

# **iOS Application for Vital Signs Monitoring**

*Polyana Barboza - 1921164*

*Advisor: Prof. Carlos José Pereira de Lucena*

## **Motivation and Goal**

The motivation of the developed iOS application I will present is based on a recently discovered virus, COVID-19. Since the beginning of 2020, all the world is inside a pandemic brought by the new coronavirus. It is a super contagious virus, forcing us to a quarantine behavior to keep social distancing. This scenario emphasized public health care and my interest in eHealth, particularly in vital sign monitoring.

The idea behind the application is to allow healthcare professionals to monitor their patients wherever they are. It maximizes the monitoring of the patients' condition, allowing them to act based on the disease's evolution, and minimizes the contact between them and the patients, considering the ailment as super contagious. Thus, the proposed solution benefits both healthcare professionals and patients.

Therefore, my goal is to develop an iOS application for continuous vital signs monitoring, aided by Software Design Patterns, such as Model-View-Controller (MVC), Singleton, and Data Access Object (DAO), and by Software Agent.

## **Requirements**

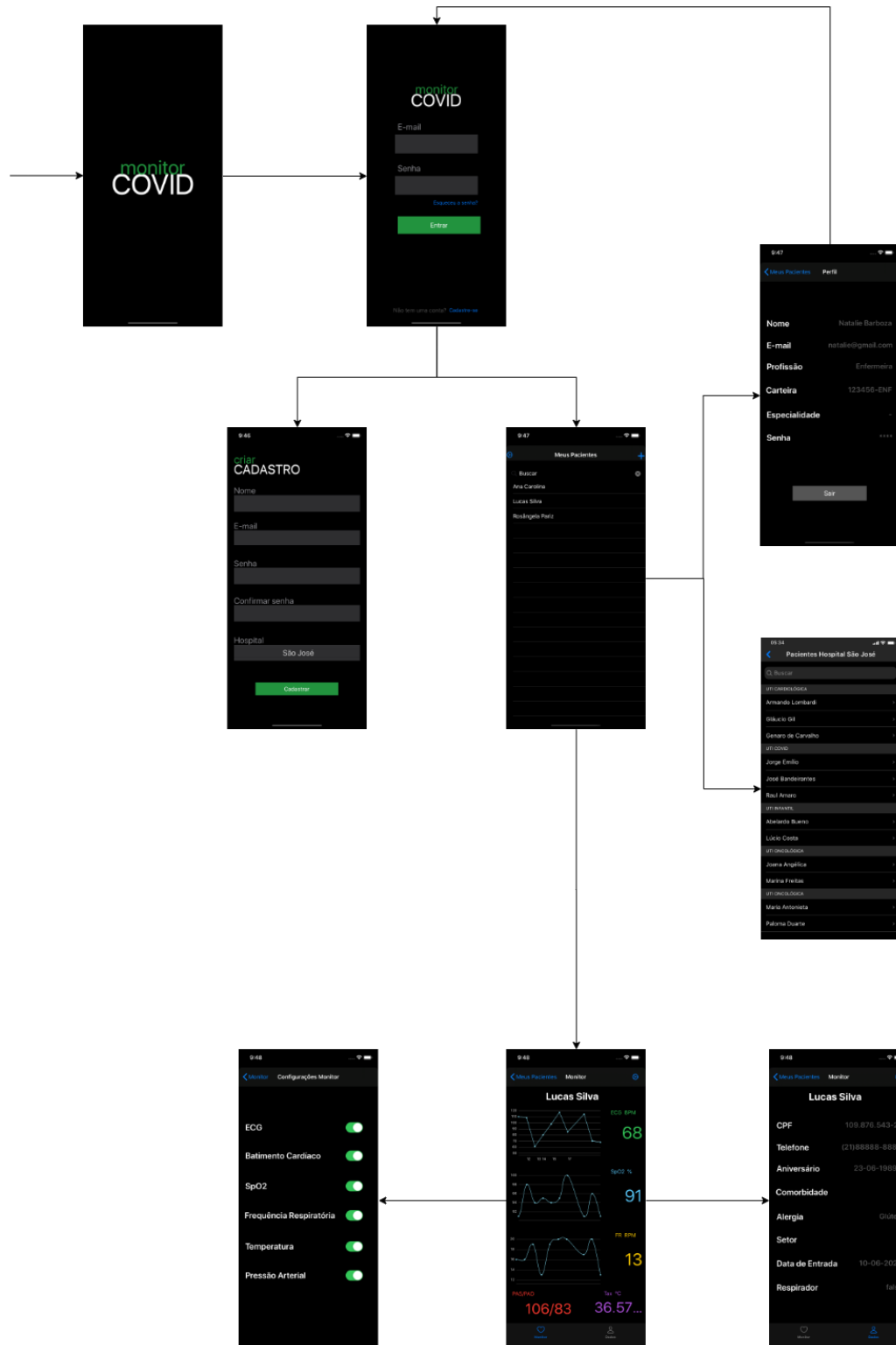
Basically, the application should receive data from patients' vital sign monitoring and show it to the final user. So, more detailed, it needs some functional requirements:

1. Users register, which should be health care workers;
2. Login;
3. Logout;
4. User information;
5. User's list of patients;
6. Choose patients to monitor;
7. Monitored patients;
8. Patients' details;
9. Monitor for each monitored patient;

10. Recover new vital sign data for all monitored patients through a software agent with cyclic behavior.

In (Figure 1), we can see a wireframe representing the flow through the application, containing the majority of these requirements explicitly.

Figure 1 - Wireframe flow through the application



## Architecture

Considering my interest in develop a Framework in iOS platform to suport health applications, I stated an initial framework structure for monitoring applications, so that the instance consumes it.

### *a) Initial Framework*

The initial framework is organized in hot spots and frozen spots. I definided five classes as hot spots:

- **HealthCareWorker**  
Class for healthcare worker objects, with their register information and related to their monitored patients.
- **Patient**  
Class for patients objects, with their basic information and related to their collected vital signs.
- **CollectorBehavior**  
Protocol to create a collector behavior to collect patients vital sign.
- **CyclicCollectorBehavior**  
Class to register a cyclic behavior inheriting from CollectorBehavior protocol. It contains functions to start and stop the cyclic collector and a provided collect function, that should be implemented depending on the data location.
- **DAOPatientVitalSign**  
Protocol to provide data access to patients vital sign (ideally from database), as a DAO, an architectural design pattern.

And as frozen spots, seven other classes:

- **Doctor**  
Class for doctors, inheriting properties from HealthCareWorker class added by two new properties for doctors' register.
- **Nurse**

Class for nurses, inheriting properties from HealthCareWorker class added by two new properties for nurses' register.

- Hospital

Class for hospital objects related to patients hospitalized in it and hospital sectors contained inside it.

- HospitalSector

Class to register hospital sectors, like UTI, CTI, newborn UTI, etc.

- VitalSignType

Class to register vital sign types that are being collected, like spo2, temperature, breath frequency, etc.

- PatientVitalSign

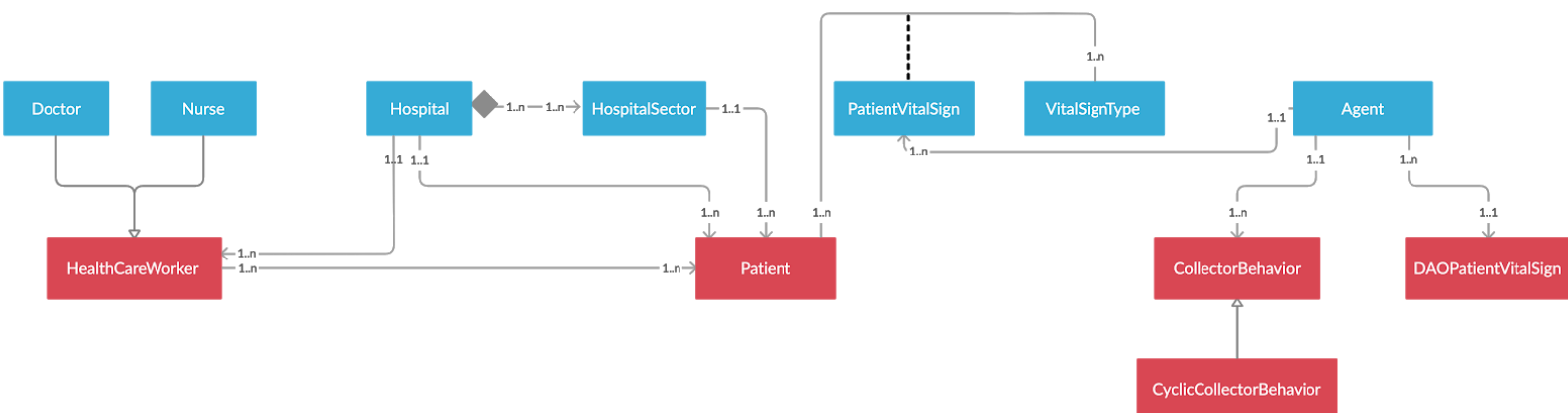
Class to register new vital signs collected from database.

- Agent

Class to create and start a software agent responsible for controlling new data collection, considering a cyclic behavior.

In (Figure 2), we can see the framework's class diagram.

Figure 2 - Framework's class diagram



*b) COVID instance*

My instance is organized as a MVC, an architectural design pattern that divides the application logic into three interconnected elements, Model, Controller, and View.

The Model element contains three instance classes that inherit from the framework classes and other five instance classes:

- COVID19CollectorBehavior

Instance class for collector behavior, inheriting from the Framework class CyclicCollectorBehavior. In this class, I implemented a function to collect new vital signs for each patient being monitored by health care worker logged in the app and to update monitor charts, adding new entries.

- COVID19Patient

Instance class Patient, inheriting from the Framework class Patient added by a new property to control if patient is using a breather.

- COVID19UsefulData

Class to model a Singleton, a creational design pattern to provide a single point of access to useful data.

- COVID19Monitor

Class to save the monitor layout for each patient.

- COVID19DAOIdentifyHCW

Class structuring a DAO to access database and collect healthcare worker register information after the login.

- COVID19DAOIdentifyHospital

Class structuring a DAO to access database and identify the healthcare worker hospital after the login, in order to identify patients that the user can monitor.

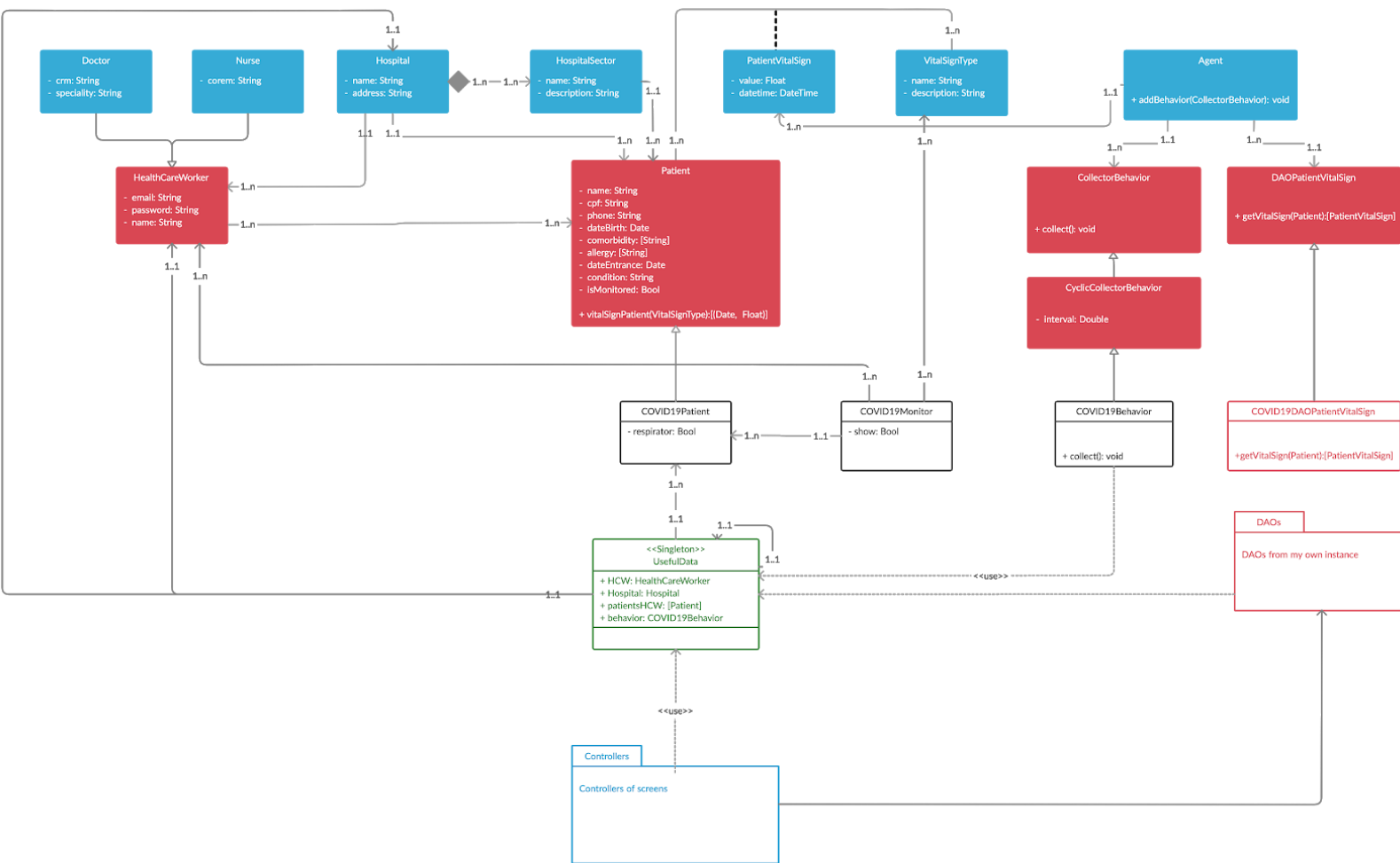
- COVID19DAOGetPatients

Class structuring a DAO to access database and identify patients already monitored by the healthcare worker after the login.

- COVID19DAOPatientVitalSign



Figure 4 - Complete instance's class diagram



## Tests

Some manually tests were conducted, based on the user's flow inside the app and considering it's main functions. These tests evaluate the system's response to user's actions. The tests were focused in login and in the application core, patients information and patients' vital sign monitor.

### Table 1 - Login tests

TC_Login		
Description/Goal	This test aims to test the sign in, when user insert login information	
Condition	Tests three possible user actions: empty fields; any wrong information; right information	
Steps	User	Sistema
1	Tries to login with empty fields	Pops-ups an error message
2	Tries to login with wrong information	Pops-ups an error message

<b>3</b>	Tries to login with right information	Goes into the application
<b>Post-condition</b>	Expected results	

Table 2 - Patient tests

<b>TC_Patient</b>		
<b>Description/Goal</b>	This test aims to test if when user selects a patient, the system shows the right information	
<b>Condition</b>	Tests the selection of one patient in each time and the access to their information	
<b>Steps</b>	<b>User</b>	<b>System</b>
<b>1</b>	Selects a patient	Goes into the patient monitor screen
<b>2</b>	Select the patient data in tab bar	Goes into the patient's general information
<b>3</b>	Back to patients list and select other patient	Goes into the monitor screen with new patients' vital sign
<b>Post-condition</b>	Expected results	

Table 3 - Monitor tests

<b>TC_Monitor</b>		
<b>Description/Goal</b>	This test aims to test if the monitor still collects new vital signs even if the user is not in the monitor screen	
<b>Condition</b>	Tests the selection of a patient to see the monitor, navigation through others screens and backing to the monitor	
<b>Steps</b>	<b>User</b>	<b>System</b>
<b>1</b>	Selects a patient and look at the monitor screen untill the next vital sign update (CyclicBehavior)	Graph evolve with the new vital sign inserted in each graph
<b>2</b>	Navigate through other screens and back to monitor	Shows up new entries, collected while user was not in screen
<b>Post-condition</b>	Expected results	



## **References**

Fernandes CO, Lucena CJP. A Software Framework for Remote Patient Monitoring by Using Multi-Agent Systems Support. JMIR Med Inform 2017;5(1):e9.

Jade.tilab. JAVA Agent Development Framework: an open source platform for peer-to-peer agent based applications URL: <http://jade.tilab.com/>

Markiewicz M, de Lucena CJ. Object oriented framework development. Crossroads 2001 Jul 01;7(4):3-9

Oh H, Rizo C, Enkin M, Jadad A. What Is eHealth: A Systematic Review of Published Definitions. J Med Internet Res 2005;7(1):e1

## **GitHub Repository**

<https://github.com/PolyanaSRB/vital-sign-monitor-ios>