*Florida International University*

*School of Computing and Information Sciences*

CIS 4911 - Senior Capstone Project

Software Engineering Focus

Final Deliverable

RMCuff

Team #10

**Team Members**

David Baez

Marc Roger

**Product Owner**: Peter Dickson

**Instructor**: Masoud Sadjadi

*This document is not to be copied or distributed without the written consent of the Product Owner, Peter Dickson, whose contact information is available below. Only proceed reading if you agree not to reproduce, duplicate, copy, sell, resell or exploit for any purpose, any portion of the document.*

Peter Dickson

Product Owner

[dicksonp@fiu.edu](mailto:dicksonp@fiu.edu)

David Baez

Software Developer

[dbaez011@fiu.edu](mailto:dbaez011@fiu.edu)

Marc Roger

Software Developer

[mroge009@fiu.edu](mailto:mroge009@fiu.edu)

***Abstract***

*A new system for joint effort tracking of blood pressure between a patient, a primary caregiver and secondary caregivers has been constructed in software and outlined in detail in the writing that follows. The mobility of cellular devices, Android devices in this case, is leveraged to provide reliable remote monitoring of patient blood pressure activity. The software works in combination with a bluetooth operated blood pressure cuff that is currently in development.*

**Table of Contents**

**Introduction**

Current System

Purpose of New System

**User Stories**

Implemented User Stories

Pending User Stories

**Project Plan**

Hardware and Software Resources

Sprints Plan

*Sprint 1*

**System Design**

Architectural Patterns

System and Subsystem Decomposition

Deployment Diagram

Design Patterns

**System Validation**

**Glossary**

**Appendix**

Appendix A - UML Diagrams

*Static UML Diagrams*

*Dynamic UML Diagrams*

Appendix B - User Interface Design

Appendix C - Sprint Review Reports

Appendix D - Sprint Retrospective Reports

**References**

# Introduction

In the introduction that follows, the system will be dissected in depth in terms of what it is like prior to a software solution or the intended software solution and what the system will become after the intended software solution is set in place. The section is the first phase of planning required to implement a new system. This section will describe the current system in place and the new system to be built in detail. In addition, this section will enumerate the steps, in units of User Stories, required to achieve the intended system.

## Current System

Since the advent of wearable technology, health applications have become extremely popular and have flooded mobile device application stores. Most of these health applications are geared around tracking fitness statistics such as distance traveled and calories burned for adults and young adults. The feature complexity and user interfaces of many of these applications are geared towards a younger audience who grew up with and are savvy of, technology. Applications for tracking blood pressure in older adults such as iHealth exist, but they fail to remain minimal and simple enough for the people who desperately need the service, the elderly, to use easily. They also lack functionality for the people who care after the blood pressure cuff user to track results and set blood pressure schedules. As an example, iHealth keeps track of blood pressure, weight, calories consumed and burned, sleep, and so on. It provides diary and goal tracking features. The excess of functionality leads to use complexity and user interface complexity. Leaving elderly users with nowhere to turn. The features are meant for self tracking, which isn’t very ideal for elderly individuals who lack the self motivation required for the mundane task of tracking blood pressure, and require the help of a related primary caregiver to stay on top of things.

In the current system, a primary caregiver takes the responsibility of tracking the blood pressure of a patient, usually a loved one, over the phone with a few occasional personal visits. Typically they would tell the patient to take their blood pressure at a certain time, and just have to take their word for it. The patient and caregiver largely communicate over the phone, and a patient is able to lie about reading outcomes, to longer support negative habits. It requires a lot of responsibility on the patient side and a lot of trust on the primary caregiver side.

## Purpose of New System

The new system has one primary purpose, and that is accessibility for elderly, less tech savvy adults. The big difference, is that this new system gives the primary caregiver the power to schedule and track readings to motivate the patient remotely via their cell phone. It is a minimized application that provides blood pressure monitoring and only features directly related to that, thus avoiding overcomplexity. Less functionality also means a clean user interface, so users could easily and quickly navigate the application with little to no instruction. The new system will send the patient’s primary caregiver blood pressure reports via a push notification to a separate primary caregiver application. The primary caregiver can then send secondary caregivers text message logs of the patient’s blood pressure. Text messages make the transfer of information immediate and more likely to be read quickly. No need for secondary caregivers to download an app, and no email reports that rarely get read, just simplicity. The point is to do one thing very well.

# User Stories

In the user stories section that follows, a comprehensive list of all user stories gathered from the product owner will be laid out and described. The user stories will be separated in terms of which will be implemented in this system and which are pending for potential future evolutions of the system.

## Implemented User Stories

**User Story # 666 - Text Notifications to Secondary Caregivers (Required)**

As a Primary Caregiver, I want to automatically or manually send text notifications to secondary caregivers so that I can alert other family members or professionals about the patient’s status.

**User Story # 669 - Store Editable Secondary Caregiver Data (Required)**

As a Primary Caregiver, I want a stored database (I can edit) of secondary caregiver’s names, phone numbers, and a checkbox of whether I want to send them notifications. So that I can easily manage the people I am sending text notifications to.

**User Story # 673 - Pull Patient new reading data as Primary Caregiver (Required)**

As a Primary Caregiver, I want to be alerted when the Patient completes a reading (via push notification). So that I can be certain the reading was taken.

**User Story # 675 - Push schedule to Patient (Required)**

As a Primary Caregiver, I want to be able to alert the patient (via push notification) when I create a new scheduled reading. So that they receive a push notification and take their blood pressure reading on time.

**User Story # 676 - Push new reading data to Primary Caregiver (Required)**

As a Patient, I want to notify a Primary Caregiver (via push notification) when I take a blood pressure reading. So that they know I’ve taken my reading.

**User Story # 677 - Pull Schedule Update as Patient (Required)**

As a Patient, I’d like to be alerted (via push notification) when my Primary Caregiver schedules a blood pressure reading. So that I know when to prepare for the reading.

**User Story # 674 - Store past reading data on Primary Caregiver side (Required)**

As a Primary Caregiver, I want to be able to see data from past readings so that I can track the Patient’s progress.

**User Story # 670 - Schedule Future Reading (Required)**

As a Primary Caregiver, I’d like to schedule readings for the patient and keep a list of upcoming scheduled readings. So that I have control over when the patient takes their reading.

**User Story #686 - Shortcut for Phone Calls (Optional)**

As a user, I want a shortcut button so that I can call the Patient / Primary Caregiver without manually exiting the application. So that, I can more easily connect to the opposite party.

**User Story # 679 - Store reading Schedule on Patient Side (Required)**

As a Patient, I’d like to keep a list of upcoming scheduled readings. So that I know ahead of time when my readings will take place.

**User Story # 671 - Store Past Readings Data on Patient Side (Required)**

As a Patient, I want to be able to see data from past readings so that I can track my progress.

**User Story # 678 - Alert patient of reading time (Required)**

As a Patient, I’d like to be alerted (via push notification) when it is time for a scheduled reading., so that I don’t miss the reading.

**User Story # 667 - Send Bluetooth Data (Required)**

As a Patient, I want to be able to activate my blood pressure cuff from my smartphone app (send Bluetooth data) So that I can begin my blood pressure test.

**User Story # 668 - Receive Bluetooth Data (Required)**

As a Patient, I want to be able to receive data (via Bluetooth) from my blood pressure cuff to my smartphone app. So that I can receive my blood pressure reading.

**User Story # 697 - Register as a user (Required)**

As a user, I want to be able to register on my first use (using my phone number as a unique identifier) so that I can receive/send push notifications.

**User Story # 706 - Start Bluetooth Reading (Required)**

As a patient, we want our device to connect to the Bluetooth device and initiate a reading so that we can measure our blood pressure.

**User Story # 707 - Receive Reading Data via Bluetooth (Required)**

As a patient, we would like to receive the reading data from the Bluetooth device, once the reading is complete. So that we can have that information stored on our phone and sent to our caregiver.

**User Story # 708 - Clean interface and functionality (Required)**

As a user, I want to navigate a clean interface with polished functionality, so that I can focus on my blood pressure (patient) or tracking someone’s blood pressure (caregiver).

## Pending User Stories

**User Story # 716 - Verify phone numbers via text message (Could)**

As a user and a caregiver I want to be notified through a text message that the app has the correct number in the system, so that I do not have to worry about any issues sending readings (patient) or receiving someone’s readings (caregiver).

**User Story # 717 - Store Online Database of blood pressure data for scientific study and research (Could)**

As the system, I would like to be able to store and online database of blood pressure data for scientific study and research.

**User Story # 718 - Show data in graphical form for patient and caregiver (Could)**

As the system I would like to be able to use all past blood pressure readings on the app to display them in a very easy to read graphical form should the user require.

**User Story # 719 - Update User Interface (Could)**

As a developer I want to be able to update the user interface to be able to make it easier and simpler to use for the users.

# Project Plan

In the project plan that follows, the organizational aspects that took place prior to attempting each sprint and the project in general will be described. This section will list and explain the use of hardware and software resources being used to develop the system. In addition, this section will outline the planning of each sprint and the user stories pertaining to said sprints. The user stories will be described in depth by tasks, acceptance criteria, and modeling.

## Hardware and Software Resources

The software resources that will be used to develop the system from beginning to end are:

* Android Studio - Android’s official integrated development environment (IDE)
* Java JDK - Includes tools for developing, debugging, and monitoring Java applications
* Android SDK - Includes development tools, an emulator, and required libraries to build Android applications
* Java - Programming language used in native Android applications
* XML - Used for layout and UI of Android applications
* PushBots - A push notification service used to send implemented to send push notifications between the 2 different apps and have them wake automatically.
* GitHub - Online repository that will store our project code
* Mingle - Used for planning scrum sprints

The hardware resources that will be used to develop the system from beginning to end are:

* Any computer that meets the specifications required by Android Studio
* Any Android device to test the application on
* A custom Bluetooth blood pressure cuff to connect to

## Sprints Plan

### Sprint 1

(08/31/2015 - 09/11/2015)

**User Story # 666 - Text Notifications to Secondary Caregivers**

***Tasks***

* Add UI elements needed for user to interact with
* Code function that sends a text message to a single phone number
* Expand function to support sending to an array of phone numbers

***Acceptance Criteria***

* Text notifications can be sent manually
* Text notification include the patient’s name and a log of their recent past blood pressure readings
* Can message all secondary caregivers listed by primary caregiver at the same time

***Modeling***

A more primitive version of the UML Class Diagram in Appendix A was created to display this feature.

### Sprint 2

(09/14/2015 - 09/25/2015)

**User Story # 669 - Store Editable Secondary Caregiver Data**

***Tasks***

* Set up Secondary Caregiver UI Page
* Set up device storage/retrieval code
* Show data in list
* Edit/Delete functionality

***Acceptance Criteria***

* Primary Caregiver can store caregiver info
* Primary Caregiver can edit/delete caregiver info
* Secondary Caregiver info is stored locally on Device

***Modeling***

A more primitive version of the UML Class Diagram in Appendix A was created to display this feature.

### Sprint 3

(09/28/2015 - 10/09/2015)

**User Story # 673 - Pull Patient new reading data as Primary Caregiver**

***Tasks***

* Set up push notification system on primary caregiver side
* Set up automated wake on push receipt
* Display Received reading data

***Acceptance Criteria***

* When the patient completes a reading the Primary Caregiver receives a push notification
* When the Primary Caregiver opens the notification, they see the reading data
* The push notification should be sent automatically upon reading completion

***Modeling***

The current UML Class Diagram in Appendix A was created to display this feature.

**User Story # 675 - Push schedule to Patient**

***Tasks***

* Set up push notification system on caregiver side
* Allow user input for schedule
* HTTP Post request to server

***Acceptance Criteria***

* The patient receives a push notification automatically when I add a new scheduled reading
* When the patient opens the notification, they will be shown the scheduled time
* The patient’s phone will automatically alert them when the scheduled time has arrived

***Modeling***

The current UML Class Diagram in Appendix A was created to display this feature.

**User Story # 676 - Push new reading data to Primary Caregiver**

***Tasks***

* Set up push notification system on patient side
* Simulate Bluetooth reading for data to submit
* HTTP Post request to server

***Acceptance Criteria***

* The Primary caregiver is automatically notified when a reading has been completed by the patient
* When the primary caregiver opens the notification, they will see the new reading data
* The reading data is stored to the local list of past reading data on the primary caregivers phone

***Modeling***

The current UML Class Diagram in Appendix A was created to display this feature.

**User Story # 677 - Pull Schedule Update as Patient**

***Tasks***

* Set up push notification system on patient side
* Set up automated wake on push receipt
* Display Received schedule data

***Acceptance Criteria***

* When the Primary caregiver sets a new schedule, the patient phone will pull that new data
* The patient phone will alert the patient via push notification
* When the patient opens the notification, they will see the new schedule data

***Modeling***

The current UML Class Diagram in Appendix A was created to display this feature.

### Sprint 4

(10/12/2015 - 10/23/2015)

**User Story # 674 - Store past reading data on Primary Caregiver side**

***Tasks***

* Grab data from device memory
* Set up UI page
* Display device data in UI

***Acceptance Criteria***

* Data is queried from a database
* App must store the last 10 readings
* Must store the time and date of each reading as well

***Modeling***

The current UML Class Diagram in Appendix A was modified to properly account for this feature.

**User Story # 670 - Schedule Future Reading**

***Tasks***

* Set up UI for user to input schedule
* Grab input data and store to device
* Send the input data to the Patient

***Acceptance Criteria***

* Must be able to schedule the time and date of the next reading
* Can choose the alert type (vibration, alert, etc)
* Upload to a database so that the patient application can query the schedule

***Modeling***

The current UML Class Diagram in Appendix A was modified to properly account for this feature.

**User Story #686 - Shortcut for Phone Calls**

***Tasks***

* Phone call shortcut (caregiver side)
* Phone call shortcut (patient side)

***Acceptance Criteria***

* The shortcut button is shown on the home screen
* It successfully transfers you over the phone application
* You are able to more easily connect to the opposite party

### Sprint 5

(10/26/2015 - 11/06/2015)

**User Story # 679 - Store reading Schedule on Patient Side**

***Tasks***

* Retrieve data properly
* Store schedule data in a united way

***Acceptance Criteria***

* The readings schedule should be stored locally on the patient phone
* All upcoming readings must be displayed
* Must query database to check for updates

***Modeling***

The current UML Class Diagram in Appendix A was modified to properly account for this feature.

**User Story # 671 - Store Past Readings Data on Patient Side**

***Tasks***

* Retrieve data properly
* Store Readings Data in a united way

***Acceptance Criteria***

* App must store the last 10 readings
* Must store the time and date of each reading as well
* Data can be uploaded to a database where secondary caregiver can query

***Modeling***

The current UML Class Diagram in Appendix A was modified to properly account for this feature.

**User Story # 678 - Alert patient of reading time**

***Tasks***

* Create Calendar event from received schedule
* Handle scheduling on receipt of push notification

***Acceptance Criteria***

* When it is finally time for a scheduled reading to take place (set in the past), the patient must be alerted
* Must work when app is in background or not open
* Must ring and send a local push notification

### Sprint 6

(11/09/2015 - 11/20/2015)

**User Story # 667 - Send Bluetooth Data**

***Tasks***

* Build Sample Bluetooth Device
* Connect to Bluetooth device and send data

***Acceptance Criteria***

* The blood pressure cuff can be connected to via Bluetooth
* The blood pressure cuff can be activated to begin a blood pressure reading
* The Bluetooth connection and activation work to a high degree

***Modeling***

The current UML Class Diagram in Appendix A was modified to properly account for this feature.

**User Story # 668 - Receive Bluetooth Data**

***Tasks***

* Build Sample Bluetooth Device
* Connect to Bluetooth device and receive data

***Acceptance Criteria***

* The application is able to receive Bluetooth connection data to confirm connection with proper device
* The application receives systolic and diastolic readings from blood pressure cuffs
* Must be tested to make sure the readings are accurate

***Modeling***

The current UML Class Diagram in Appendix A was modified to properly account for this feature.

**User Story # 697 - Register as a user**

***Tasks***

* Save Registration info persistently
* Apply registration form to Patient and PCG Apps
* Check registration input
* Create Registration Form Page

***Acceptance Criteria***

* Can register name and phone number
* Can register second party’s name and phone number
* Happens on first use

### Sprint 7

(11/23/2015 - 12/04/2015)

**User Story # 706 - Start Bluetooth Reading**

***Tasks***

* Communicate with custom Bluetooth cuff

***Acceptance Criteria***

* The blood pressure cuff can be connected to via Bluetooth
* Data could be sent to the Bluetooth device

***Modeling***

The current UML Class Diagram in Appendix A was modified to properly account for this feature.

**User Story # 707 - Receive Reading Data via Bluetooth**

***Tasks***

* Incoming Bluetooth communication

***Acceptance Criteria***

* The blood pressure cuff can be connected to via Bluetooth
* Data could be properly received from the Bluetooth device

***Modeling***

The current UML Class Diagram in Appendix A was modified to properly account for this feature.

**User Story # 708 - Clean interface and functionality**

***Tasks***

* Smooth out functionality
* Update User Interface
* Create app Icons
* Create Registration Splash screens

***Acceptance Criteria***

* Application runs without crashing
* User interface could be navigated easily
* App icons and registration screen help tell the apps apart

# System Design

In the system design section that follows, the structural design, or structural blueprint, of the system will described textually and some portions drawn out using UML diagrams. This section will identify and explain the architectural patterns used. The major subsystems will be illustrated using a UML package diagram. The connection between subsystems and hardware will be illustrated using a UML deployment diagram. In addition, this section will identify and explain the design patterns used.

## Architectural Patterns

Model View Controller (MVC) Architecture - We make use of this pattern to separate user interface code from data access code, making it easier to modify one without affecting the other.

Client/Server Architecture - We implemented a custom php web service hosted on our web server, which we communicate with via the app. This web service handles the sending of push notifications. One app sends data up to the server; the server takes that data and sends it to the specified user.

## System and Subsystem Decomposition

The subsystems in Figure 1 (below) are listed and described in detail:

* Patient Information/Caregiver Information Subsystems - This subsystem consists of all the current live data attached to the user object and the current state of the user object (Patient object or Caregiver object depending on the application used)
* User Interface Subsystem - This is the system the user interacts with through graphically displayed data. This interaction is how the user object could be altered.
* Persistent Data Subsystem - This subsystem is in charge of storing data to the android device. It is accessed by the user object subsystem (Patient or Caregiver) and the Push/Pull subsystem when new data comes in or needs to go out.
* Bluetooth Device Subsystem - This subsystem consists of the Bluetooth blood pressure cuff, which can send/receive data between the patient application.

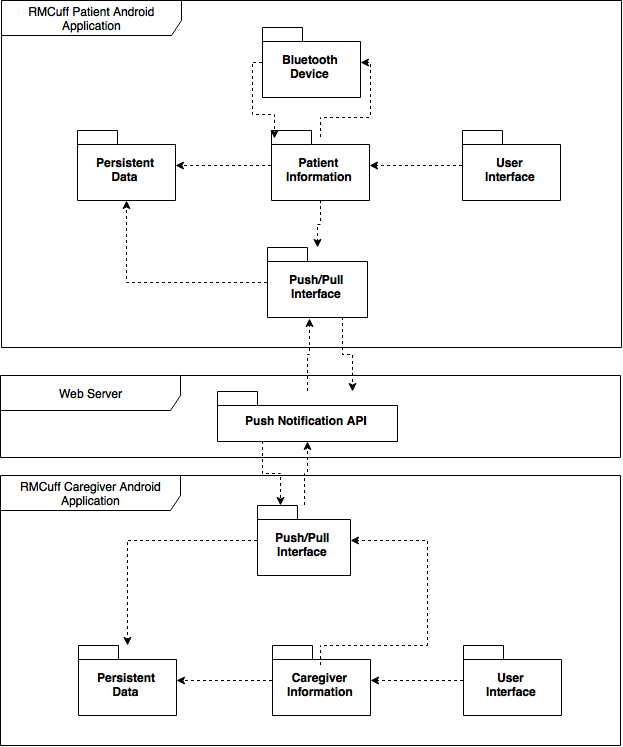


Figure 1 - UML Package Diagram

## Deployment Diagram

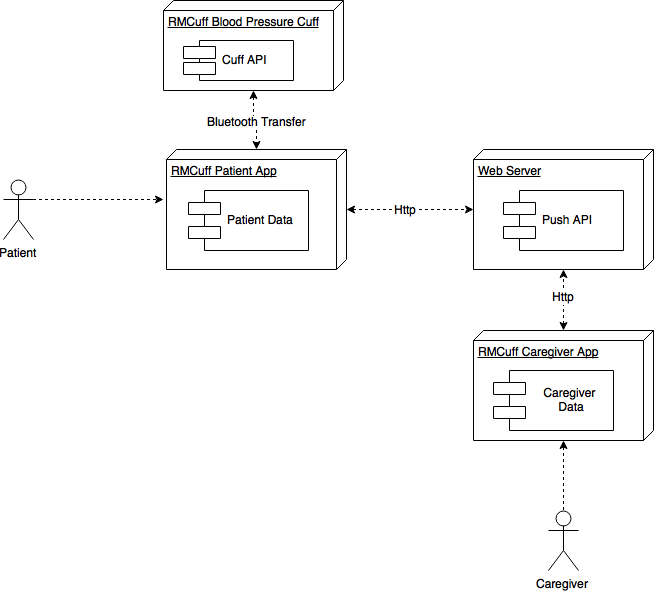


Figure 2 - UML Deployment Diagram

## Design Patterns

**Singleton Pattern** - Our applications only support one user to be registered at any instant in time. Having more than one user would result in an error of functionality, so we limit the possibility of creating more than one Patient or Caregiver object at a time. A singleton is the perfect way to achieve this.

# System Validation

**Unit test cases -** Throughout the project we were able to produce a handful of unit test cases in order to test certain elements of the application. These included tests for Bluetooth, SMS, and modifying caregivers. The validation of these cases was easy for some and complicated for others due to the nature of validating on the android API.

**Automated tests -** In addition to the few unit test cases, we were also able to generate an automated test script that ran off an android emulator. The use of the android emulator made testing very complicated for many reasons including the hardware running very slow and even crashing at most time when running within the emulator.

With the tests that we added, we found later that it was much easier to manually test the application each time we made changes. The manual tests included:

* SMS
* Bluetooth
* Phone call
* Activity management
* Adding, deleting, editing caregiver
* Saving of the caregiver, reading, and settings information

Manual tests proved to be much more feasible and allowed us to complete our work much faster than using automated or even gated checks. Due to the nature of the android application was the only reason this was viable otherwise normal testing methods would of been faster.

# Glossary

* **Android Studio -** Android’s official integrated development environment (IDE)
* **Blood Pressure Reports (Text Reports) -** Blood Pressure Reports (or simply Text Reports) refer to the logs of past blood pressure readings a primary caregiver can send out to secondary caregivers via text message.
* **Caregiver** - *See Primary Caregiver definition*
* **Caregiver** **Application** **-** We have built two applications, one is specifically for caregivers.
* **Cuff -** Within the scope of this project, a cuff will refer to a blood pressure reading device that is wraps around the wrist.
* **Diastolic** **Blood** **Pressure** **-** The diastolic blood pressure number or the bottom number indicates the pressure in the arteries when the heart rests between beats. A normal diastolic blood pressure number is less than 80. A diastolic blood pressure between 80 and 89 indicates prehypertension.
* **Patient** **-** Within the scope of this project, a patient is a user who suffers from bad blood pressure and is using our application to track it.
* **Patient** **Application** **-** We have built two applications, one is specifically for patients.
* **Primary** **Caregiver (Caregiver) -** Within the scope of this project, a primary caregiver (or just plain caregiver) is a user is a helping a loved one that suffers from bad blood pressure (patient) using our application.
* **Pushbots** **-** The Push Notification API used to implement our own custom push notification API
* **Reading -** Used as shorthand for blood pressure reading. A patient takes readings on their patient application and sends them over to the primary caregiver.
* **Schedule -** Within the scope of this project, a schedule is a date and time that is set by the primary caregiver and sent over to the patient. The schedule lets the patient know when they should take their blood pressure reading. A patient will be alerted when it is time for a blood pressure reading.
* **Secondary** **Caregiver -** This includes any other caregiver who is not the primary caregiver; this group might consist of family members, friends, or physicians. They do not have an application for tracking the patient’s blood pressure, but the primary caregiver could choose to send them text reports containing blood pressure data over time.

**Systolic** **Blood** **Pressure** **-** When your heart beats, it contracts and pushes blood through the arteries to the rest of your body. This force creates pressure on the arteries. This is called systolic blood pressure. A normal systolic blood pressure is 120 or below.

# Appendix

## Appendix A - UML Diagrams

### Static UML Diagrams

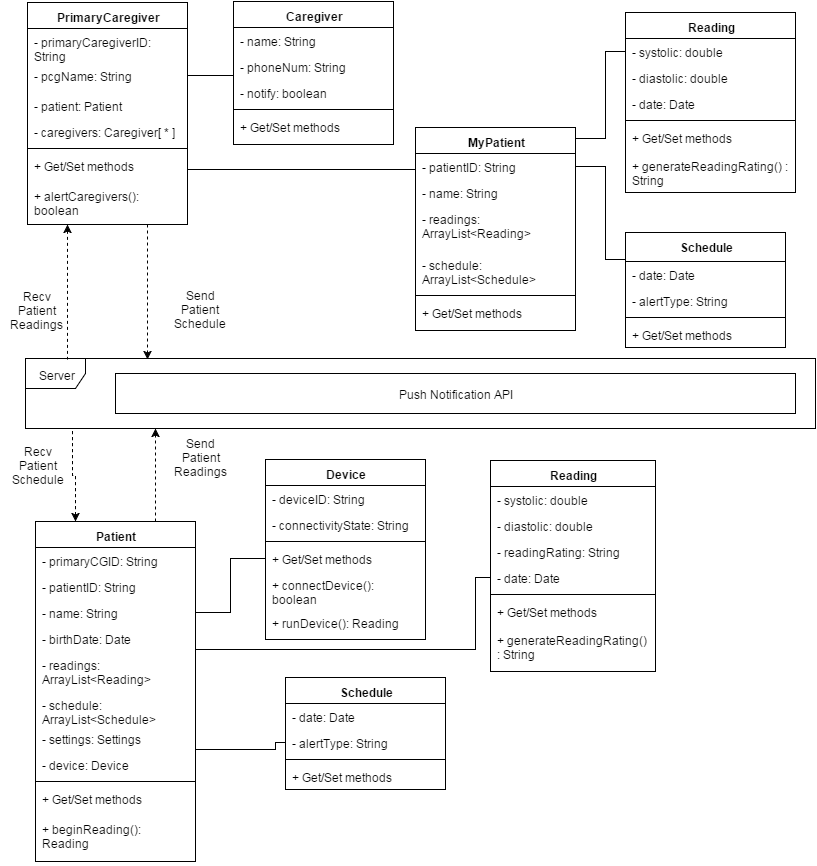


Figure 3 - UML Class Diagram

### Dynamic UML Diagrams

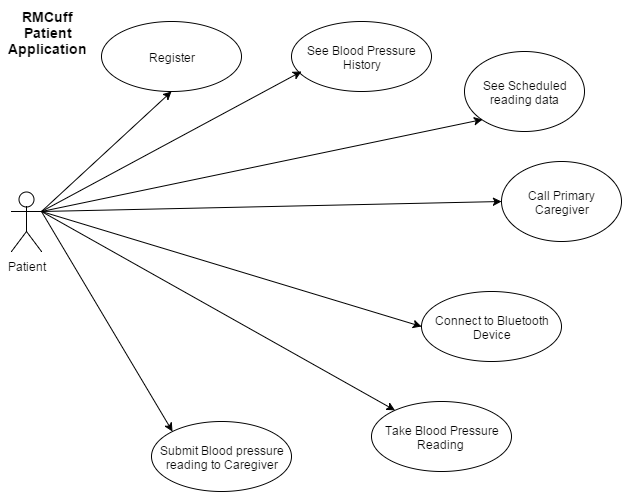


Figure 4 - UML Use Case Diagram (Patient)

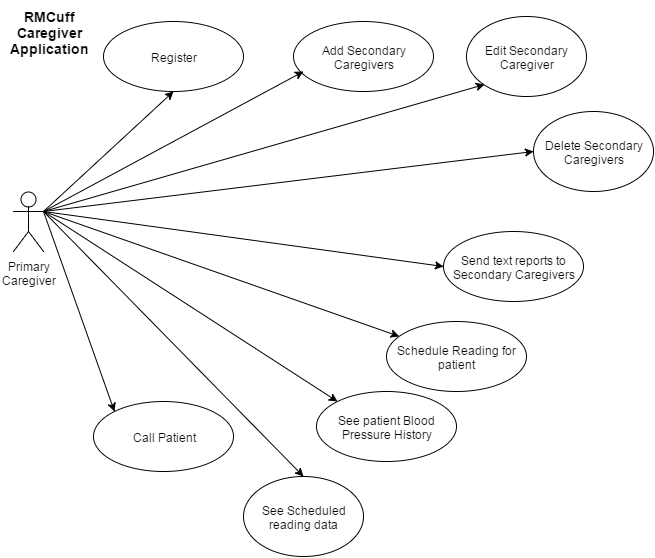


Figure 5 - UML Use Case Diagram (Caregiver)

## Appendix B - User Interface Design

## 

Figure 6 - Registration Splash Screen UI

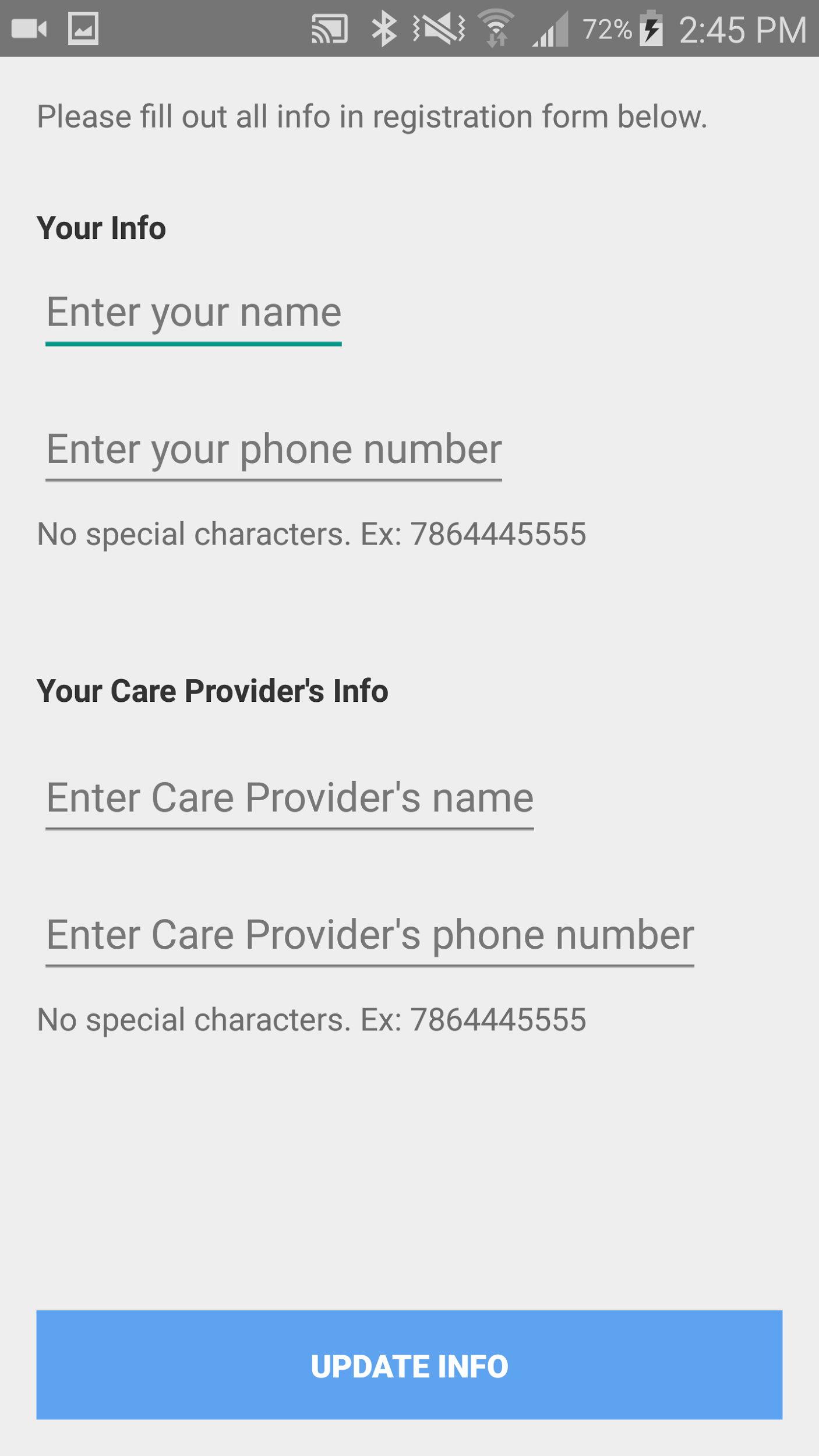
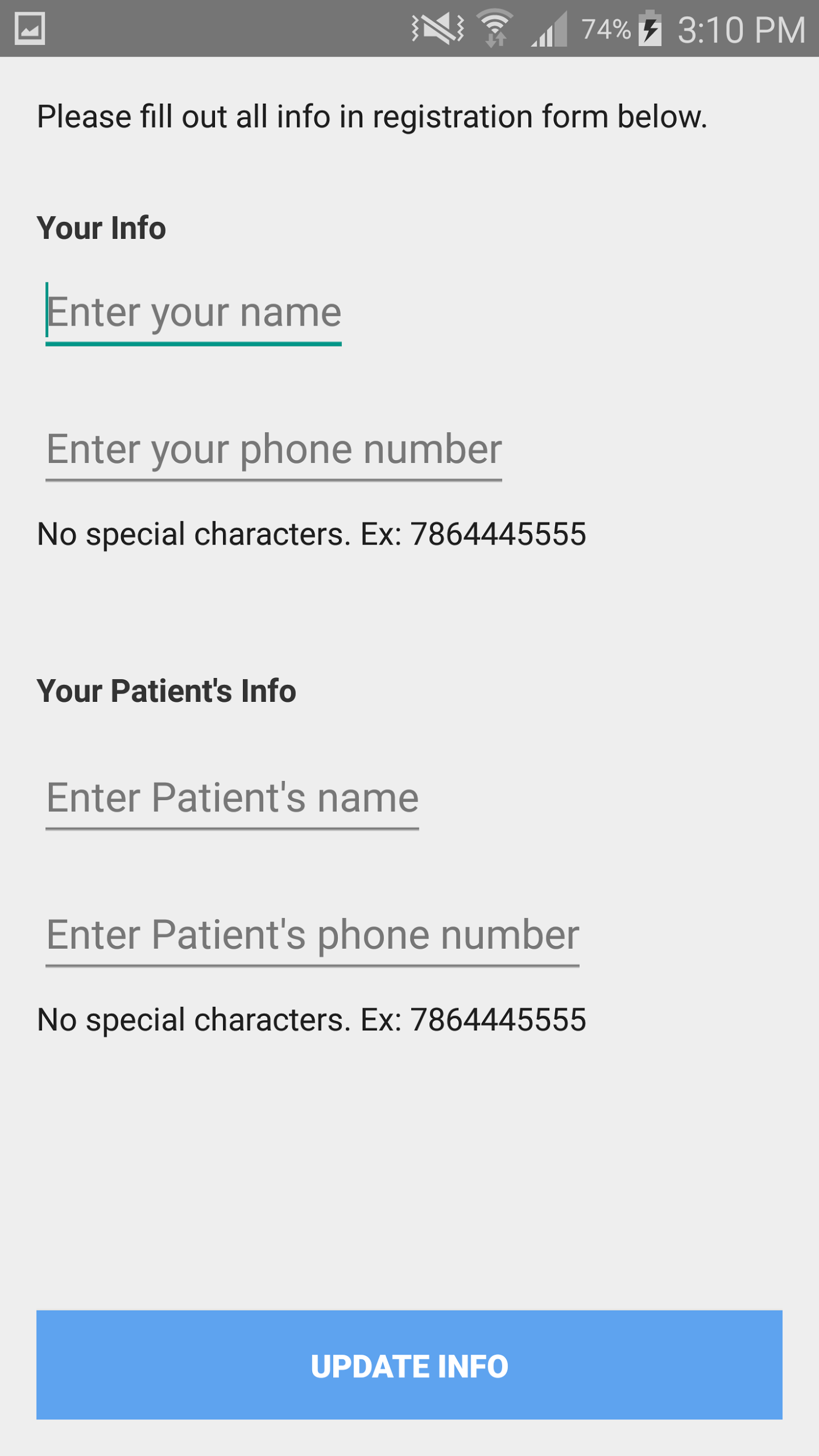


Figure 7 - Registration Page UI

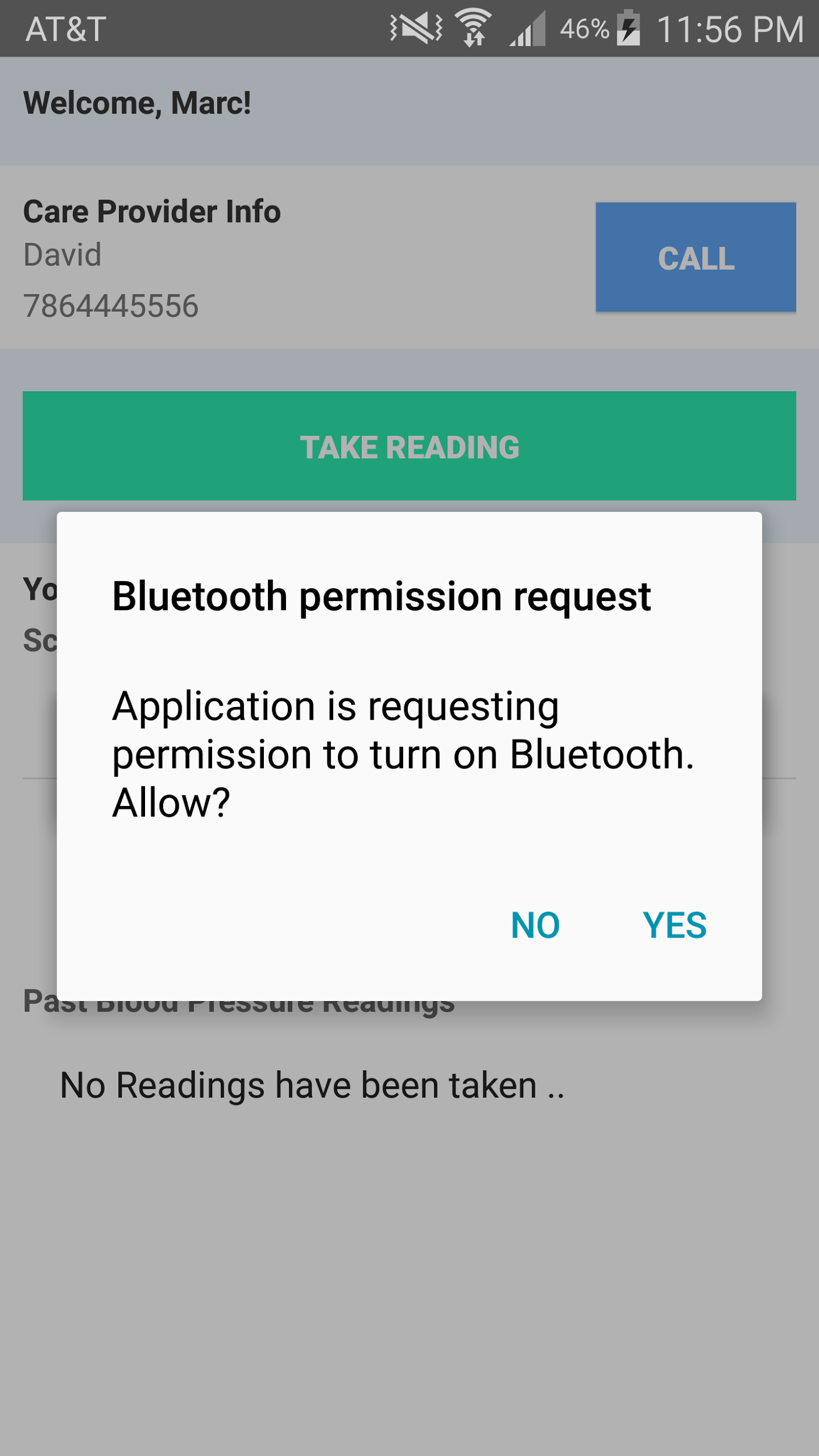
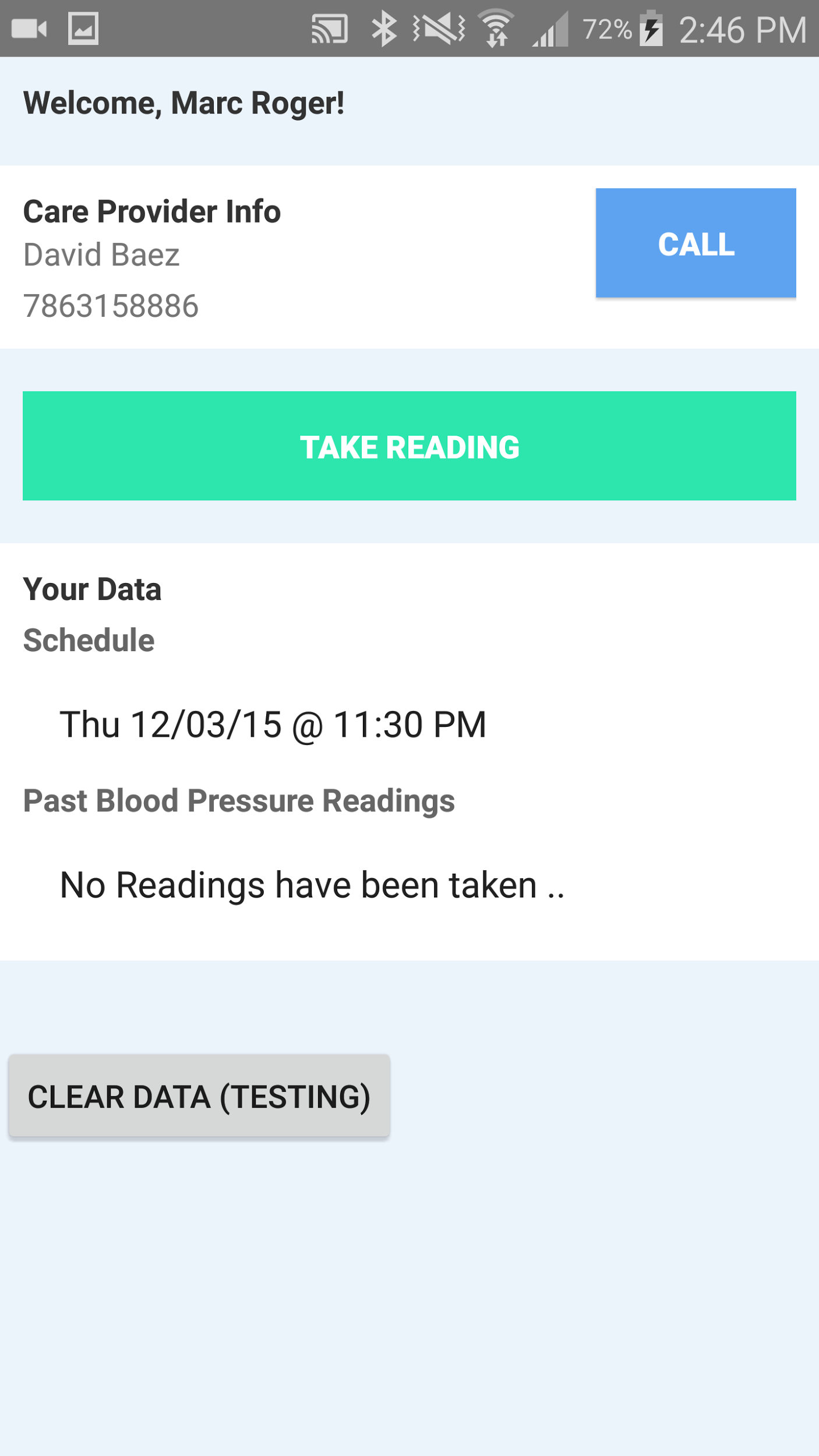


Figure 8 - Patient Application UI

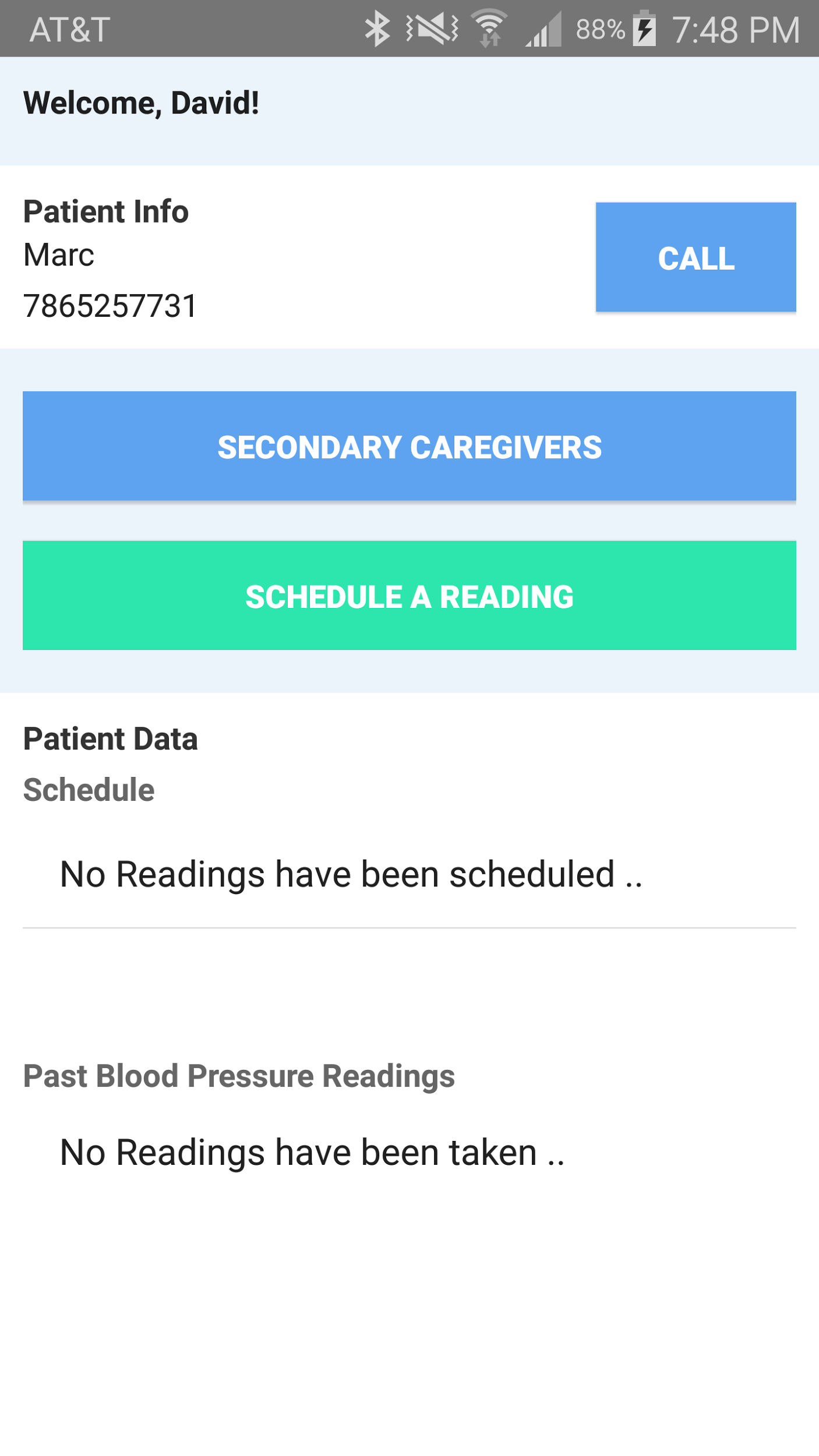
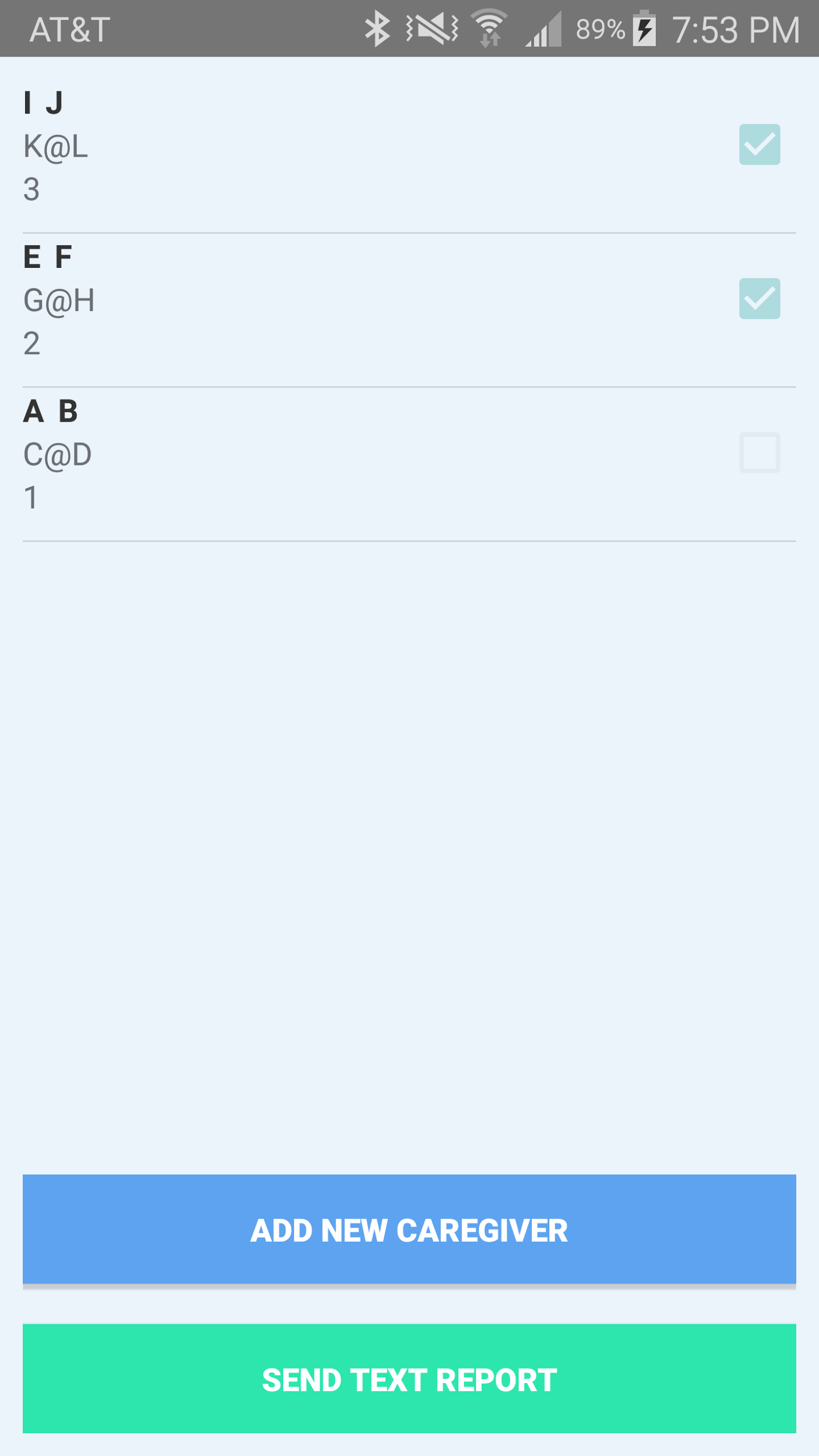


Figure 9 - Caregiver Application UI

## Appendix C - Sprint Review Reports

**Sprint 1 Report**

**Date:** September 11, 2015

**Attendees:** David Baez, Marc Roger, Peter Dickson, Masoud Sadjadi

**Discussed Topics:**

For the first sprint, only one user story was taken on as we continued to define the system and work on initial structuring. That user story was “Sending text reports to secondary caregivers” and it was successfully completed.

**Sprint 2 Report**

**Date:** September 25, 2015

**Attendees:** David Baez, Marc Roger, Peter Dickson, Masoud Sadjadi

**Discussed Topics:**

The second sprint consisted of one user story, “Store Editable Secondary Caregiver Data”, which was completed successfully with no speed bumps. At the end of the sprint, the entire project was restructured by the product manager and we expanded it by 13 user stories. Peter was very happy, and the sprint went well.

**Sprint 3 Report**

**Date:** October 9, 2015

**Attendees:** David Baez, Marc Roger, Peter Dickson, Masoud Sadjadi

**Discussed Topics:**

This sprint was focused on structuring and coding the brand new push notification system the product owner added in last minute. The system was created on time and worked successfully. All features were implemented without any speed bumps, peter was very happy, and the sprint went well.

**Sprint 4 Report**

**Date:** October 23, 2015

**Attendees:** David Baez, Marc Roger, Peter Dickson, Masoud Sadjadi

**Discussed Topics:**

This sprint was focused on allowing the caregiver to schedule a reading and send the schedule via push to the patient. Secondary features such as storing data and a phone call feature were also implemented. All features were implemented without any speed bumps, peter was very happy, and the sprint went well.

**Sprint 5 Report**

**Date:** November 6, 2015

**Attendees:** David Baez, Marc Roger, Peter Dickson, Masoud Sadjadi

**Discussed Topics:**

This sprint revolved around data storage. The entire data storage system was restructured successfully. And alerts for notifying patients of reading times were also put in place. The sprint went well, but since most updates were on the backend there wasn’t much physical progress to show. Either way, without understanding much of the changes, Peter was very happy with the application and everything was successful.

**Sprint 6 Report**

**Date:** November 20, 2015

**Attendees:** David Baez, Marc Roger, Masoud Sadjadi

**Discussed Topics:**

This sprint revolved around bluetooth. Since peter was never able to find someone to build the actual device on time, we took the task into our own hands and built a sample bluetooth device to simulate the Bluetooth process. We were unable to complete this task, but since neither of us are used to dealing with hardware, I think it was acceptable to take a little longer to complete this. The registration feature was completed successfully. Though Peter was not able to attend, he was very happy with the results in prior meetings.

**Sprint 7 Report**

**Date:** December 4, 2015

**Attendees:** David Baez, Marc Roger, Masoud Sadjadi

**Discussed Topics:**

This sprint revolved around finishing the Bluetooth connection and fixing all final bugs and UI updates. We were successful at all aspects of this sprint with no speed bumps. Though Peter was not able to attend, he was very happy with the results in prior meetings.

## Appendix D - Sprint Retrospective Reports

**Sprint 1 Retrospective**

**Date:** September 11, 2015

**Attendees:** David Baez, Marc Roger, Peter Dickson

**Discussed Topics:**

That user story “Sending text reports to secondary caregivers” was successfully completed, no no road bumps. It was our only user story for the sprint and Peter was happy with the results. We need to finish up the project UML so we are certain that we are structuring our system properly.

**Sprint 2 Retrospective**

**Date:** September 25, 2015

**Attendees:** David Baez, Marc Roger, Peter Dickson

**Discussed Topics:**

We only had one user story this sprint and completed it successfully. We had a meeting with Peter during the week where he was inspired with ideas and expanded the project by double the user stories in one sitting so now the rush is on. In order to complete the project on time, we have got to take on at least 3-4 user stories per sprint.

**Sprint 3 Retrospective**

**Date:** October 9, 2015

**Attendees:** David Baez, Marc Roger, Peter Dickson

**Discussed Topics:**

The entire push notifications system was built this sprint. Also, the entire UML structure had to be restructured. This was a huge task, but we completed it successfully and on time. Now we can start pushing real data between applications in the next sprint.

**Sprint 4 Retrospective**

**Date:** October 23, 2015

**Attendees:** David Baez, Marc Roger, Peter Dickson

**Discussed Topics:**

This sprint we focused on sending data back and forth within the patient and caregiver application. We were successfully able to make this a reality with no speed bumps, the only thing is now we have to update the storage system next sprint because it is outdated.

**Sprint 5 Retrospective**

**Date:** November 6, 2015

**Attendees:** David Baez, Marc Roger, Peter Dickson

**Discussed Topics:**

This sprint was about updating the data storage to properly store the data being transferred between the two user applications. We were able to do this successfully and we need to finally start working on the Bluetooth code which is a huge part of the system. Peter was not able to get a real custom blood pressure cuff built, so we will be building a Bluetooth cuff that will act as a simulator.

**Sprint 6 Retrospective**

**Date:** November 20, 2015

**Attendees:** David Baez, Marc Roger, Peter Dickson

**Discussed Topics:**

This sprint revolved around the Bluetooth portion of the project. We built a custom Bluetooth cuff to simulate the future RMCuff using an Arduino board with a Bluetooth chip. We successfully completed the cuff, but we were unable to finish the Bluetooth code implementation on time. We have never worked with hardware building before so it took us quite a bit to catch on. Now, next sprint we need to make sure that we complete the feature, and fix all past bugs and features.

**Sprint 7 Retrospective**

**Date:** December 4, 2015

**Attendees:** David Baez, Marc Roger, Peter Dickson

**Discussed Topics:**

This sprint was all about wrapping up the project. We needed to finish up the entire Bluetooth system, and we did successfully. Our simulator cuff is now able to connect to the patient application, and create sample blood pressure readings. Bugs were fixed in past features, and all functionality and UI was updated and brought up to date.

# References

PushBots, INC - We used the pushbots push messaging API to implement our own custom Push Notification API for use in our RMCuff Applications.