QARDIO

Test assignment.

There is a single screen app - to display ECG-chart and some ECG related statistics… and beautiful animation.

Now there are just AppDelegate, DataProvider, DataController, an xcdatamodel, and a dummy ViewController with several UI elements, some of which are not even bound to any data. One of the elements is ECG-chart which is being updated in real time and displays last 300 HR measurements.

There is a DataProvider which provides data at a hight rate (Heart Rate measurements, 500 measurements/sec). To receive it's updates you need to subscribe to it.

There is a Core Data data model with entities for average HR and average RHR. There is a DataController class to control the data model. Unfortunately, it's legacy code and written in Objective C. For some reasons you cannot migrate it on Swift, so you have to bridge it to the rest of the project.

Your tasks are:

- *To connect the DataProvider and the ViewController (You are not allowed to create another instances of DataProvider, and the only one should remain in the AppDelegate)*
- *To calculate and display an average Heart Rate (HR) in real time. For calculating, you should take the most recent 50000 measurements. You shouldn't display any average until you have enough data to calculate it.*
- *To calculate and display an average Resting Heart Rate (RHR) in real time. An HR measurement should only be considered as a RHR if it is less than 65. For calculating an average you should take the most recent 1000 RHR measurements. You shouldn't display any average until you have enough data to calculate it*
- *To calculate and display the most frequent HR value. Every 5 seconds enumerate through the recent 20000 HR values, pick the one value that appears most frequently, and display it.*
- *To simultaneously store a log of the average HR's and average RHR's in their respective Core Data entities. (Optionally it can be done in a separate background context)*
- *Optionally: implement session length realtime counter (time from the beginning of the session)*
- *Optionally: to cover the most critical logic with unit tests*

The solution should be efficient and properly organised in terms of multithreading. Keep an eye on clear architecture, classes' structure and responsibilities don't hesitate to change the code that is already in the project. For instance ViewController has already acquired some extra responsibilities and made a first small step towards Massive View Controller. Consider the project as not a prototype, but the app which is meant to be developed further.

ECG-chart shouldn't slow down or become choppy, the DP-rate (which is logged in DataProvider) should remain around the same value as well.

You are not allowed to use any third party libraries.