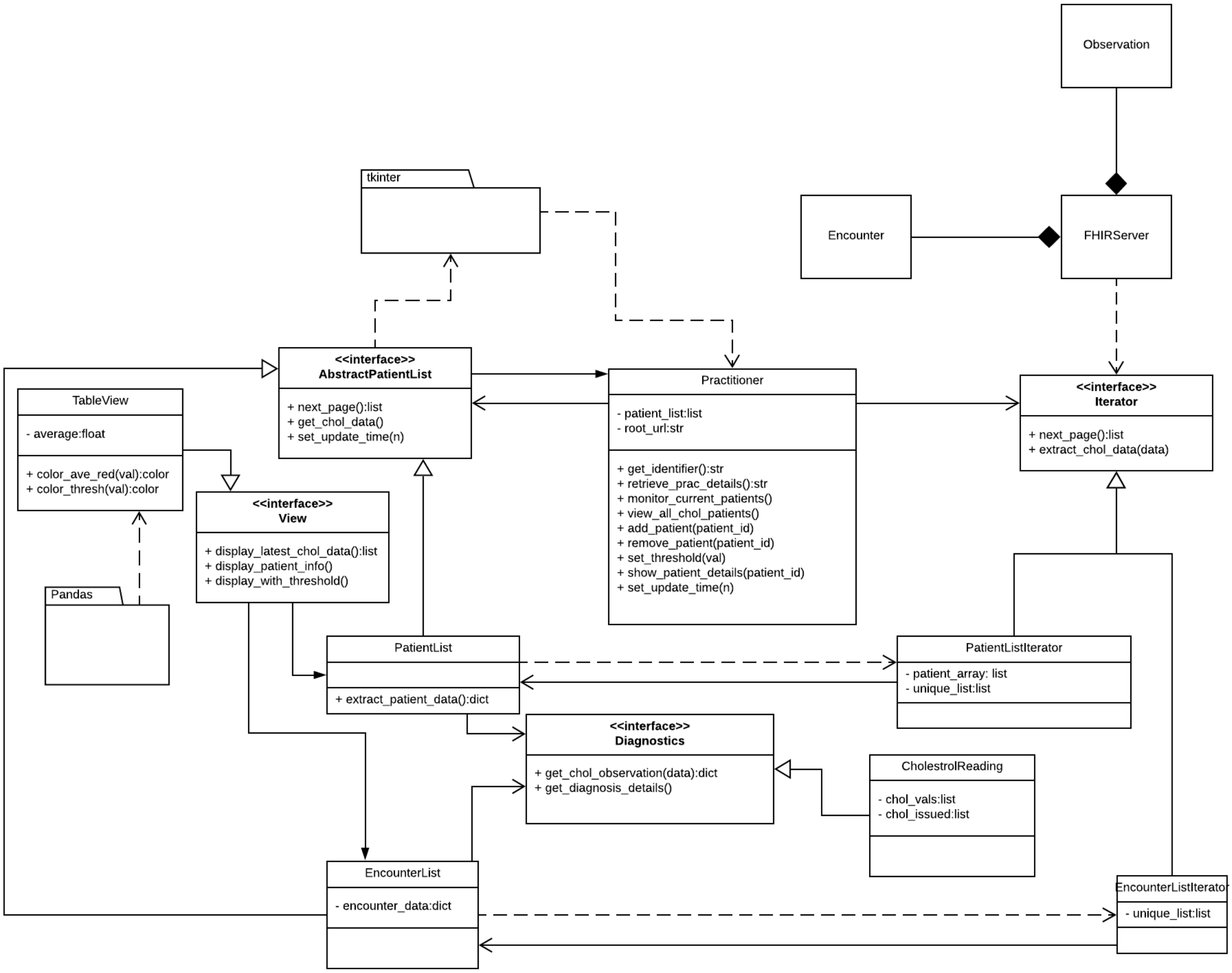
Assignment 2

Simone Dawn Matsukis 29061202
Akshay Inderpersad 28077490
Steve Harris 25509322

Contents

1.0 Class Diagram 2
2.0 Patterns & Principles 2
2.1 Iterator Pattern 3
2.2 Observer Pattern 4
2.3 The Open-Closed Principle 4
2.4 Single Responsibility Principle 5
3.0 Bonus Question 6
3.1 Linear Regression 6
3.2 Training Phase 6
References 8

1.0 Class Diagram



2.0 Patterns & Principles

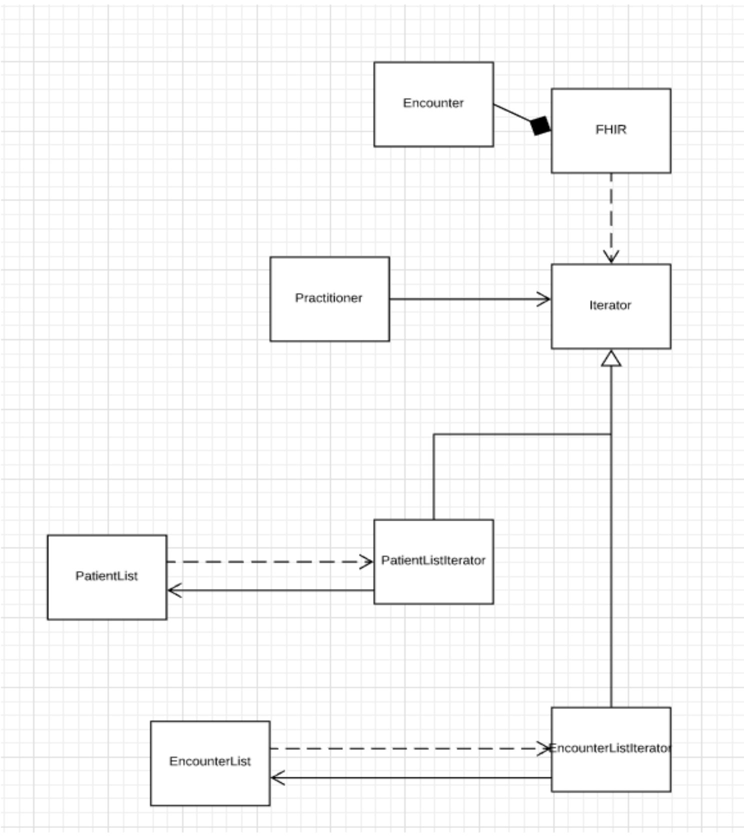
2.1 Iterator Pattern

The Iterator pattern is a behavioural design pattern, it is used when we need to sequentially access elements of a collection/aggregate object without having to understand its representation (Hodges 2019). The iterator keeps track of elements and knows which elements have already been crossed. In addition, variants can be added that won't impact the interface of objects nor the data structure (Hodges 2019).

The Iterator retrieves data from ***Encounter*** through the FHIR server. The class diagram designed above uses this pattern as a way for the practitioner to access the patient data. The ***Patient List Iterator*** will get the patients and their respective cholesterol from the ***Patient List*** class.

The ***Encounter List Iterator*** will get data related to each patient encounter. This data will then go to the ***Iterator*** and the iterator will loop through this data. The practitioner can access the patient and encounter information that the iterator has retrieved.

One of the main reasons for implementing this pattern was that it is an efficient way of accessing large amount of data/records (García 2018). By using the ***List Iterator,*** it allowed us to traverse a list forward and backwards (bi-directional) and is allowed to use other methods e.g. add(), next(), remove()



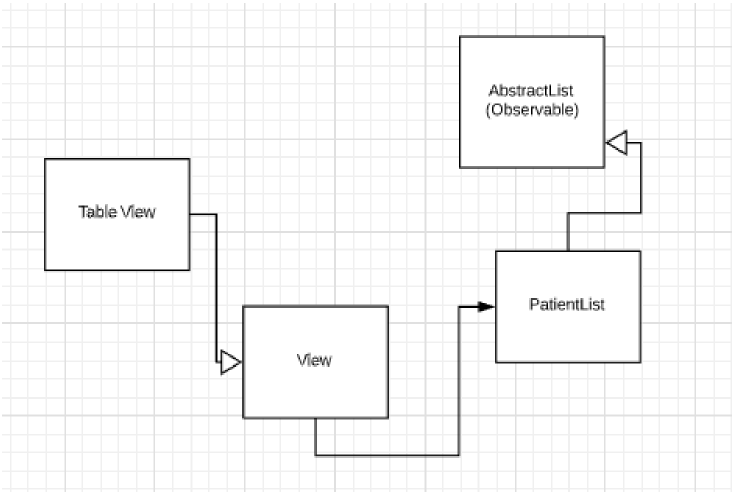
2.2 Observer Pattern

The Observer pattern is a behavioural pattern, this pattern is to be used when objects have a one-to-many relationship. When one of the objects are altered or modified, then any object that is dependent on the altered object will be automatically notified (Poyias 2019).

With our design, we have a class called ***View*,** this view class allows us to take the patient list and view it in different ways. Extending from ***View*** we have the class ***Table*View,** which will allow us to view the patient information in a particular format.

Essentially, ***View*** gets data from the returned lists of patients and displays this information in a table. The observer pattern will allow us to add on functionality, if the system wanted to add a different way of displaying the information, you would add another class extending from ***View***.

Using this pattern allows our classes to interact with each other and it allows us to send data to other objects effectively without any change in the Subject or Observer classes (Poyias 2019). We also implemented this pattern because it is easy to handle in terms being able to add or remove an observer at any time.

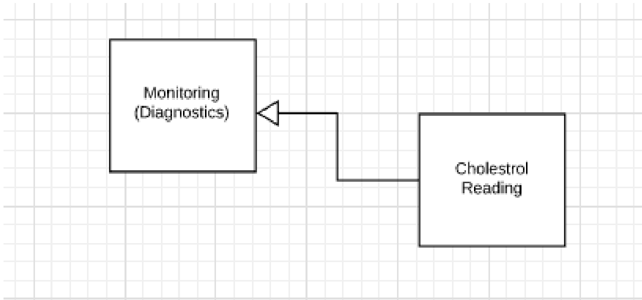


2.3 The Open-Closed Principle

Classes should be open for extension but should be closed for modification. This principle is about building classes in such a way that you can extend them with inheritance, once you created the class it no longer needs to be changed (Janssenmarch 2018).

In the class diagram, we have the class, ***Monitoring (Diagnostics*)** and extending from this class we have the ***Cholesterol Reading*** class. The class diagram was designed to use the open-closed principle so that if the system needs to add additional features to monitor different aspects of the patient, then this is possible by adding another class that will extend from the ***Monitoring (Diagnostics*)** class. For example, if there was to be an additional feature to monitor the patient's blood pressure, we can add a *blood pressure* class that extends from the ***Monitoring (Diagnostics)*** class without the need of modifying

The most notable reason for using this principle is because it allows us to extend our code without affecting existing code, this streamlines our code as well as makes maintenance easy while minimizing the risk of breaking the existing code (Despoudis 2017).



2.4 Single Responsibility Principle

Each class should focus on performing a single task, meaning that each class should have a unique responsibility. This follows the good design philosophy that stipulates a class should have only one reason to change (Perez 2018). With our class diagram design, each of our classes has its own separate responsibility, our classes each focus on completing a certain task. This ensures that our design has high cohesion as none of our classes is doing more than what it should do.

The added benefit to using this principle is classes, software components and microservices that have only one responsibility are easier to explain (Perez 2018).

3.0 Bonus Question

The machine algorithm we chose to predict what causes high cholesterol is:

3.1 Linear Regression

It is used to estimate real values (cost of houses, number of calls, total sales etc.) based on continuous variables. To find the characteristic that we think causes high cholesterol we first established a relationship between the independent(y) and dependent(x) variables by fitting the best line. This best fit line is known as the *regression line* and represented by a linear equation **Y= aX + b**.

The first step we took:

3.2 Training Phase

We took a randomly selected specimen of patients from the FHIR server (**training data**), and made a table of all the physical characteristics of each patient and chose the best variable to test cholesterol to test would be **age**, along with the age we needed to get the total cholesterol value and that was done by taking the latest **issued** test which would be our output variable.

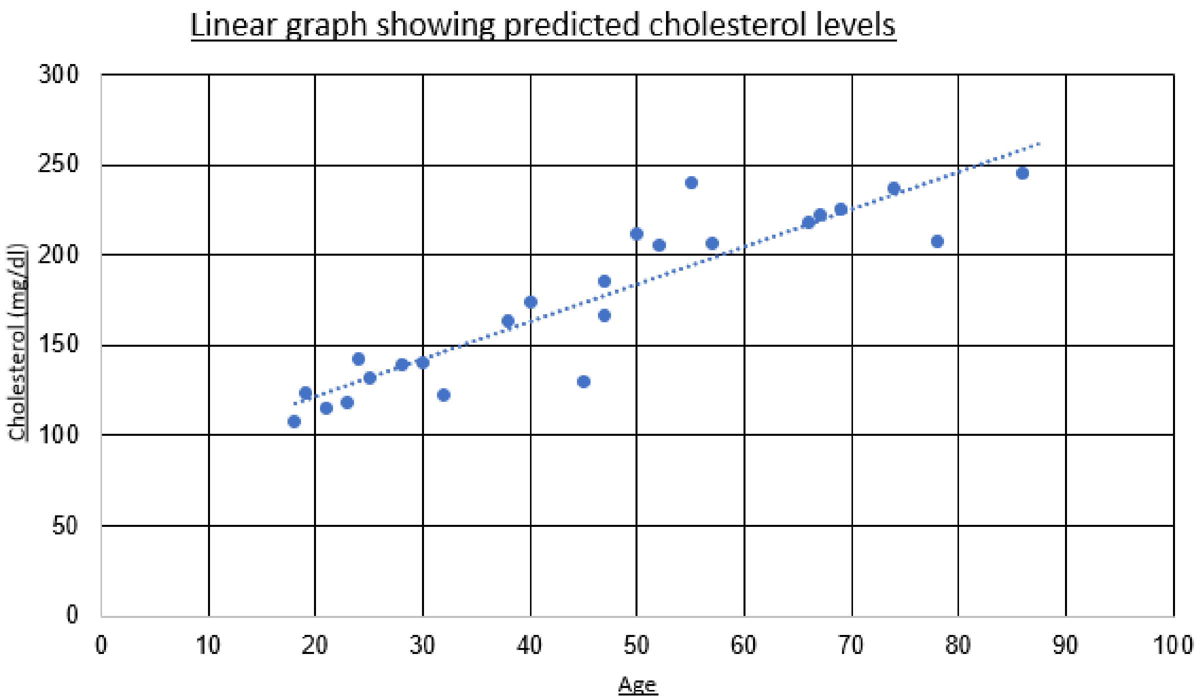
	Patient ID	Name	Cholestrol	Issued
0	3689	Mr. Abdul Weber	180.02	1999-08-24T15:48:33.327+10:00
1	3689	Mr. Abdul Weber	178.88	2002-09-10T15:48:33.327+10:00
2	3689	Mr. Abdul Weber	174.75	2005-09-27T15:48:33.327+10:00
3	3689	Mr. Abdul Weber	175.00	2008-10-14T16:48:33.327+11:00

By inputting this data into our model (regression) the machine will then learn the correlation between a patients age and their cholesterol and show this relationship on a graph.

By using the following r code:

```
#Load Train and Test datasets
#Identify feature and response variable(s) and values must be
numeric and numpy arrays
x_train <- input_variables_values_training_datasets
y_train <- target_variables_values_training_datasets
x_test <- input_variables_values_test_datasets
x <- cbind(x_train,y_train)
# Train the model using the training sets and check score
linear <- lm(y_train ~ ., data = x)
summary(linear)
#Predict Output
predicted= predict(linear,x_test)
```

We would have gotten the following graph:



This graph essentially shows that there is a positive correlation between age and cholesterol. This algorithm developed simply means that a patients age will determine what cholesterol level they are. In order to read and determine which patients have high cholesterol, refer to Figure 1 attached in references.

References

Despoudis, Fanis. 2017. “Understanding SOLID Principles: Open Closed Principle.” *CodeBurst.*

García, Daniel. 2018. “Paving the way towards high-level parallel pattern interfaces for data stream processing.” *Future Generation Computer Systems.*

Hodges, Jason. 2019. “ Design Patterns In Software Engineering from Scratch.” *Apress.*

Janssenmarch, Thorben. 2018. “SOLID Design Principles Explained: The Open/Closed Principle with Code Examples.” *Stackify.*

Perez, Severin. 2018. “Writing Flexible Code with the Single Responsibility Principle.” *Medium.*

Poyias, Andreas. 2019. “Design Patterns — A quick guide to Observer pattern.” *Medium.*

Cholesterol Guidelines

	Low Heart Disease Risk	Borderline Heart Disease Risk	High Heart Disease Risk
Total Cholesterol	Less than 200	200 - 239	240 and higher
LDL Cholesterol (the “bad” cholesterol)	Less than 130	130 - 159	160 and higher
HDL Cholesterol (the “good” cholesterol)	60 and higher	50 - 59	Less than 50
Triglycerides	Less than 150	150 - 199	200 and higher

Figure 1: Cholesterol Measurements