

# Virtual Cardiac Rehabilitation Nurse

---

For Coronary Bypass Surgery Patients

Design Specification

Pioneers Group – (Team 4) Aneesh Garg, Elham Mahmoudi, Evan McClure, Philip Andrews,  
Rajini Mamidi

## Contents

1.	Introduction .....	4
1.1	Project Overview and Statement of Proposal.....	4
1.2	Project Scope and Objectives .....	5
2.	Non-Functional Requirements.....	6
2.1	Product Requirements .....	6
2.2	Organizational Requirements .....	7
2.3	External Requirements .....	7
2.4	Security Requirements .....	7
3.	Functional Requirements.....	8
3.1	Use Case Diagram.....	8
3.2	Actor Description .....	9
3.2.1	User .....	9
3.2.2	Patient .....	9
3.2.3	Medical Professional.....	9
3.2.4	Notification Server.....	9
3.3	Use Case Description .....	10
3.3.1	Generate Rehab Plan.....	10
3.3.2	Manage Rehab Plan .....	11
3.3.3	Manage Rehab Log .....	13
3.3.4	Analyze Rehab Log.....	15
3.3.5	Generate Progress Report .....	16
3.3.6	Notify User.....	17
3.3.7	Login .....	18
3.3.8	Logout.....	19
4.	Architectural Design.....	20
4.1	Subsystem Architecture.....	21
4.2	Deployment Model.....	25
5.	Use Case Realization Design .....	27
5.1	Generate Rehab Plan .....	27
5.2	Manage Rehab Plan.....	29
5.3	Manage Rehab Log .....	31
5.4	Generate Progress Report.....	33
5.5	Notify User.....	35
5.6	Login .....	36

5.7	Logout .....	38
6.	Subsystem Design.....	39
6.1	Application UI.....	39
6.2	View Subsystems .....	39
6.2.1	Account View .....	39
6.2.2	Medical Professional View .....	40
6.2.3	Patient View.....	40
6.3	Service Subsystems .....	41
6.3.1	Account Service .....	42
6.3.2	Support Service.....	42
6.3.3	Notification Service .....	42
6.3.4	Report Service.....	42
6.3.5	Core Service.....	42
6.4	Database Subsystems .....	42
6.4.1	DataStore Subsystem .....	43
6.4.2	DBConnection Subsystem .....	43
6.4.3	EntityTables Subsystem.....	43
7.	Use-Case Storyboards.....	45
7.1	Login .....	46
7.2	Logout .....	46
7.3	Notify User.....	47
7.4	Generate Rehab Plan .....	48
7.5	Manage Rehab Plan.....	49
7.6	Manage Rehab Log .....	50
7.7	Generate Progress Report.....	50
8.	Prototype User Interfaces.....	51
8.1	Login Page.....	51
8.2	Patient Home Page .....	51
8.3	Medical Professional Home Page .....	52
8.4	Create Rehab Plan Page .....	52
8.5	Manage Rehab Plan Page.....	54
8.6	Patient Rehab Log Page .....	55
8.7	Progress Report Page .....	57
9.	System/Data Dependencies.....	57
9.1	Systems.....	57

9.2 Data .....	57
10. Appendices.....	58
10.1 Project Status.....	58

## 1. Introduction

Coronary bypass surgery is a procedure that restores blood flow to the patient's heart muscle by using a healthy blood vessel taken from his/her leg, arm, chest or abdomen and connects it to the other arteries in the heart. This diverts the flow of blood around a section of a blocked artery. However, patients are at potential risk to discontinue care after hospital discharge. According to a Brandeis study in *Circulation: Journal of the American Heart Association*, despite strong evidence that cardiac rehabilitation reduces disability and prolongs life, fewer than one in five people receives rehabilitation services after a heart attack or coronary bypass surgery. Cardiac rehabilitation reduces the overall risk of dying from a heart attack, the risk of future heart problems, decreases pain and the need for medicines to treat heart or chest pain, lessens the chance that the patients have to go back to the hospital or emergency room for a heart problem, improves overall health by reducing the risk factors for heart problems and improves the quality of life, making it easier for the patients to work, take part in social activities, and exercise. An absence of appropriate care and monitoring can lead to deterioration of the patient's health and safety, hospital readmission, and excessive costs to the health care system.

Technology can be utilized to contend with the nursing shortage and improve the quality of care. Virtual nursing can provide services for multiple facilities or individuals, and provide accessible, low cost nursing interventions for the patients, anytime and anywhere. The key objective of this technology is to provide the patients with guidance and information whenever they need it during their daily lives and not just during their scheduled visits. The patients can benefit from this application which offers a cybernetic nurse system at a relatively lower cost.

### 1.1 Project Overview and Statement of Proposal

Virtual Cardiac rehabilitation nurse will be a web based system. It will store the patients' data (medical and health history - medications, vital signs: blood pressure, heart rate, respiration rate, activity tolerance and exercise etc.) in a database before the surgery. Patients will have the ability to update data after the surgery, as per the planned rehabilitation interventions. The application will compare and track the patients' progress as they complete the recovery plan. We will create a web application where a patient will login with the username and password. The web application will allow patient to access his/her cardiac rehabilitation plan and have various categories like exercise, diet, medication, etc. Each category will have instructions which should be followed by the patient according to the calendar set for them. After completion of the calendar the patient and medical professional will get the feedback related to his health condition

compared to the patient's data before the surgery and continue with the plan. If the patient is not following the instructions, the application will generate an alert for the patient or medical professional. To implement the application we will create dummy rehabilitation plans and health data in database.

**Statement of Proposal:** *We propose to create a web application for patients undergoing cardiac rehabilitation after coronary bypass surgery.*

## 1.2 Project Scope and Objectives

The objective of the project is to provide virtual nurse services to a patient who has undergone a cardiac or coronary bypass surgery. The patient will be able to receive advice through the online system of the medical institution where he/she underwent surgery. The advice will be given to the user after collecting medical history and current information of the patient.

The scope of the project is proposed as below.

1. Patients will have online access to their medical center/hospital website where they had surgery through a secure access system, which uses their user name and password. We plan to create an imaginary hospital website, database and patient data as model for this purpose.
2. When the patient logs in, he/she goes to his/her personal rehabilitation plan page which has been designed by medical professionals who are responsible for the patient's rehabilitation plan. The plan includes exercise and activity, medication, diet, vital signs assessment etc. The feedback section under each weeks rehab plan will detail if he/she completed that part of the rehabilitation plan successfully or what areas need improving to optimize recovery. Moreover, the patient will be notified if she/he did not follow the instructions or the health conditions needs to be reported.
3. Her/his current health conditions can be compared to the baseline data (before surgery) and feedback can be provided as well. This feedback can be presented on the patient's webpage as text or by animation or diagram.

## 2. Non-Functional Requirements

Below are the non-functional requirements (grouped according to type) identified for this project.

### 2.1 Product Requirements

Req. No	Product Non – Functional Requirements
<b>PR.1</b>	The user interface of the application shall be implemented using HTML/CSS.
<b>PR.2</b>	The application /web server should be an Apache Tomcat version 6.
<b>PR.3</b>	The database should be a version of Microsoft SQL server 2008.
<b>PR.4</b>	Application and Database Server maintenance schedule should be between 11.00PM Saturday to 5.00AM Sunday every week.
<b>PR.5</b>	Rehabilitation plans must be simple enough to view that a patient can understand them with no external training.
<b>PR.6</b>	Progress reports must be generated within 120 seconds so that the data is relevant.
<b>PR.7</b>	Application must be available to Patients/Medical Staff 24/7 with the exception of scheduled outages with a minimum of 98% uptime.
<b>PR.8</b>	SQL Database should be able handle at least 2 GB of data, scale and maintain data integrity.
<b>PR.9</b>	The application should require no more than 1 hour of training for the typical user to be proficient with the system. Once trained, the typical user should be able to fill out their daily progress report in less than 15 minutes.
<b>PR.10</b>	SQL Database will store and archive historical data (over 180 days old).
<b>PR.11</b>	Nightly backups will be performed on SQL Database.
<b>PR.12</b>	Multiple users should be able to login to the application at one time.
<b>PR.13</b>	Appropriate validation, escaping, and handling of special characters to avoid XSS and other server side related application vulnerabilities.
<b>PR.14</b>	The database should have logs of who last modified the patient logs and rehabilitation plan.

<b>PR.15</b>	The response time per click should be under 5 seconds for navigation and less than 10 seconds if a query or request is sent.
<b>PR.16</b>	Percentage of events causing failure should be minimal (<5%).
<b>PR.17</b>	A log of changes to the rehabilitation plan must be maintained in the application

## 2.2 Organizational Requirements

Req. No	Organizational Non – Functional Requirements
<b>OR.1</b>	The virtual cardiac rehabilitation nurse system development process and deliverable documents must conform to the standards set by Dr. Tolone.
<b>OR.2</b>	The deliverables must be submitted using Moodle.
<b>OR.3</b>	The deliverables must be in the English language.

## 2.3 External Requirements

Req. No	External Non – Functional Requirements
<b>ER.1</b>	Virtual cardiac rehabilitation nurse application must be accessible from all HTML5/CSS3 compatible web browsers.

## 2.4 Security Requirements

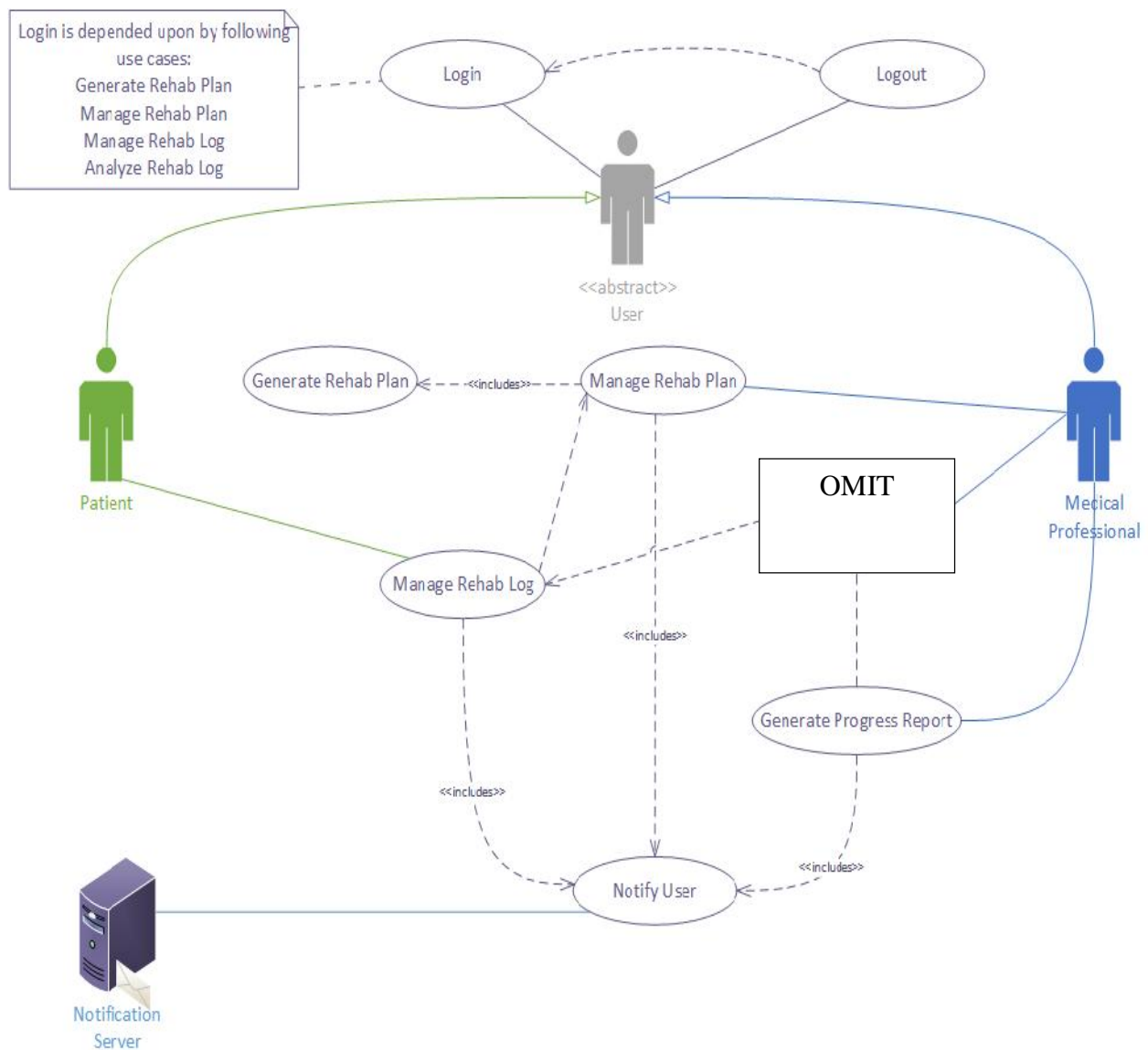
Req. No	External Non – Functional Requirements
<b>SR.1</b>	The patients record must be visible only the patient and his medical professionals.
<b>SR.2</b>	Patient record must be inaccessible to anyone except patient and medical professionals.
<b>SR.3</b>	The patient should not be able to edit the rehabilitation plan.
<b>SR.4</b>	Doctors should have access to edit the information of their own patients only.
<b>SR.5</b>	Doctors should be able to see anybody's medical history or progress.
<b>SR.6</b>	The database should store password in encrypted form only.
<b>SR.7</b>	Data must be stored in a secure persistent data source.



<b>SR.8</b>	The password of the user should be encrypted at the client end before passing it to the server.
<b>SR.9</b>	User information should not be sent to server in a query string.
<b>SR.10</b>	Users can not login without a valid username/password.
<b>SR.11</b>	Strong passwords required (6-10 characters, Upper/Lower Case required, Min. 1 number).

### 3. Functional Requirements

#### 3.1 Use Case Diagram



The Use Case Diagram consists of 7 use cases, 3 real actors and 1 abstract user as an actor. The actors Patient and Medical Professional are specialized Users. The diagram contains the Manage Rehab Plan use case which includes Generate Rehab Plan use case. The Manage Rehab Plan use case helps the Medical Professional to manage rehabilitation plan and is depended upon by the Manage Rehab Log use case. Manage Rehab Log use case helps patient enter his daily log of activities and is depended upon by Analyze Rehab Log use case. Analyze Rehab Log use case helps medical professionals analyze activities entered by the Patient and is depended upon by Generate Progress Report use case. Generate Progress Report use case generates the progress report of the Patient to be seen by medical professionals and patient. The use cases Manage Rehab Plan, Manage Rehab Log and Generate Progress Report includes Notify User use case in which Notification Server notifies corresponding user of any changes made due to the three use cases. The Use Case Diagram also contains Login and Logout use cases which helps user login and logout of the application. Logout use case depends upon Login use case while Login use case is depended upon by Generate rehab Plan, Manage Rehab Plan, Manage Rehab Log and Analyze Rehab Log use cases.

## **3.2 Actor Description**

### **3.2.1 User**

User is an abstract actor which is specialized by Patient and Medical Professional.

### **3.2.2 Patient**

Patient is an actor who has undergone a cardiac surgery and has signed up for Virtual Cardiac Rehabilitation Nurse application for rehabilitation after surgery.

### **3.2.3 Medical Professional**

Medical professional is an actor either a doctor or nurse who supervises rehabilitation of a patient.

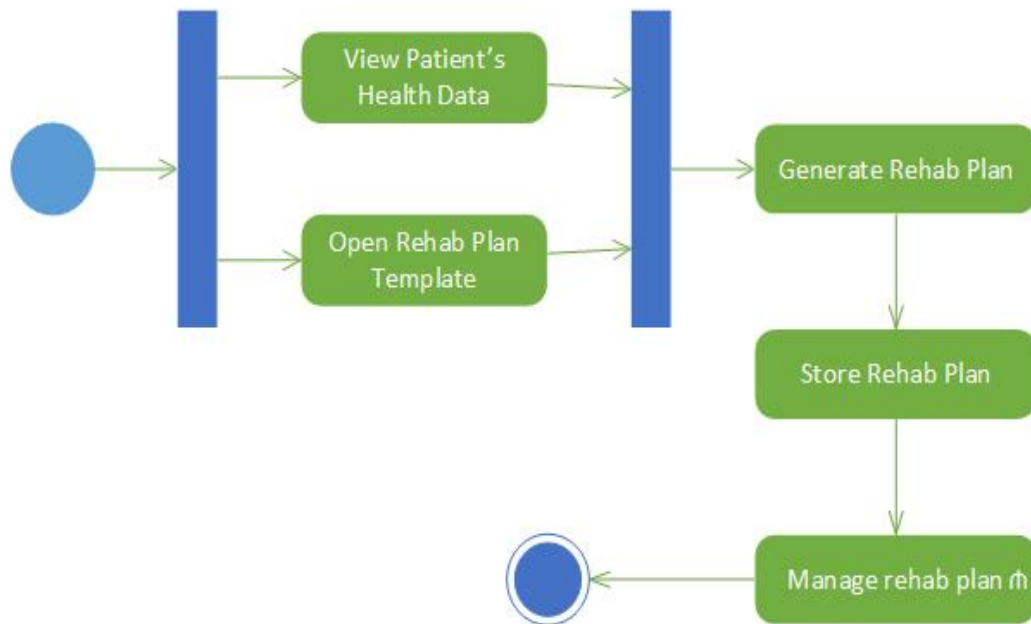
### **3.2.4 Notification Server**

Notification Server is an external server which sends notifications to all users.

### 3.3 Use Case Description

#### 3.3.1 Generate Rehab Plan

Elements	Description
Use case	Generate Rehab Plan
Participating Actors	Medical Professional
Goal	Medical Professional should be able to generate rehab plan
Pre-Conditions	<ul style="list-style-type: none"><li>• The system has patient's health data</li><li>• The system has rehab plan template</li><li>• Medical Professional is logged in</li></ul>
Post-Conditions	<ul style="list-style-type: none"><li>• Rehab plan is available for patient notification</li></ul>
Triggers	<ul style="list-style-type: none"><li>• New patient is admitted</li><li>• Rehab plan is generated and manual action on rehab template is done</li></ul>
Primary and Alternative Flow of Events	<ol style="list-style-type: none"><li>1. The use case begins when Medical Professional views patient's health data and opens rehab plan simultaneously.</li><li>2. He then generates the rehab plan using above data.</li><li>3. He then stores generated rehab plan.</li><li>4. Manage Rehab Plan use case is triggered</li><li>5. The use case ends successfully.</li></ol>



### 3.3.2 Manage Rehab Plan

Elements	Description
Use case	Manage Rehab Plan
Participating Actors	Medical Professional
Goal	Medical Professional should be able to view/update rehab plan.
Pre-Conditions	<ul style="list-style-type: none"> <li>The system has patient's health data</li> <li>The system has rehab plan template</li> <li>Medical Professional is logged in</li> </ul>
Post-Conditions	<ul style="list-style-type: none"> <li>Rehab plan is available for patient notification</li> </ul>
Triggers	<ul style="list-style-type: none"> <li>New patient is admitted</li> <li>Analyzed Rehab Log is available</li> <li>Manual action on rehab template is done</li> </ul>

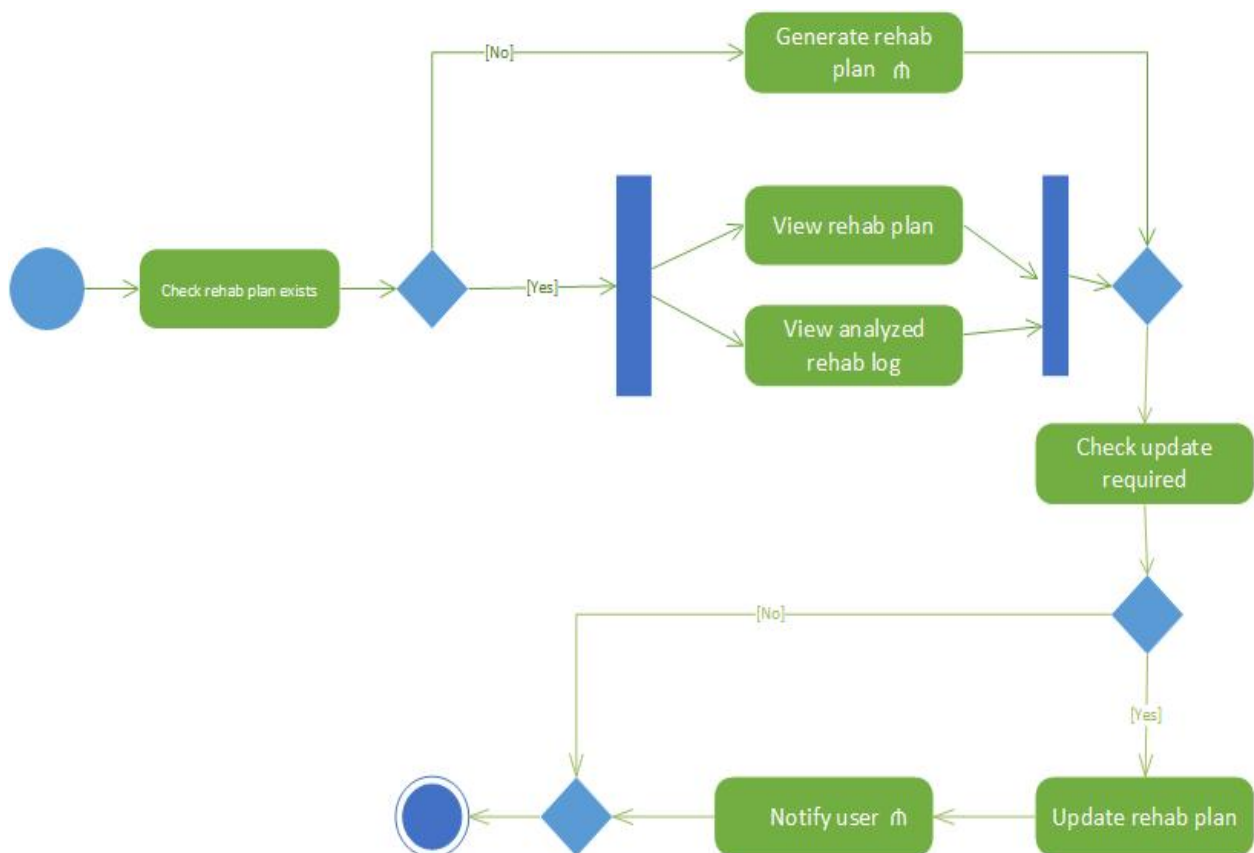
## Primary and Alternative Flow of Events

### Primary Flow:

1. The use case begins with Medical Professional checking the existence of rehab plan.
2. If a plan exists Medical Professional views rehab plan and analyze rehab log simultaneously.
3. He then checks whether update is required.
4. If update is required then he updates the rehab plan.
5. After update Notify User use case is triggered.
6. The use case ends successfully.

### Alternative Flows:

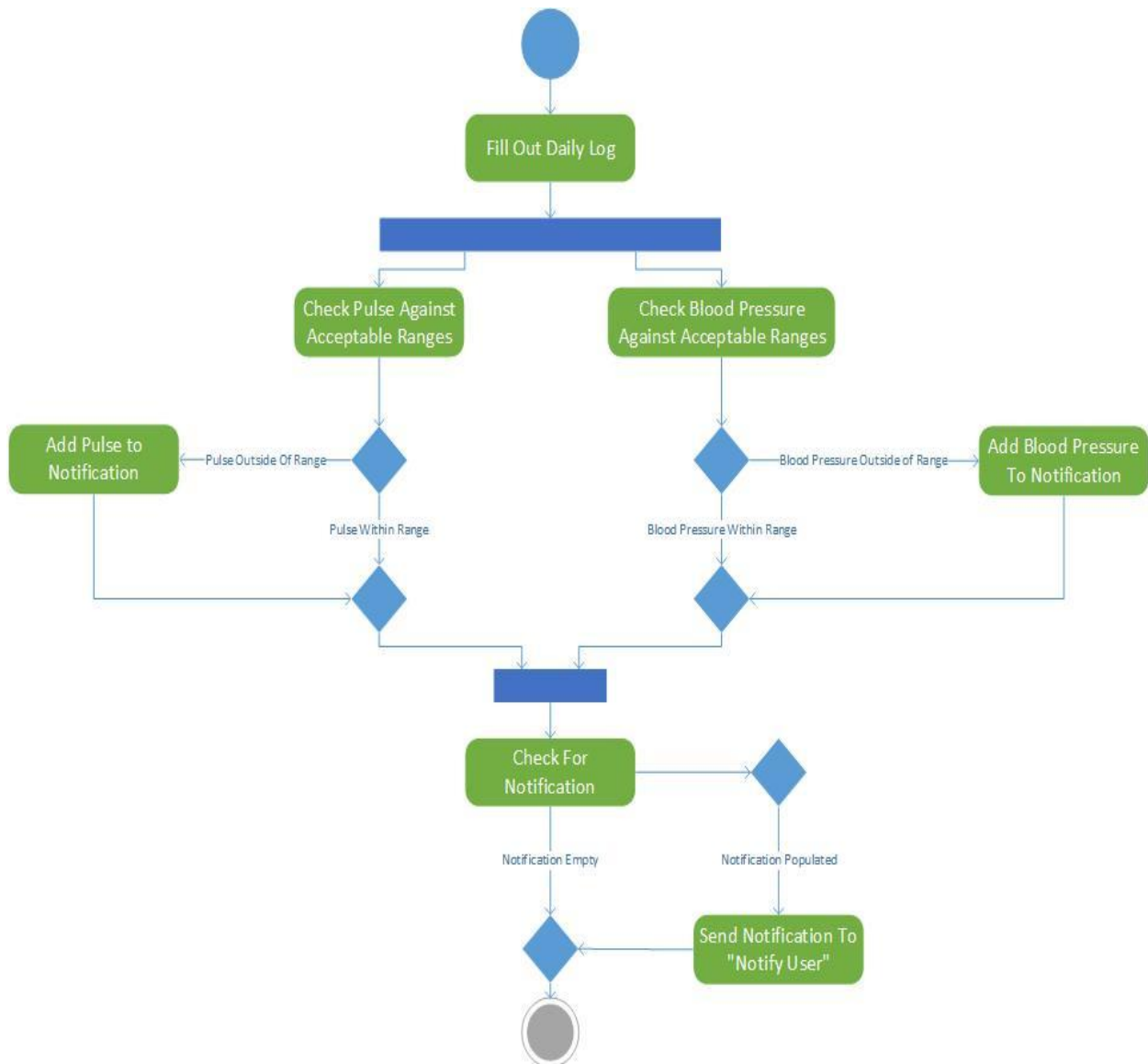
- If in step 1 Rehab plan does not exist then Generate Rehab Plan use case is triggered after the use case resumes at step 3.
- If in step 3 update is not required the use case ends successfully.



### 3.3.3 Manage Rehab Log

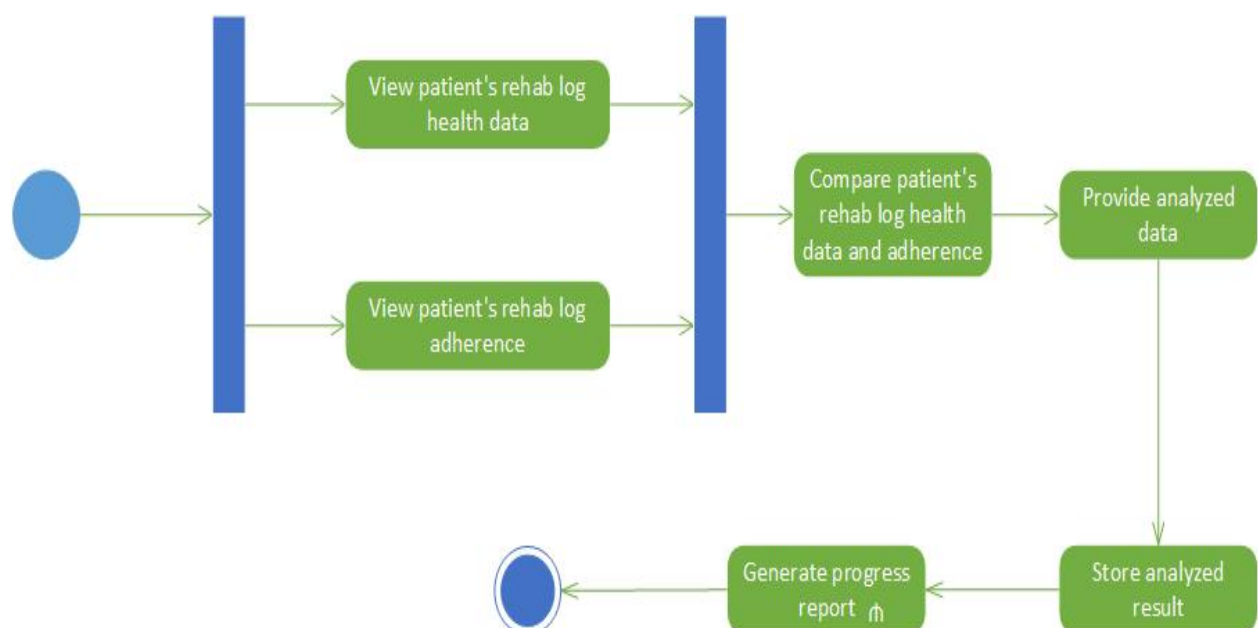
Elements	Description
Use case	Manage Rehab Log
Participating Actors	Patient
Goal	Allow the patient to fill out a daily log of activities and other metrics that will be used to track their progress during their rehabilitation.
Pre-Conditions	<ul style="list-style-type: none"> <li>Rehab Plan must be created via the “Manage Rehab Plan” use case.</li> <li>Patient must be registered for application.</li> <li>Patient must be logged in.</li> </ul>
Post-Conditions	<ul style="list-style-type: none"> <li>Rehab log is available to be used in the “Analyze Rehab Log” use case</li> <li>If the data entered into the report falls outside of predefined ranges specified in the “Manage Rehab Plan” use case, the “Notify User” use case is activated.</li> </ul>
Triggers	<ul style="list-style-type: none"> <li>Manual selection by Patient</li> </ul>
Primary and Alternative Flow of Events	<ol style="list-style-type: none"> <li>1. A user will, either on their own volition, or because they have been reminded to by a notification, sent from the “Notify User” use case, will use the application to fill out a rehab log of their daily activities.</li> <li>2. This log will include both critical and non-critical data. Critical data will consist of blood pressure and pulse, and all other metrics will be considered non-critical.</li> <li>3. Upon saving blood pressure and pulse, the values entered into the log will be compared with the acceptable ranges previously defined in the “Manage Rehab Plan” use case.</li> <li>4. If the values entered fall outside of the acceptable range, a notification addressed to both the medical professional and patient will be created containing the information relevant to this exception.</li> </ol>

5. If a notification has been generated when comparison to acceptable ranges is complete, the notification will be sent to both the patient and the medical professional via the “Notify User” use case and the process terminates. If no notification has been generated, the process terminates without sending anything.



### 3.3.4 Analyze Rehab Log

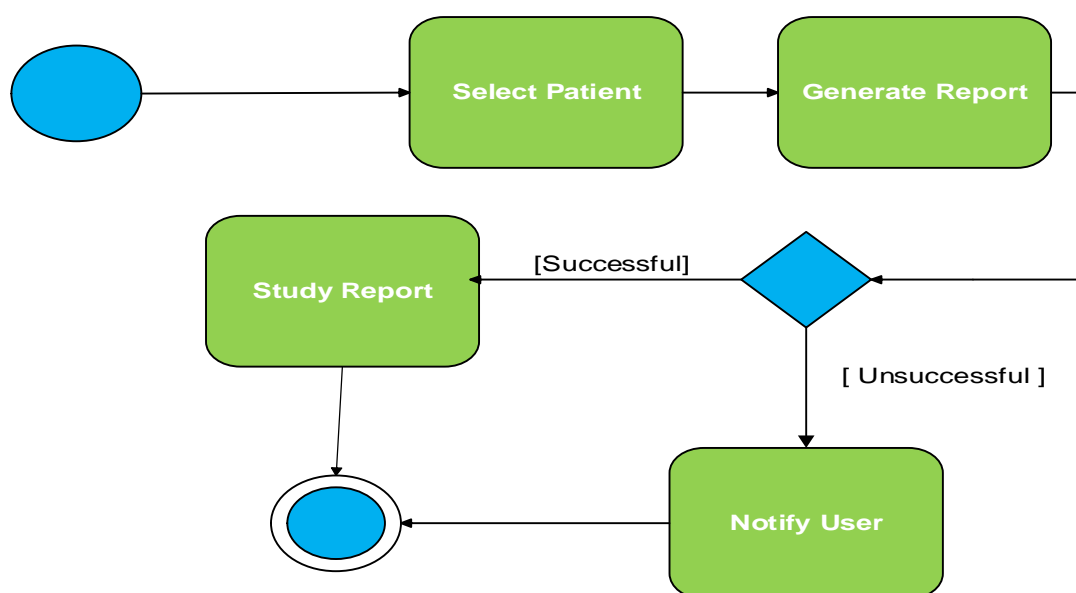
Elements	Description
Use case	Analyze Rehab Log
Participating Actors	Medical Professional
Goal	Medical Professional should be able to analyze rehab log.
Pre-Conditions	<ul style="list-style-type: none"> <li>Rehab log is available for analyzing</li> </ul>
Post-Conditions	<ul style="list-style-type: none"> <li>Patient's rehab log's analysis is available for generating progress report.</li> </ul>
Triggers	<ul style="list-style-type: none"> <li>Manual selection by Medical Professional</li> </ul>
Primary and Alternative Flow of Events	<ol style="list-style-type: none"> <li>The use case starts with Medical Professional viewing patient's rehab log health data and patient's rehab log adherence simultaneously.</li> <li>He then compares patient's rehab log health data and adherence.</li> <li>Medical Professional provides analyzed data.</li> <li>He then stores the analyzed result.</li> <li>This use case ends successfully with the triggering of Generate Progress Report use case.</li> </ol>





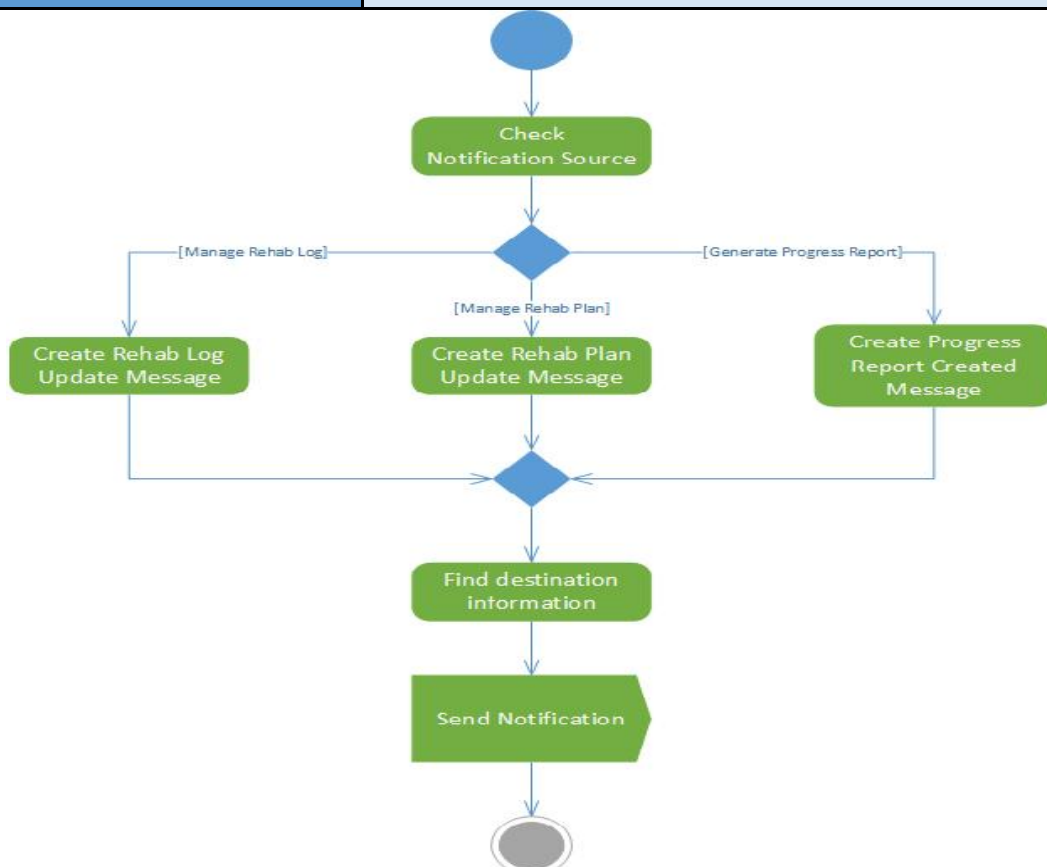
### 3.3.5 Generate Progress Report

Element	Description
Use Case:	Generate Progress Report
Participating Actors:	Medical Professional
Goal:	Produce documents which present the progress of the patient's recovery to the medical professional.
Pre-Conditions:	<ul style="list-style-type: none"> <li>• Access to the application</li> <li>• Access to the patient record</li> <li>• Rehabilitation Plan Existence</li> <li>• Analyzed Rehab log information is available</li> </ul>
Post Conditions:	If necessary Medical professional updates Rehabilitation plan
Triggers:	<ul style="list-style-type: none"> <li>• Manually triggered by the Medical Professional.</li> </ul>
Primary and Alternative Flow of Events:	<ol style="list-style-type: none"> <li>1. Medical Professional selects the patient for which he wants to see the report.</li> <li>2. He then runs the report for the selected patient.</li> <li>3. If application is not able to generate a report, notifications will be sent to medical professional with the error message. (The notify user use case will be initiated).</li> <li>4. The Medical Professional studies the report</li> </ol>



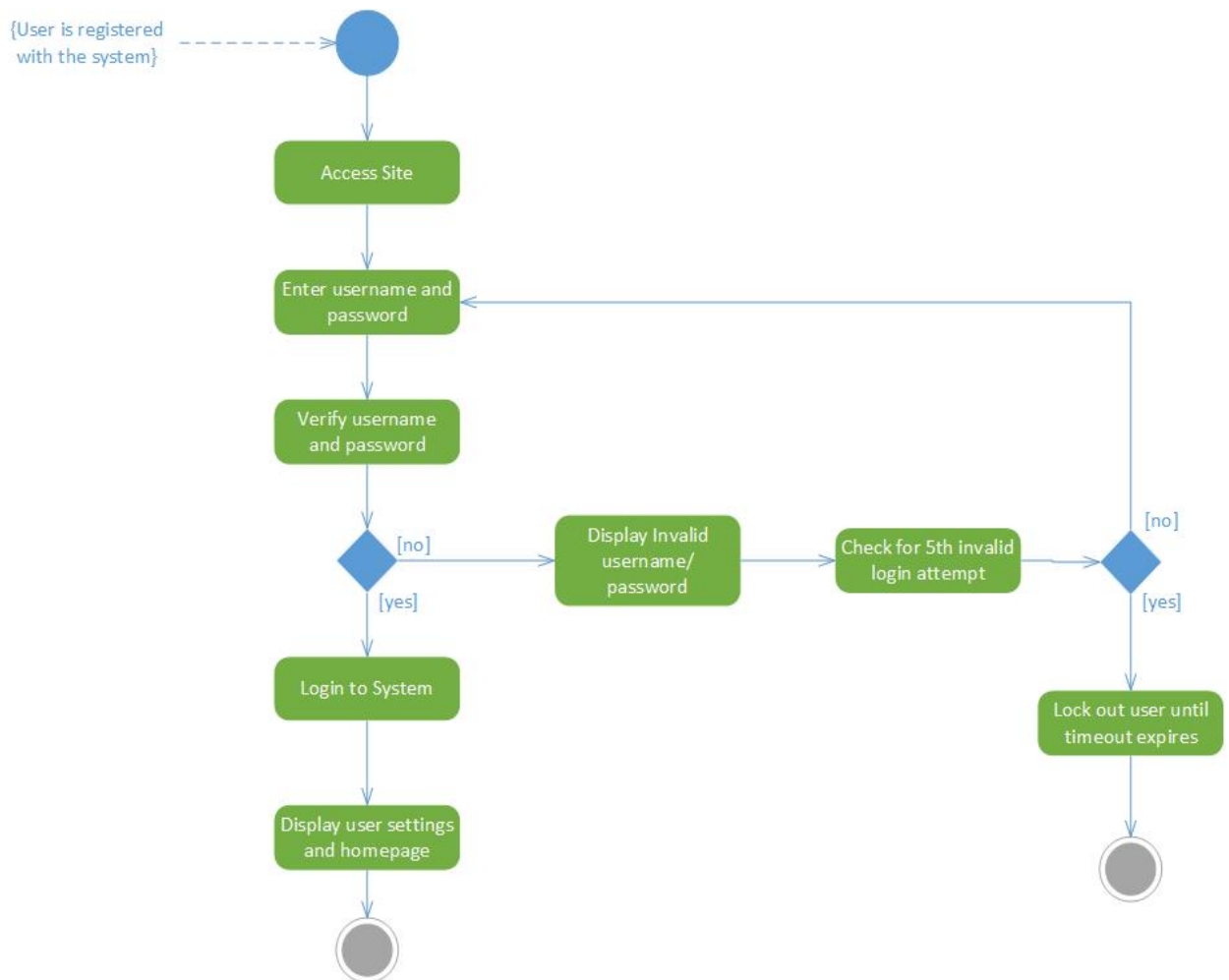
### 3.3.6 Notify User

Element	Description
Use Case:	Notify User
Participating Actors:	Notification Server
Goal:	Send notifications to Patient or Medical Professional.
Pre-Conditions:	<ul style="list-style-type: none"> <li>Notification server is online</li> </ul>
Post Conditions:	Notification is sent to user to take further action
Triggers:	<ul style="list-style-type: none"> <li>Changes are made to rehab plan</li> <li>Changes are made to Rehab Log</li> <li>Progress Report is Generated</li> </ul>
Primary and Alternative Flow of Events:	<ol style="list-style-type: none"> <li>1. Check for the source of notification message</li> <li>2. Create the corresponding message.</li> <li>3. Find the destination information where notification needs to be sent.</li> <li>4. Send the notification to the destination.</li> </ol>



### 3.3.7 Login

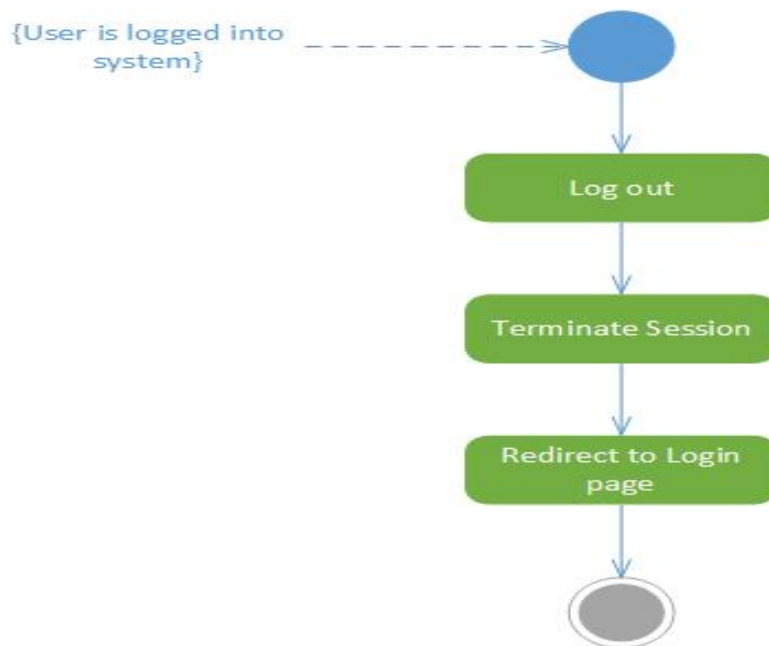
Element	Description
Use Case:	Login
Participating Actors:	User
Goal:	Users should be able to access the web application and login with valid username and password in order to access rehab plan.
Pre-Conditions:	<ul style="list-style-type: none"><li>• User has Internet access</li><li>• User is already registered</li><li>• Application is available (no outage)</li></ul>
Post Conditions:	<ul style="list-style-type: none"><li>• Homepage content is displayed to patient, doctor, or medical staff</li><li>• System pulls user specific information from Database and displays to end user</li><li>• Additional application functionality is available to user.</li></ul>
Triggers:	<ul style="list-style-type: none"><li>• Manual triggering</li></ul>
Primary and Alternative Flow of Events:	<p><b>Primary Flow:</b></p> <ol style="list-style-type: none"><li>1. User accesses the web application by visiting URL</li><li>2. User enters username</li><li>3. User enters password</li><li>4. User submits information for authentication</li><li>5. System validates information</li><li>6. User logs into system and is directed to homepage</li></ol> <p><b>Alternative Flow:</b></p> <ol style="list-style-type: none"><li>1. User accesses the web application by visiting the URL</li><li>2. User enters incorrect username or</li><li>3. User enters incorrect password</li><li>4. System displays error message</li><li>5. After 5 incorrect tries system locks user out</li></ol>



### 3.3.8 Logout

Element	Description
Use Case:	Logout
Participating Actors:	User
Goal:	A User should be able to logout of the web application.
Pre-Conditions:	<ul style="list-style-type: none"> <li>User has Internet access</li> <li>User is logged into account (UC 3.3.7)</li> <li>Application is available (no outage)</li> </ul>
Post Conditions:	<ul style="list-style-type: none"> <li>User is redirected out of application</li> <li>All functionality of application is disabled</li> </ul>
Triggers:	<ul style="list-style-type: none"> <li>Manual triggering</li> <li>Session Expired</li> </ul>
Primary and Alternative Flow of Events:	<b>Primary Flow:</b> <ol style="list-style-type: none"> <li>User logs out of application</li> </ol>

	<ol style="list-style-type: none"> <li>2. System terminates user session</li> <li>3. Redirected to login screen for user to login</li> </ol> <p><b>Alternative Flow:</b></p> <ol style="list-style-type: none"> <li>1. User continues navigation of web application content and functionality</li> </ol>
--	--

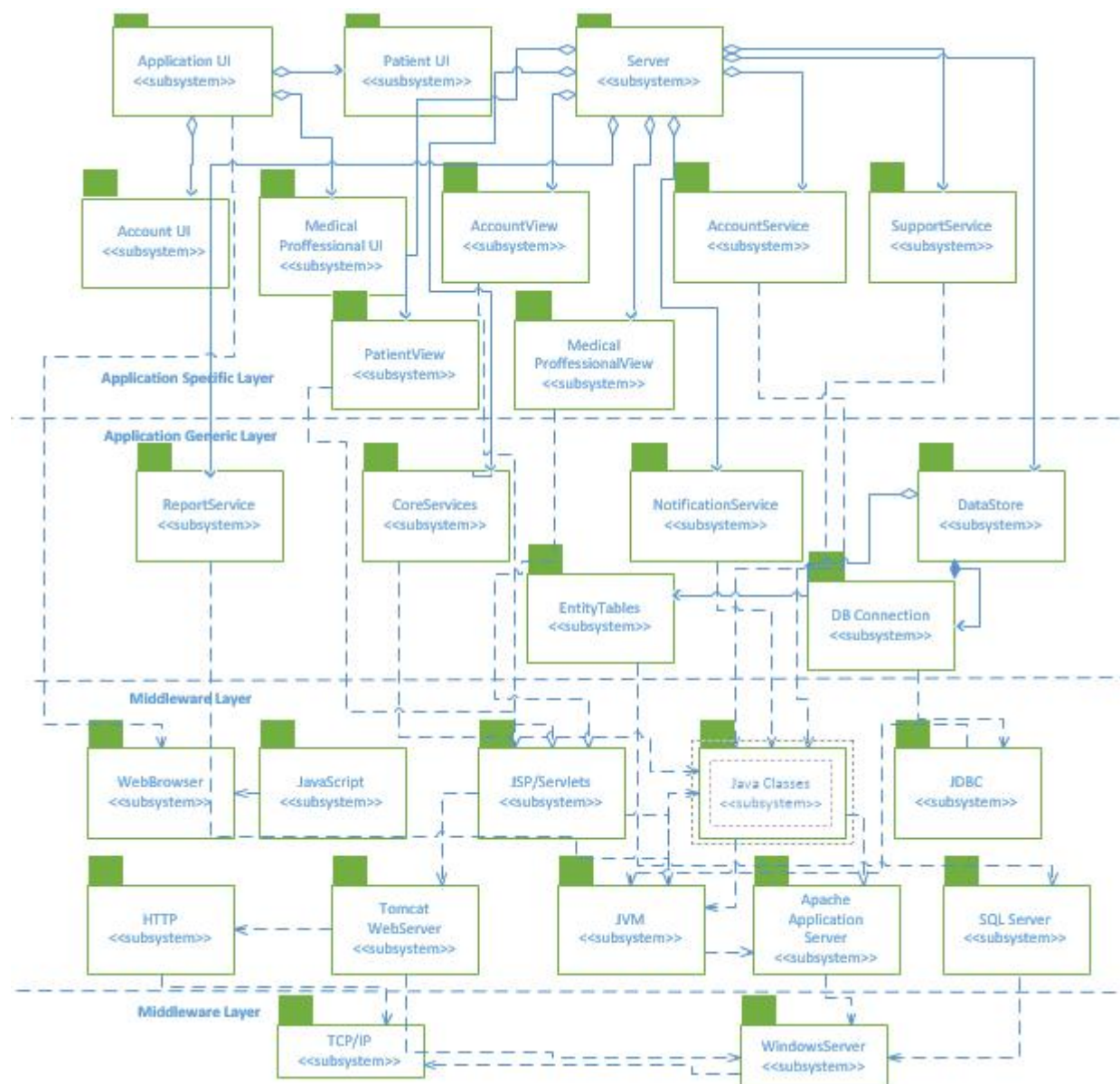


## 4. Architectural Design

Our Subsystem architecture contains 4 layers: Application Specific ( includes Application UI, Analysis Server, and Server), Application Generic( includes Notification Server, Application Data Store , Database connection, and Database Tables), Middleware ( includes Tomcat Web Server, Web Browser, Java script, HTML/CSS, Java Servlet, JVM, IIS Web Server, Net Framework, Apache Web Server, ODBC, MySQL , HTTP and JVM) , and System Software( includes TCP/IP and Windows Server). The reason we chose this design over the alternatives and used IIS as well as Tomcat is for risk mitigation. By leverage both technologies we are using the strengths of our two developers and allowing them to design or code in the areas they are most familiar with. Many of the subsystems are standard to any application such as HTTP, ODBC, HTML/CSS, and Java Servlet which allow the user to communicate and interact with the application and data store. The Alternatives we discussed was a centralized web server which handled all the different functionality including services, queuing, and other configuration

settings. Although, this approach would work in many instances we believe it would increase the risk and could cause potential issues that we don't have the expertise to troubleshoot. It could also increase the development time as both programmers would work together seamlessly instead of splitting parts of the system up. We also discussed other alternatives such as OCI for our database connection but decided to stay with the more standard ODBC for connectivity. The majority of our subsystem architecture decisions came from the identified risks as we worked to mitigate these.

## 4.1 Subsystem Architecture



## **Application Specific Layer:**

***Server subsystem*** – The server subsystem is an aggregate of several other service and view subsystems which is the foundation of the application. This subsystem handles the interactions between the clients and system. It helps to manage resource allocation.

***Patient View subsystem*** – The patient view subsystem contains all the views or JSP pages and servlets that will be returned to the patient upon request. These are dependent on Entity Tables and JVM subsystems.

***Account View subsystem*** – The account view subsystem contains all the views or JSP pages and servlets that will be returned to the patient or medical professional when they login and request information. This is dependent on Entity Tables and JVM subsystems.

***Medical Professional View subsystem*** – This subsystem contains all the views or JSP pages and servlets that will be returned to the medical professional when they request information.

***Account Service*** – This service handles the verification and sessions related to login and logout of users. It connects to the database to retrieve username and password.

***Support Service*** - Is responsible for getting the template and patient health data that will be used for generating a rehab plan. This will be deployed on the application server. This is dependent on Java classes and Apache subsystems.

***Application UI subsystem*** – The application user interface subsystem is what the different users interact with in order to navigate around the application. The user must authenticate by first passing their login and password to the system. Different forms or UI's will be displayed based on the user's permissions and allows the user to update rehab log, view logs, check rehab plan, update rehab plan, or other functions based on roles. The Application UI depends on notification server and the user having a web browser that is compatible with HTML/CSS and Javascript.

***Account UI subsystem*** – The account UI subsystem displays the login page where the user will enter the username and password. Upon verification they will go to the homepage.

***Medical Professional UI subsystem*** – This subsystem will display the homepage of a medical professional when they login with their username and password if they have the role of doctor.

***Patient UI subsystem*** – This subsystem will display the homepage of a patient when they login with their username and password if they have the role of patient.

## **Application Generic Layer:**

**Data Store subsystem** – The application data store subsystem contains two subsystems and controls the interactions between the application level and data. It contains two subsystems – database connection and entity tables. It does not require a high performance database but must be large enough to handle patient and rehab plan data. The database system will assist with report generation and complex queries.

**Database Connection subsystem** – This subsystem will allow the application to communicate with the database in order to retrieve or store data within the database tables. The database connection subsystem depends on ODBC to communicate.

**Entity Tables subsystem** – This subsystem organizes data in a logical fashion and helps control relationships between data elements. It depends on SQL Server in order to return data for queries relevant to patient, rehab log, or rehab plan data.

**Notification Service subsystem** – This subsystem helps controls the interactions between the user and the messages that are sent from the system. This will include alerts and notifications which help the patient and doctor communicate. It will assist with letting the doctor know how the patient is doing on the rehab plan as well as any emergencies such as high blood pressure which requires immediate attention. The notification server depends on Application Data Store.

**Report Service subsystem** – This subsystem helps manage the rehab logs the patient fills out and executes the business logic in order to generate a progress report to display to the patient and doctor. It depends on SQL server and Apache Application server.

**Core Services subsystem** – This subsystem handles the bulk of the business logic and processing for the application. It is dependent on Apache Application server and Tomcat as well as windows server. This will handle generation of the rehab plan and progress reports.

## **Middleware Layer:**

**Tomcat Web Server subsystem** – This subsystem deploys all the web services that are implemented in java.

**Apache Web Server subsystem** – This subsystem deploys all the UI related HTML, CSS, Javascript files.



**Javascript subsystem** – This subsystem is used to build the different javascript client side validations and web service calls. It contains the javascript files that will validate on client side and call web services to get required data.

**JSP/Java Servlet subsystem** – The java servlet subsystem provides the servlets that will be called by the JVM subsystem. This will return results based on a request from the user.

**JVM subsystem** – This subsystem acts as the intermediate layer between the java program and the web server. This subsystem is responsible for compiling and executing the java.

**Web Browser subsystem** – This subsystem communicates with the web server using the HTTP protocol. Any modern day web browser that supports HTML 5.0/CSS can be used.

**HTTP subsystem** – The Hypertext Transfer Protocol subsystem allows the web browser and servers to communicate. It is required that username and passwords are not passed in plaintext over HTTP. Several other subsystems depend on HTTP to communicate.

**SQL Server subsystem** – The SQL server subsystem provides a structured language to execute queries and pull data from the database. This subsystem depends on the DB tables being populated and data store. It is depended upon by ODBC.

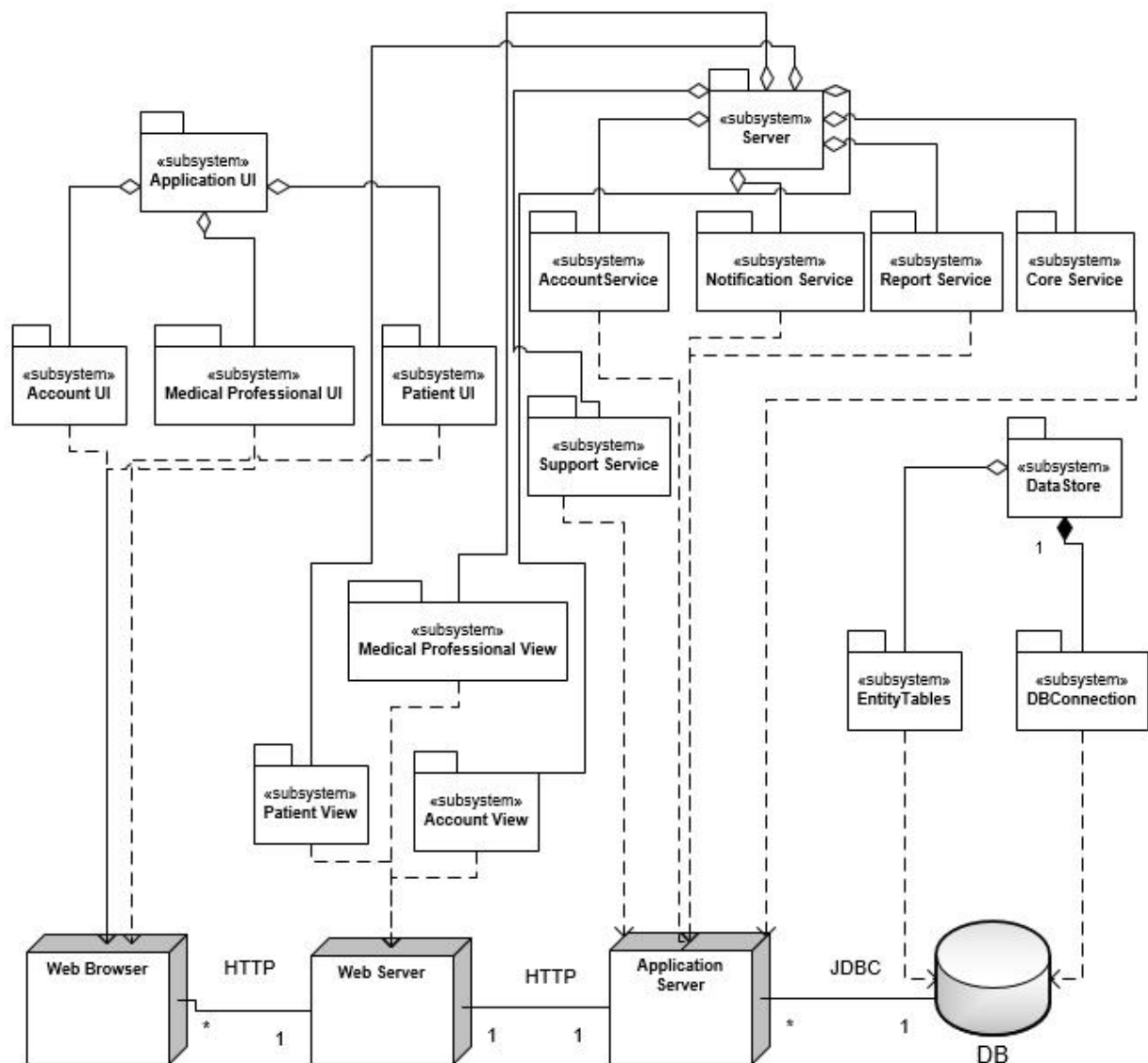
**ODBC subsystem** – The ODBC subsystem allows the database and web server to communicate with each other. This will allow data to be transferred and displayed to end users as well as allow for database connectivity for the application to store data. The ODBC subsystem depends on MySQL in order to work effectively.

### **System Software Layer:**

**TCP/IP subsystem** – The TCP/IP subsystem will allow for end-to-end connectivity and specify how data should be formatted, addressed, transmitted, routed, and received by the system.

**Windows Server subsystem** – The windows server subsystem hosts the web application and allows multiple services to operate on top of it. Windows server depends on TCP/IP to transmit messages to other parts of the application. Almost every other subsystem has to contact the windows server subsystem to operate as intended. It manages resources of assignments, gets request from the browser, returns assignments to the browser, verifies users with authentication, and configures the system based on configuration files.

## 4.2 Deployment Model



For our deployment three servers (*Application Server, Web Server, and Database Server*) and a client (*Web Browser*) have been identified where the subsystems would be deployed. However, for implementation this will most likely be a single host configuration.

The application server will just be a Tomcat *Server* which will host the application. We decided to use Tomcat to leverage a team member's strength and reduce the risk of using unfamiliar technologies.

The *server subsystem* is acknowledged as an aggregate of *Account Service, Notifications Service, Report Service, Core Service and Support Service*.

**Account Service Subsystem:** Contains the methods used to authenticate users and verify username and password credentials. This service will manage user sessions and determine what access permissions they have to the application. This is deployed on the Tomcat Application Server.

**Notifications Service Subsystem:** Distributes the notifications to the Users and is deployed on the Tomcat Application Server.

**Report Service Subsystem:** Contains the business logic for managing the Rehab logs from the patient and is deployed on the Tomcat Application Server. This will generate progress reports and provide status updates to the patient.

**Core Service Subsystem:** Contains the meat of the business logic. This service will handle the business computation and generation of the progress reports and rehab plan. It will be deployed on the Tomcat application server.

**Support Service Subsystem:** Is responsible for getting the template and patient health data that will be used to generate a rehab plan. This will be deployed on the application server.

**Application Data Store Subsystem:** Stores the data used by the application and deployed on the Tomcat Application Server.

The application data kept in the **Database** appears in a table format form of DB tables. These communicate with the application server using a DB connection. Hence the **Data Store Subsystem** has been identified as an aggregate of **Entity Tables** and **DB Connection** subsystem. The Entity Tables deployed on the **Database Server** and the DB Connection is deployed on the application Server to connect to the Database Server using an ODBC Connection. The Association multiplicity between Application Server and Database Server is identified as 1:1.

The **Application UI** consists of interface components between the application and the user. The Application UI subsystem is deployed at the Client or Web Brower. It consists of the login UI, Patient UI, and Medical Professional UI.

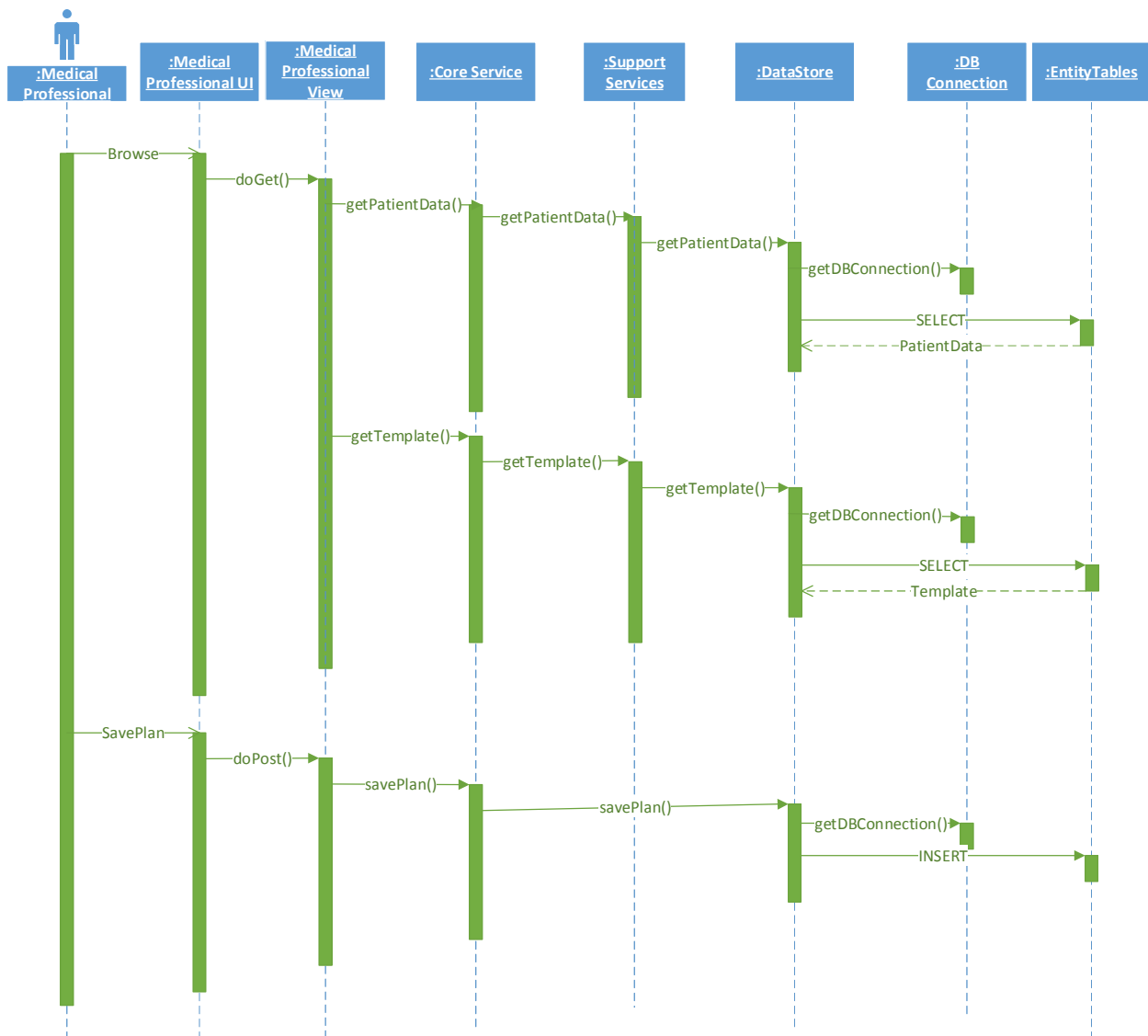
The **Web browser** communicates with the Web Server using http. Multiple requests can come from many web browsers or clients, the association multiplicity is identified as \*:1.

The **Web Server** dialogues with the Application Server using REST protocol. The **Tomcat server** communicates to each other using web services.

## 5. Use Case Realization Design

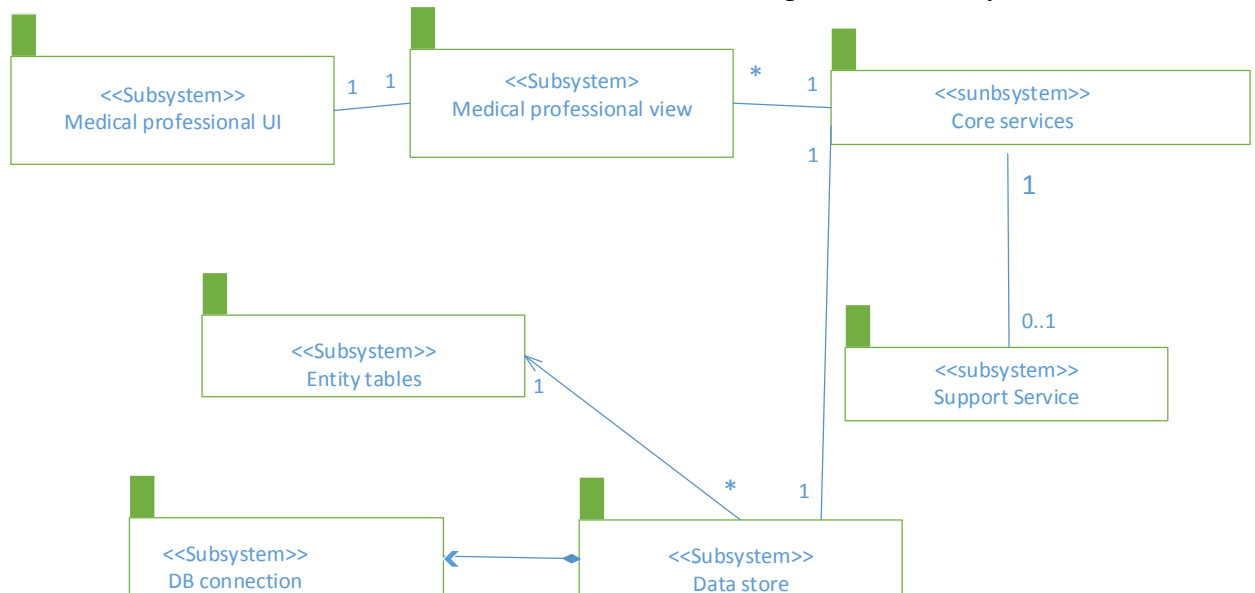
This section includes for each use case a sequence diagram depicting important interactions among subsystems as well as class diagrams depicting relationships among interaction participants.

### 5.1 Generate Rehab Plan



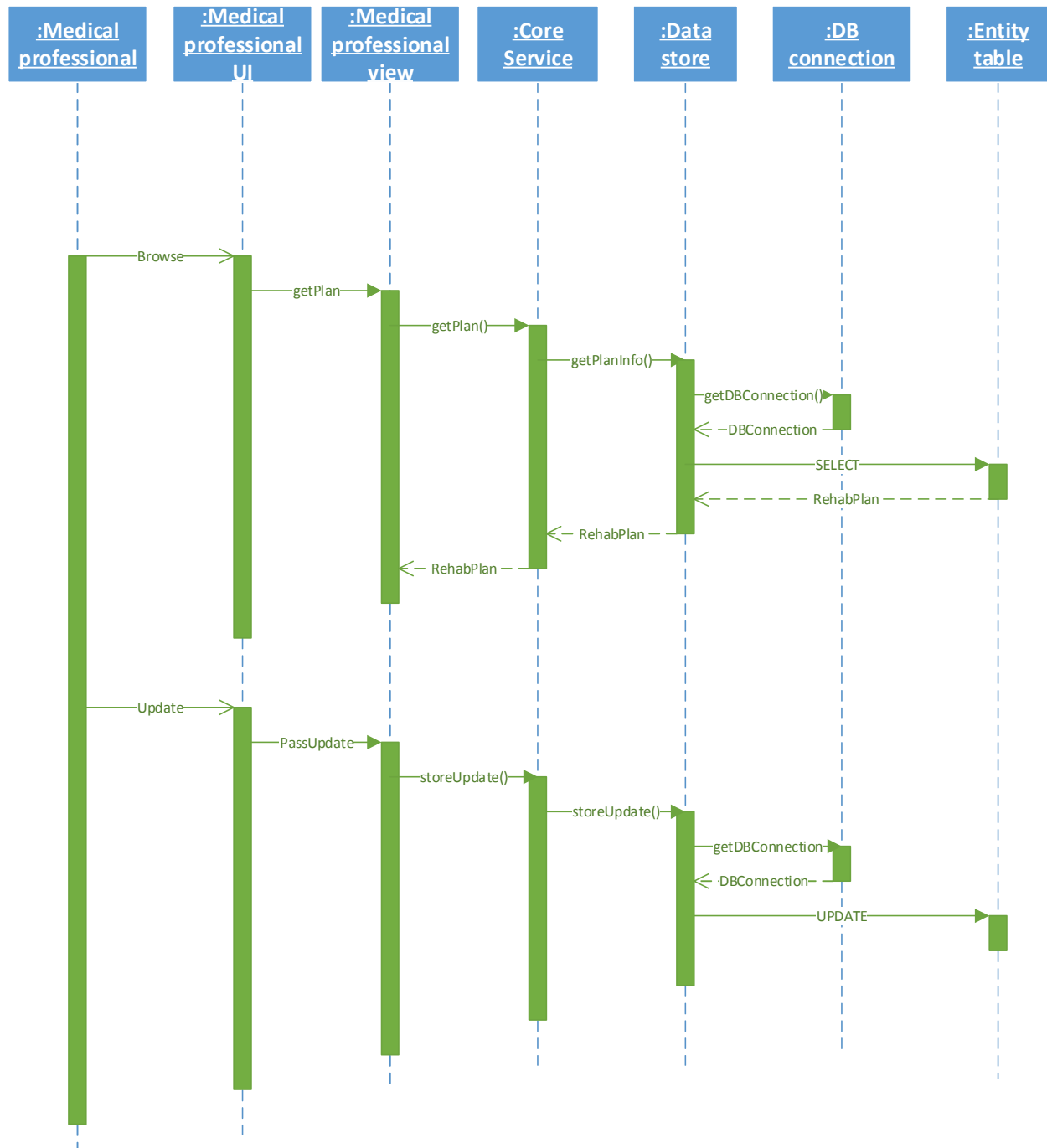
1. Medical Professional browses the Medical Professional UI to read the patient's health data and view the template for the plan or saves the plan he provides.
2. The Medical Professional view takes the command from user to display the patients' health data , template for the plan or save the plan given by the medical professionals
3. The Core Services handles the getPateintData(), getTemplate() and savePlan() method where the health data, rehab template of the patient is obtained or the rehab plan is stored
4. The support services subsystem provides the health data and template data for the core services to process.

5. The Data Store subsystem gets the health data of the patient, the rehab plan template stored in the database or saves the plan to database
6. The DB Connection subsystem helps the DataStore to get a connection instance to the database
7. The Data store then used the connection to select the patient data and the Template data from the database. The Datastore also uses the connection to save plan in the entity tables.



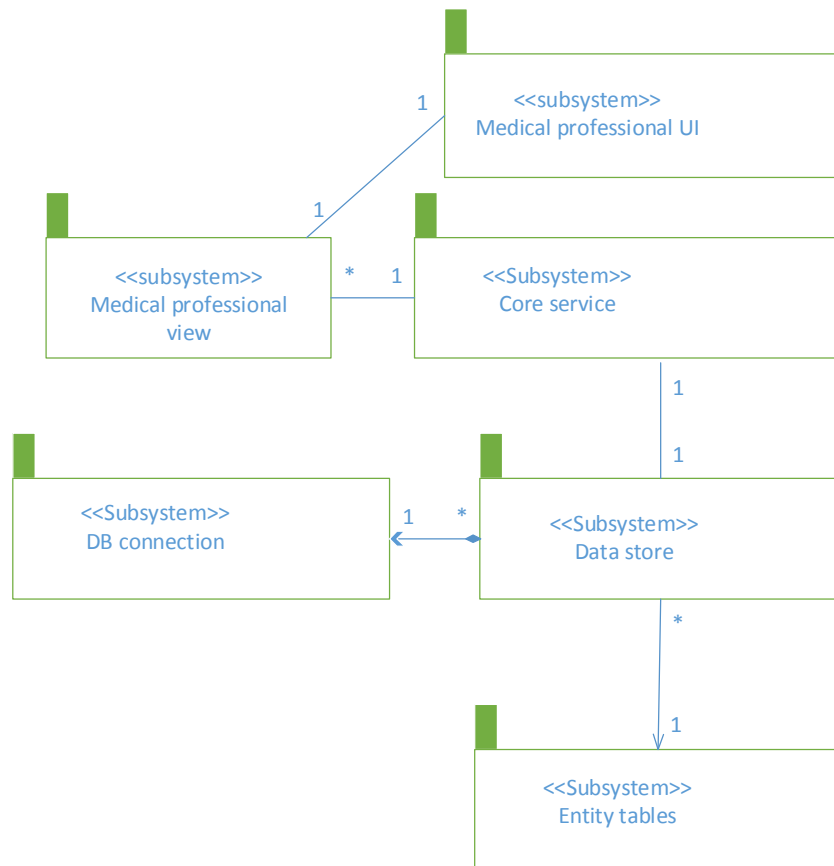
8. Each Medical Professional UI subsystem will be supported by one Medical Professional View subsystem.
9. Each Core Services subsystem will provide services to many instances of Medical Professional View subsystem
10. Each core services may or may not use the instances of the support Service subsystem
11. Each instance of the core services subsystem will use one instance of the Data store subsystem
12. Each instance of data store subsystem will instantiate one instance of DB connection
13. Each instance of Entity tables subsystem can be accessed by any number of instances of datastore subsystem.

## 5.2 Manage Rehab Plan



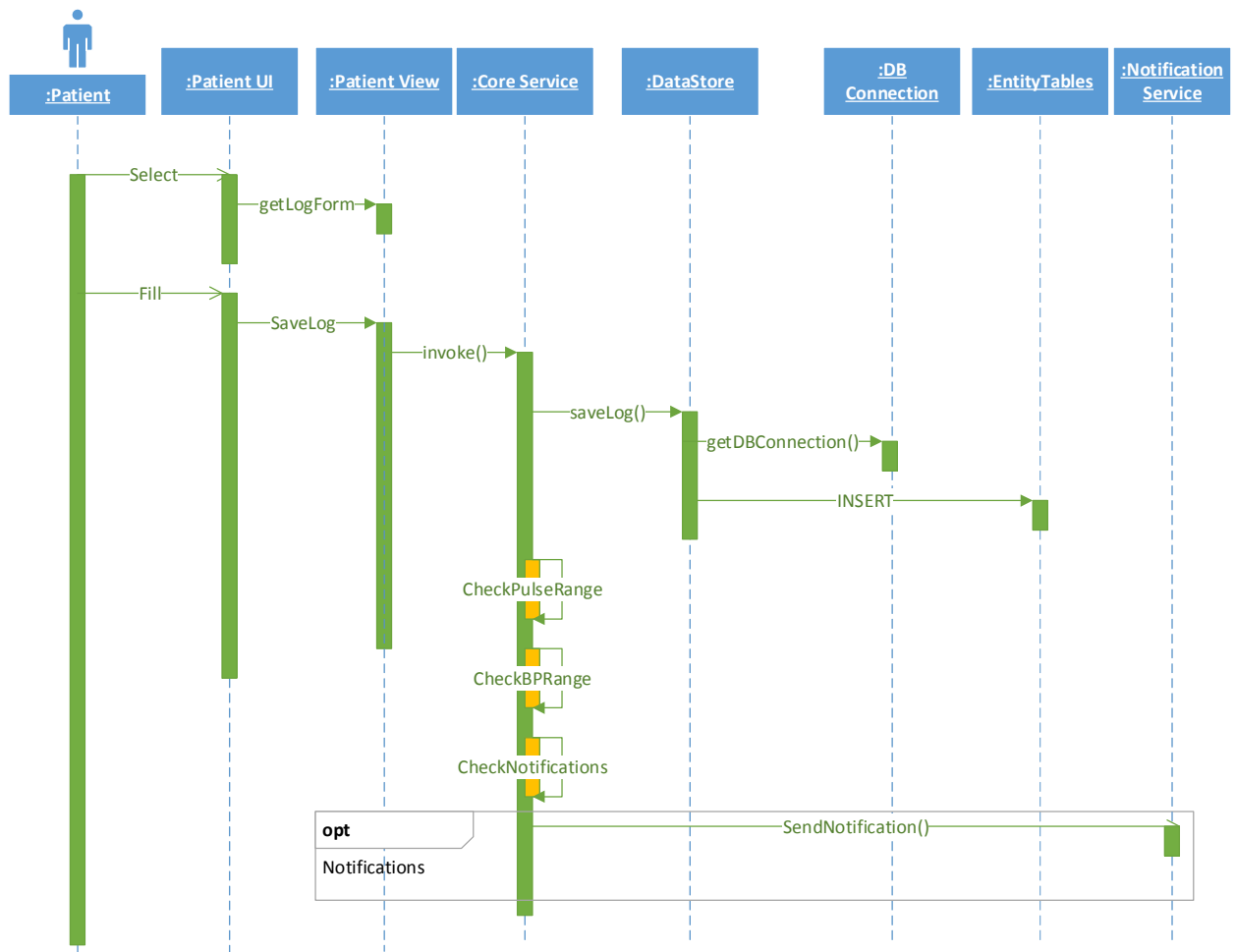
1. Medical Professional browses the Medical Professional UI to review the rehab plan or update the rehab plan.
2. The Medical Professional view takes the command from user to display the rehab plan or get the changes to the rehab plan.
3. The Core Services handles the `getPlan()` and `storeUpdate()` methods where the rehab plan is obtained and the updates are stored onto the plan respectively.
4. The Data Store subsystem gets the rehab plan of the patient from database and updates the rehab plan stored in the database.

5. The DB Connection subsystem helps the DataStore to get a connection instance to the database
6. The Data store then used the connection to select the rehab plan data and insert the rehab plan changes to the database. The Datastore also uses the connection to save plan in the entity tables.



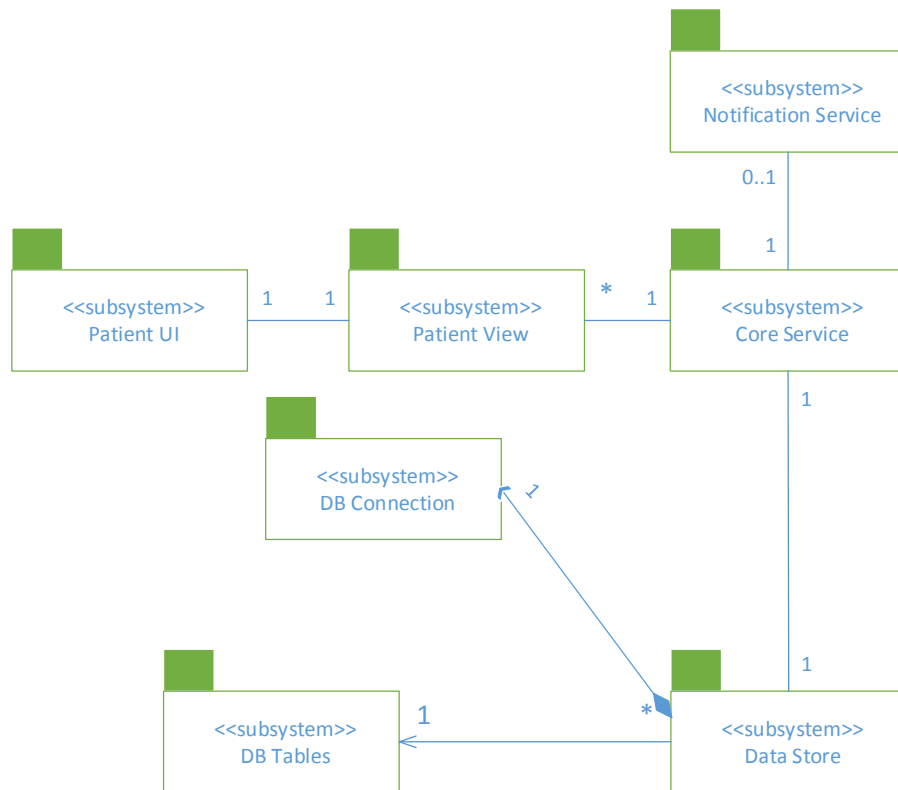
7. Each Medical Professional UI subsystem will be supported by one Medical Professional View subsystem.
8. Each Core Services subsystem will provide services to many instances of Medical Professional View subsystem
9. Each instance of the core services subsystem will use one instance of the Data store subsystem
10. Each instance of data store subsystem will instantiate one instance of DB connection
11. Each instance of Entity tables subsystem can be accessed by any number of instances of datastore subsystem.

## 5.3 Manage Rehab Log



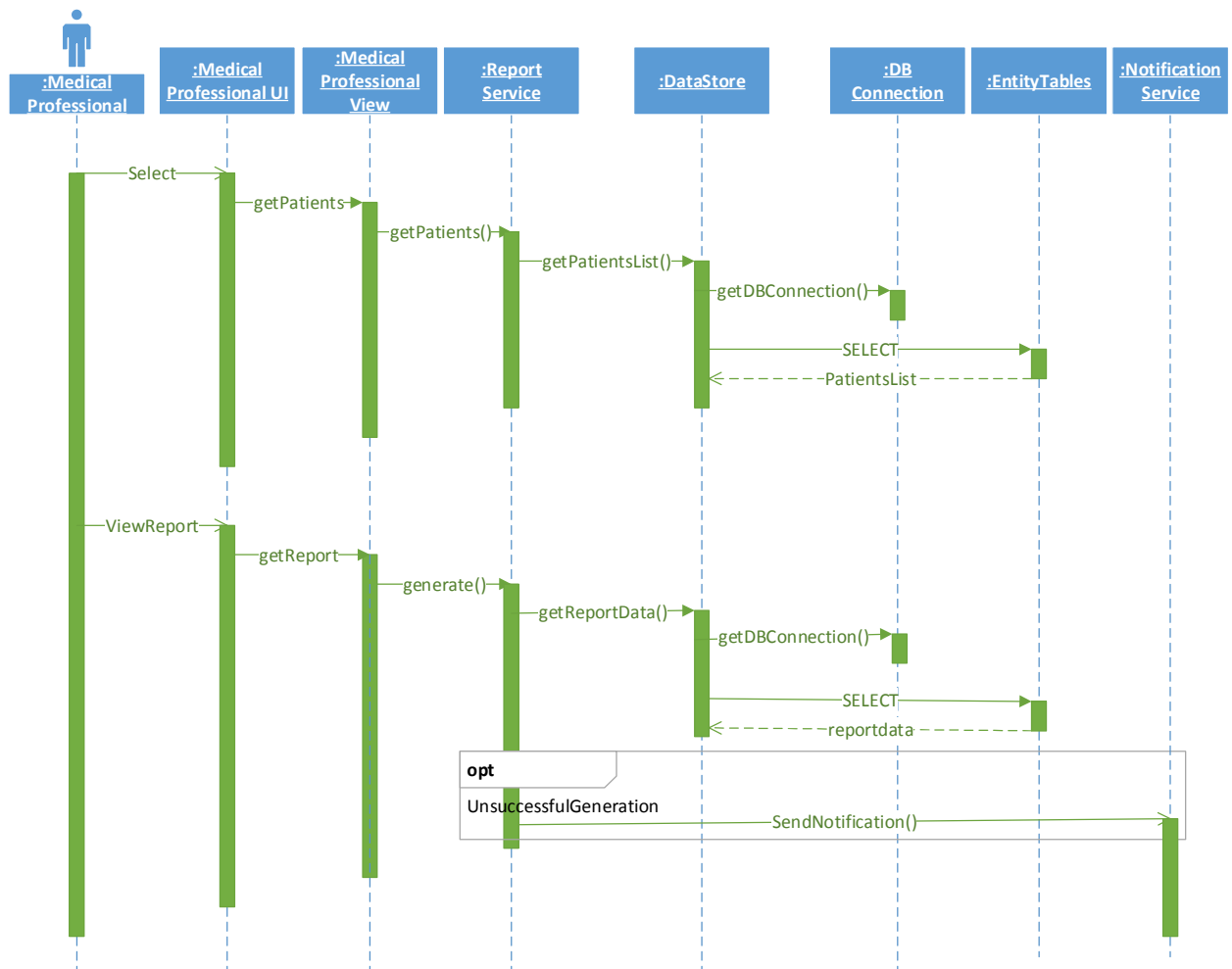
1. Patient selects the option to fill the Rehab Log on the Patient UI.
2. The Patient view takes the command from user to save the log that the user has filled. All the information is passed from the UI to the View.
3. The Core Services handles the invoke() methods where method to store the rehab log is invoked. The core services subsystem also checks the log input by user for any vitals that have exceeded the acceptable range and invokes the send Notification method of the notification services to send a notification to the doctor if vitals are not in range.
4. The Data Store subsystem saves the rehab log of the patient to the database.
5. The DB Connection subsystem helps the DataStore to get a connection instance to the database
6. The Data store then used the connection to save the rehab log changes to the database. The Datastore also uses the connection to save plan in the entity tables.



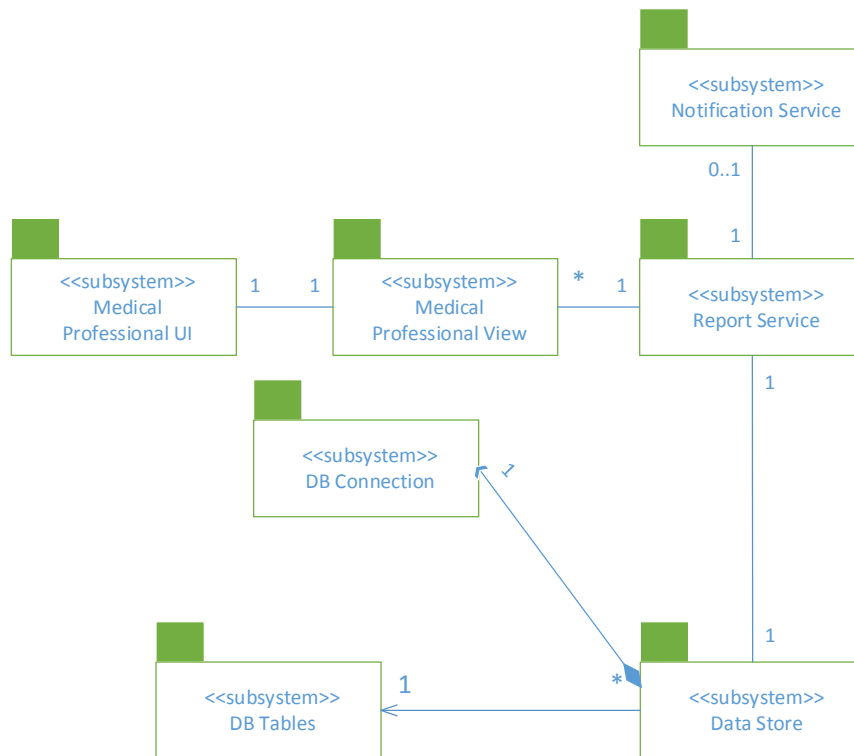


7. Each Patient UI subsystem will be supported by one Patient View subsystem.
8. Each Core Services subsystem will provide services to many instances of Patient View subsystem.
9. Each instance of the core services subsystem may or may not send notifications using the notification services subsystem.
10. Each instance of the core services subsystem will use one instance of the Data store subsystem
11. Each instance of data store subsystem will instantiate one instance of DB connection
12. Each instance of Entity tables subsystem can be accessed by any number of instances of dataStore subsystem.

## 5.4 Generate Progress Report

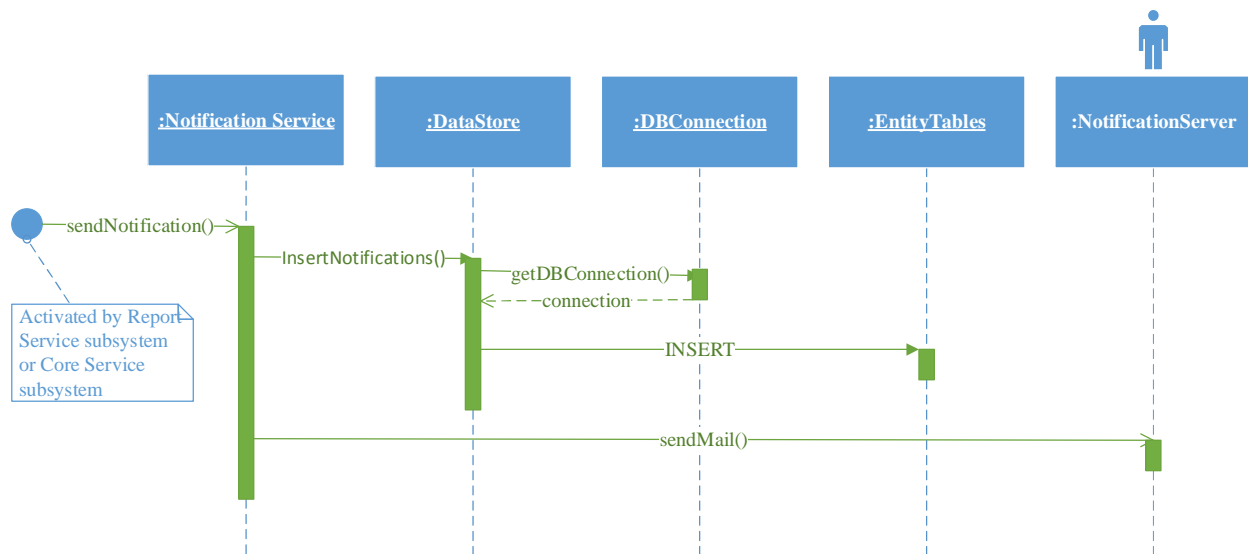


1. Medical Professional browses the Medical Professional UI to select the patient or view the report of the patient.
2. The Medical Professional view takes the command from user to display the patients list or display the report for the patient.
3. The report Services gets the list of patients using the `getPatientsList()` method or generates the report.
4. The Data Store subsystem gets the list of patients from the database or gets the data to be displayed in the report from the database for that particular patient
5. The DB Connection subsystem helps the DataStore to get a connection instance to the database
6. The Data store then used the connection to select the patient data and report data from the database. The Datastore also uses the connection to save plan in the entity tables.

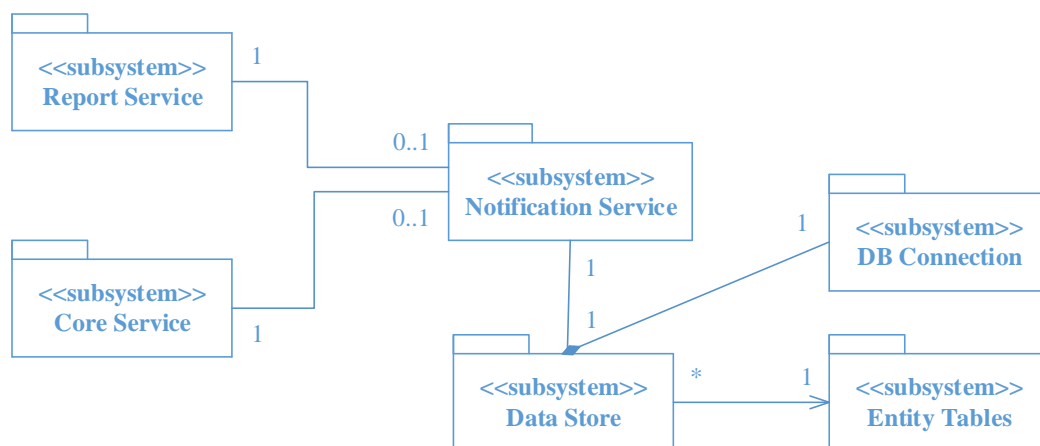


7. Each Medical Professional UI subsystem will be supported by one Medical Professional View subsystem.
8. Each reportServices subsystem will provide services to many instances of Medical Professional View subsystem
9. Each Report services subsystem may or may not send notifications using the instances of notification Service subsystem.
10. Each instance of the Report services subsystem will use one instance of the Data store subsystem
11. Each instance of data store subsystem will instantiate one instance of DB connection
12. Each instance of Entity tables subsystem can be accessed by any number of instances of dataStore subsystem.

## 5.5 Notify User

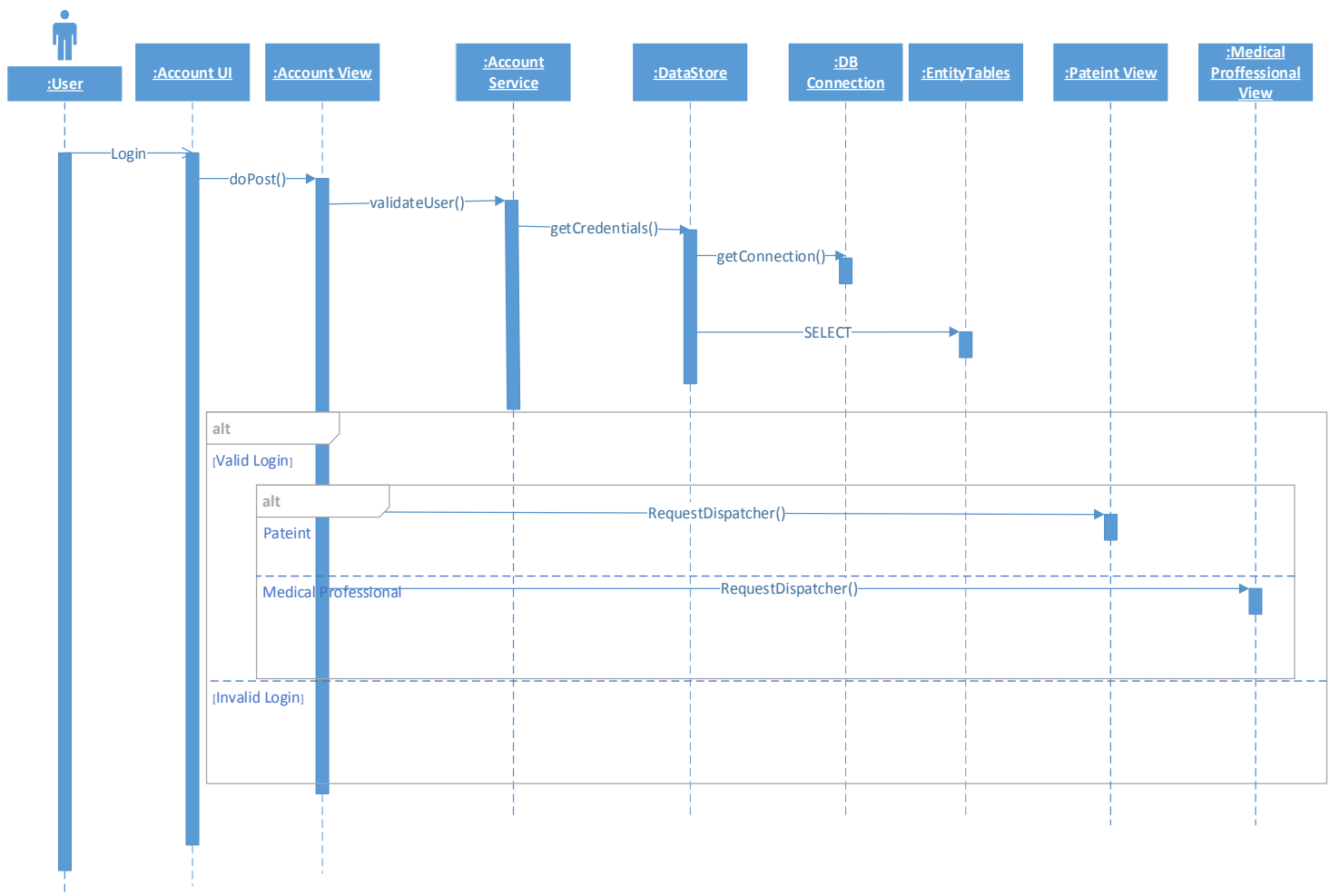


1. The Notification Service subsystem when invoked by the report services or core services subsystem, does manipulations to store the notifications in the database and also send mails to the doctor.
2. The Data Store subsystem gets the notification details and stores in the database.
3. The DB Connection subsystem helps the DataStore to get a connection instance to the database
4. The Data store uses the connection to insert the notifications into the entity tables in the database .

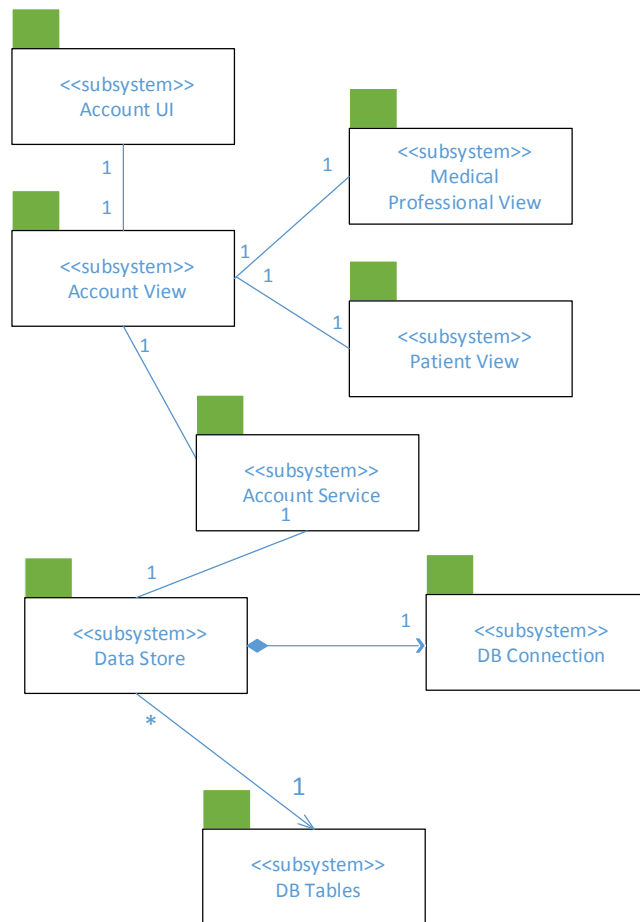


13. Each Report Service and Core Service instance may or may not send notifications using the instances of notification Service subsystem.
14. Each instance of the notification service subsystem will use one instance of Data store to store notifications
15. Each instance of data store subsystem will instantiate one instance of DB connection
16. Each instance of Entity tables subsystem can be accessed by any number of instances of dataStore subsystem.

## 5.6 Login

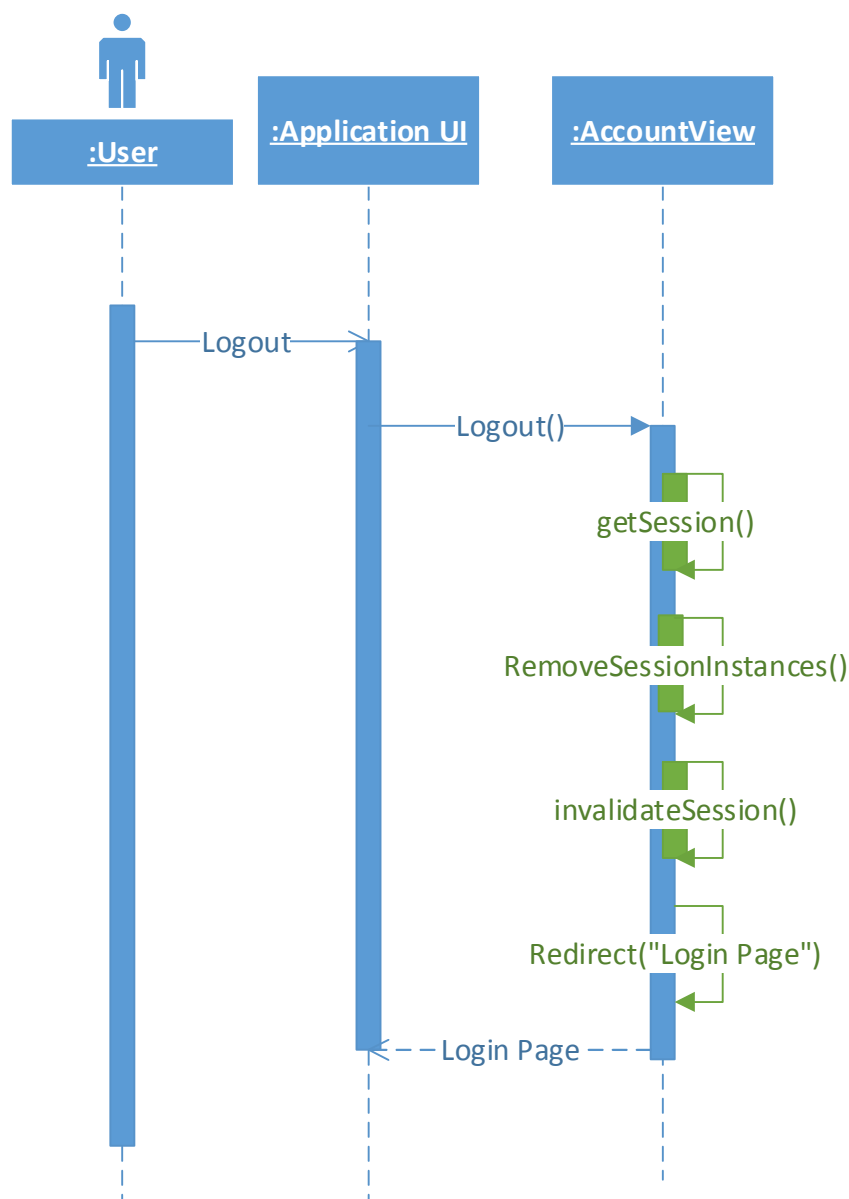


1. User enters URL into browser to access the application.
2. User is directed to Login page or Account UI.
3. User enters username and password and submits login information.
4. doPost() request is sent to Account Service to validate username and password
5. Account Service executes getCredentials() to get username and password from Database.
6. Database Connection is established.
7. Username and Password are selected from the Entity Tables.
8. Username and Password are returned and verified.
9. If the username and password are valid user is directed to Patient or Medical Professional Views based on user's role.
10. If the username and password are invalid user is directed back to the Login page.

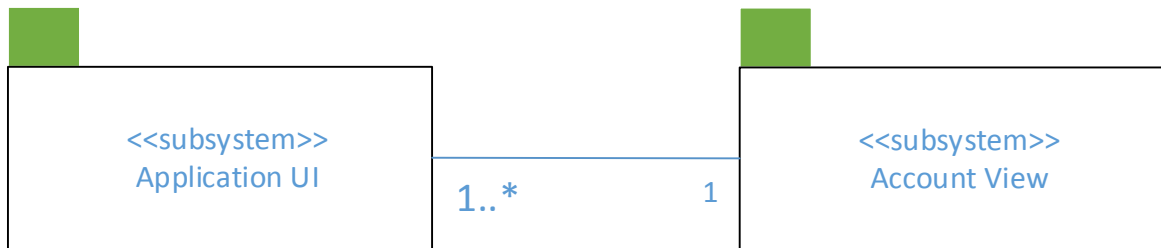


11. One user can initiate one session at a time and login if credentials are valid.
12. Multiple users can be logged in at a time.
13. One database connection can be created from account service to Data Store at a time.
14. The Data Store can access multiple DB tables.
15. The user can either be a Medical Professional or Patient depending on their role but not both.

## 5.7 Logout



1. User Clicks Logout link to initiate logout on Application UI.
2. AccountView executes `getSession()` to get the session of the user, `removeSessionInstances()` to remove all the temporary variables of the session and terminates the session by calling the `invalidateSession()`.
3. User is redirected to the Login or Account View Page.



4. The user interacts with the Application UI to click logout.
5. Account View terminates the session and user is directed to AccountView.
6. Each instance of Account view can be used by one or more instances of Application UI to terminate the session.

## 6. Subsystem Design

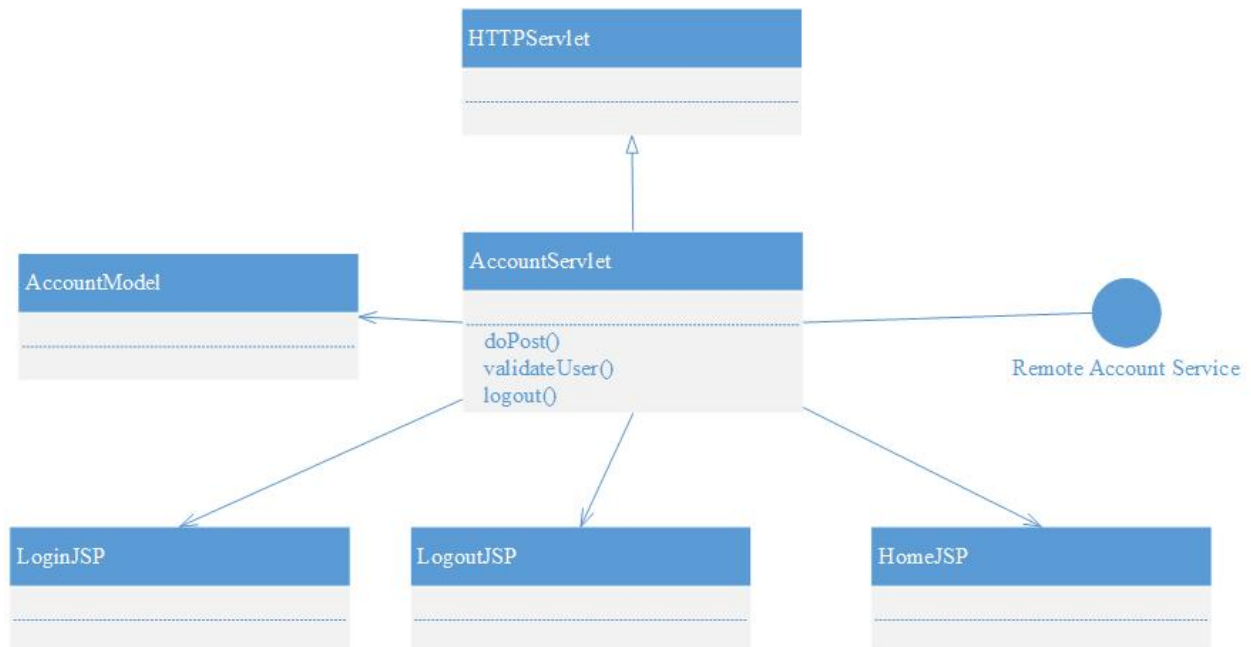
### 6.1 Application UI

Application UI Subsystem and its nested subsystems are described in detail in sections 7 and 8.

### 6.2 View Subsystems

View Subsystems contains presentation logic which helps in presenting UI to users. It contains servlets, jsp pages and model.

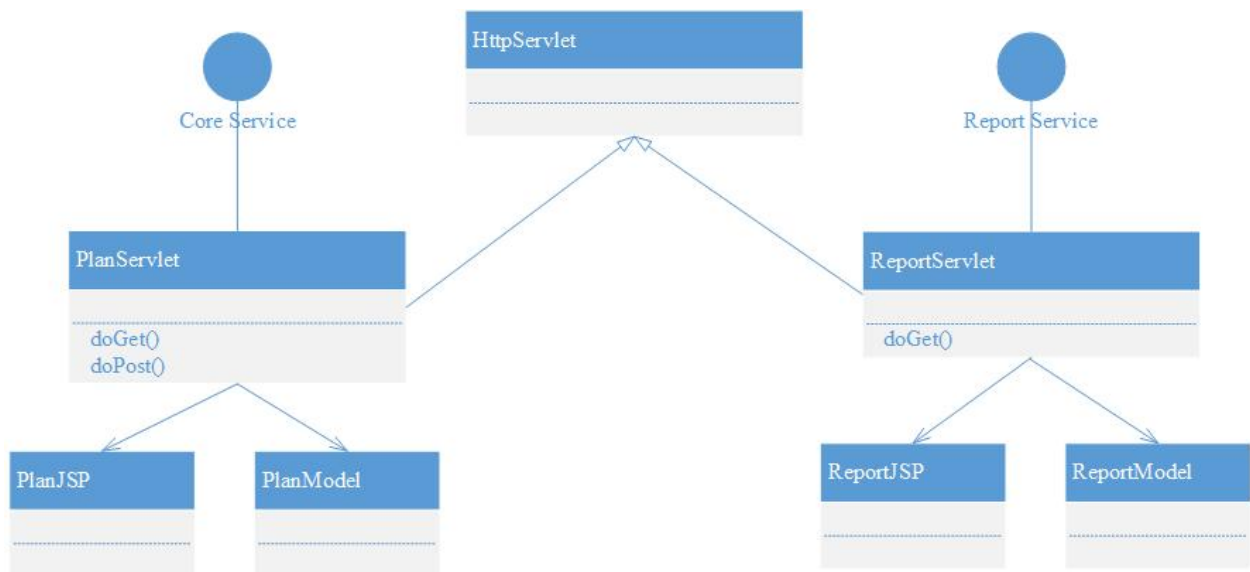
#### 6.2.1 Account View



Account View subsystem follows Page Controller Design Pattern. When user submits username password in LoginJSP page AccountServlet validates user using remote account service creates and AccountModel and redirects to HomeJSP according to AccountModel. It also creates session object for that user. If user validation fails AccountServlet redirects back to LoginJSP page. When user clicks on LogoutJSP, the AccountServlet destroys the user's session and redirects him to LoginJSP page.



### 6.2.2 Medical Professional View

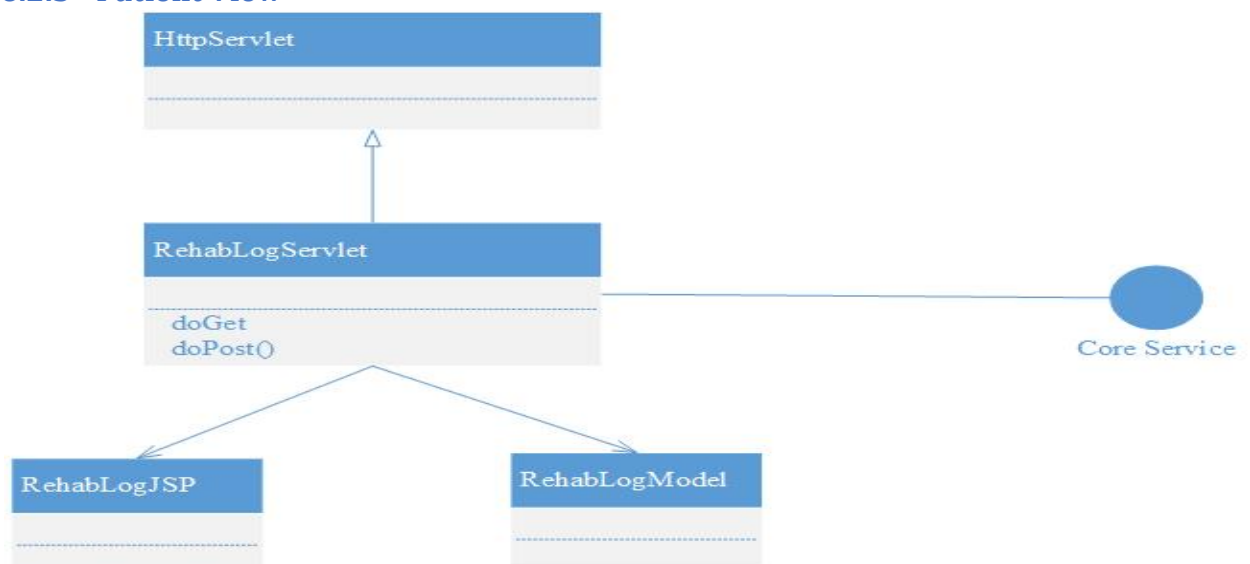


Medical Professional View subsystem consists of views related with Medical Professional. This subsystem follows Page Controller design pattern.

When an user wants to view a rehab plan `doGet` method of Plan Servlet will create `PlanModel` suing Remote Core Service and display `PlanJSP` view. When user wants to generate or update a rehab plan `doPost` method of `PlanServlet` will store or update the rehab plan using remote Core Service.

When user wants to view a patient's progress report, `doGet` method of `ReportServlet` will call remote `ReportService` and with the help of `ReportModel` redirect to `ReportJSP` page.

### 6.2.3 Patient View

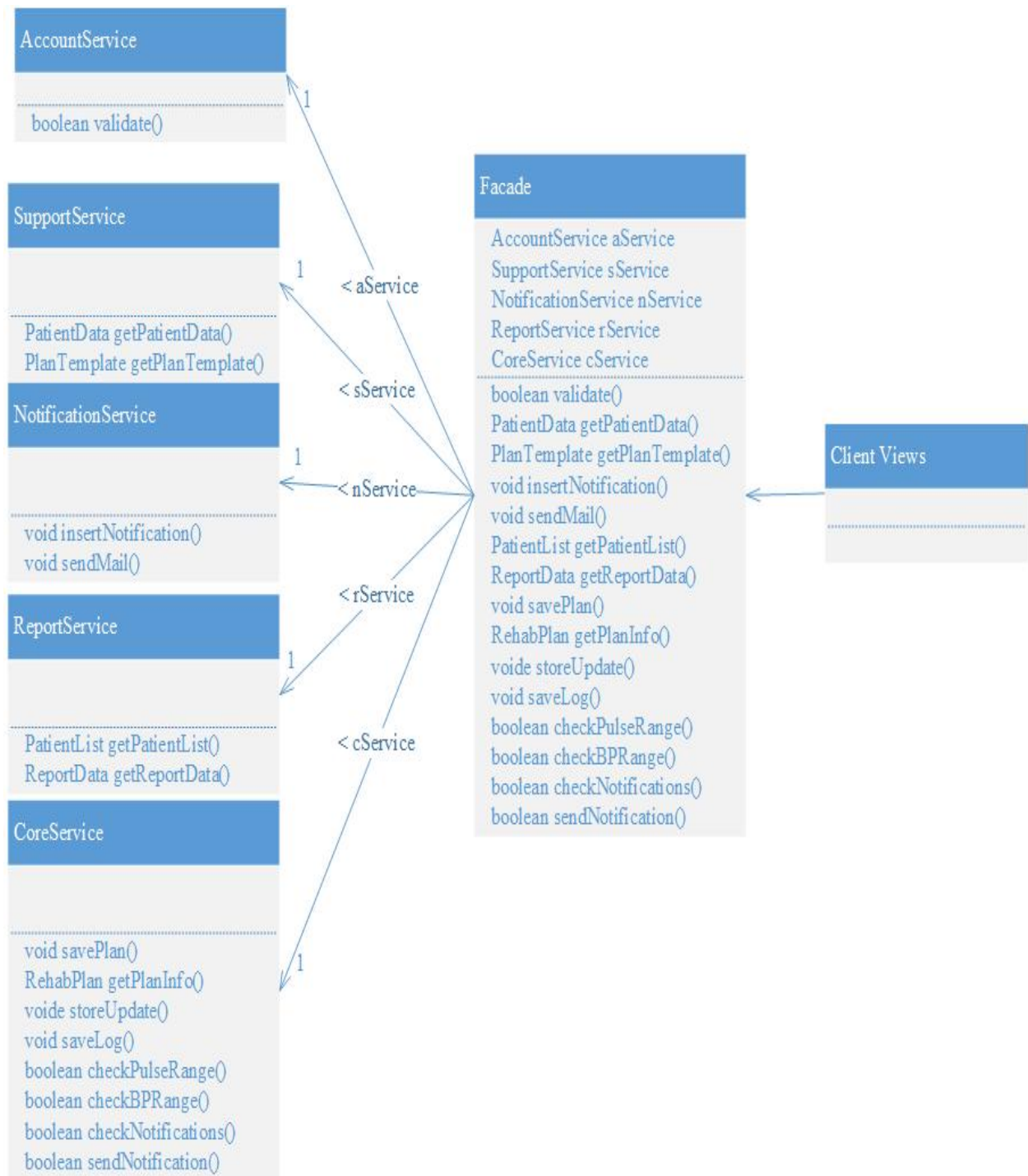


Patient View subsystem consists of views related with patients and it follows Page Controller design pattern. When patients enters a log entry `doPost` method of `RehabLogServlet` stores log data in database using remote core service. If Patient wants to view log enteries `doGet` method of

RehabLogServlet will get log entries from database using remote Core Service and with the help of RehabLogModel display RehabLogJSP.

### 6.3 Service Subsystems

Service subsystems contains the actual business logic of the application. These subsystems follow Facade design pattern where client will Facade for a particular functionality which in turn calls corresponding Service subsystem.



### 6.3.1 Account Service

AccountService subsystem contains validate method which passes username and password to database subsystems and returns the result to view subsystems.

### 6.3.2 Support Service

SupportService subsystem contains support methods which helps other services. It contains getPatienthealthData and getPlanTemplate methods which gets Patient Health Data and Plan Template which are used while creating a new rehab plan.

### 6.3.3 Notification Service

NotificationService subsystem contains methods to store notification in database using database subsystems and send a mail to user for that notification.

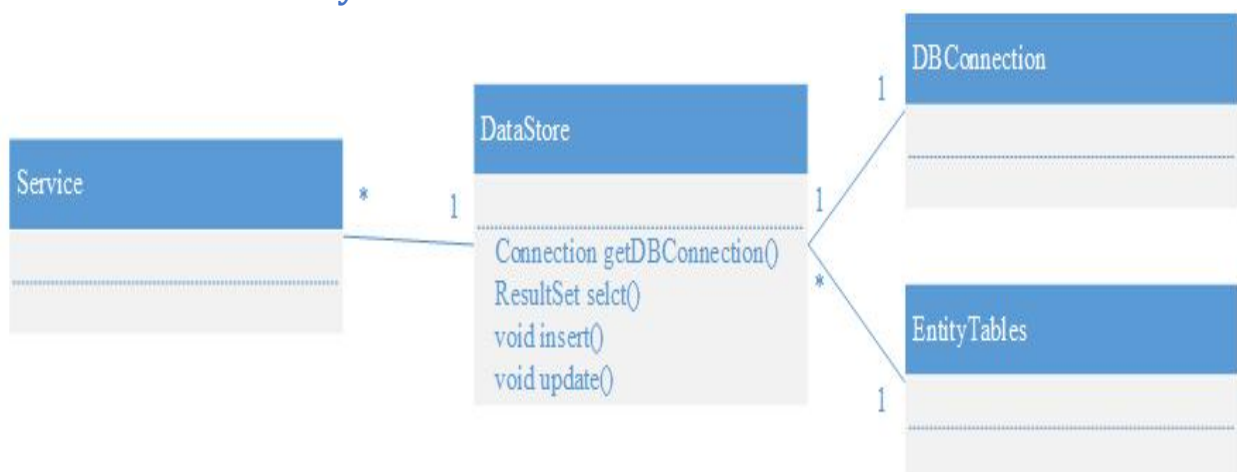
### 6.3.4 Report Service

ReportService subsystem contains methods to get patient list and get report for a patient in list using database systems and send it to view subsystems.

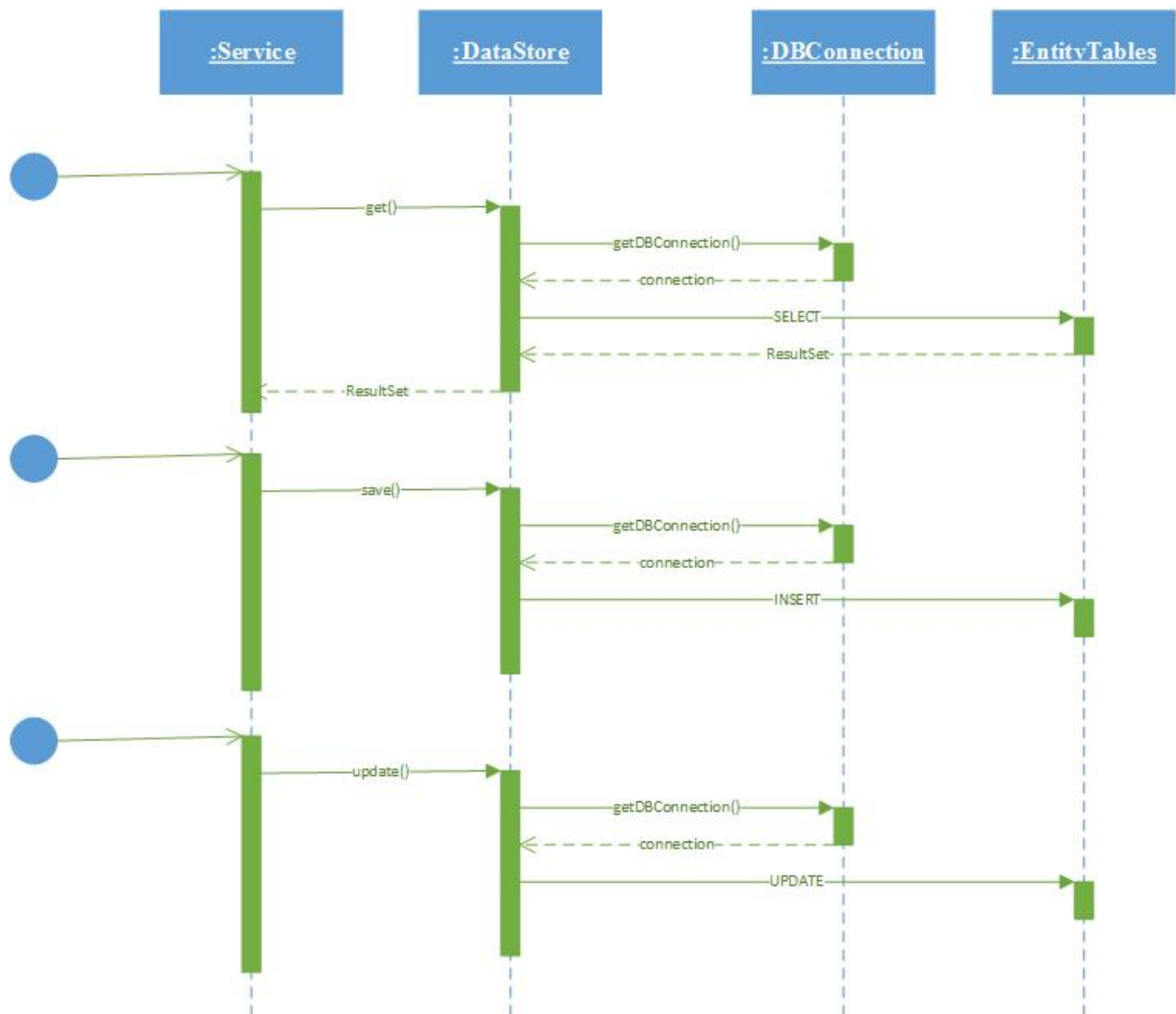
### 6.3.5 Core Service

CoreService subsystem contain methods which are related to core functionalities. It saves, gets and updates rehab plan using database subsystems. It also checks pulse range, BP range for patient's log entry. It then checks whether there is a need to send notification. If yes then it calls notification service to send notification and then stores the daily log entry in database using database subsystems.

## 6.4 Database Subsystems



Database subsystems follow Table Data Gateway design pattern and implements all actions related with database. Any service subsystem that want to interact with database contacts DataStore subsystem. The DataStore subsystem gets DBConnection from DBConnection subsystem and uses this connection to perform tasks on database entity tables.



#### 6.4.1 DataStore Subsystem

DataStore subsystem acts as Table Data Gateway and acts as sole point of contact between service and database tables. This systems gets a connection and uses SELECT, INSERT and UPDATE queries.

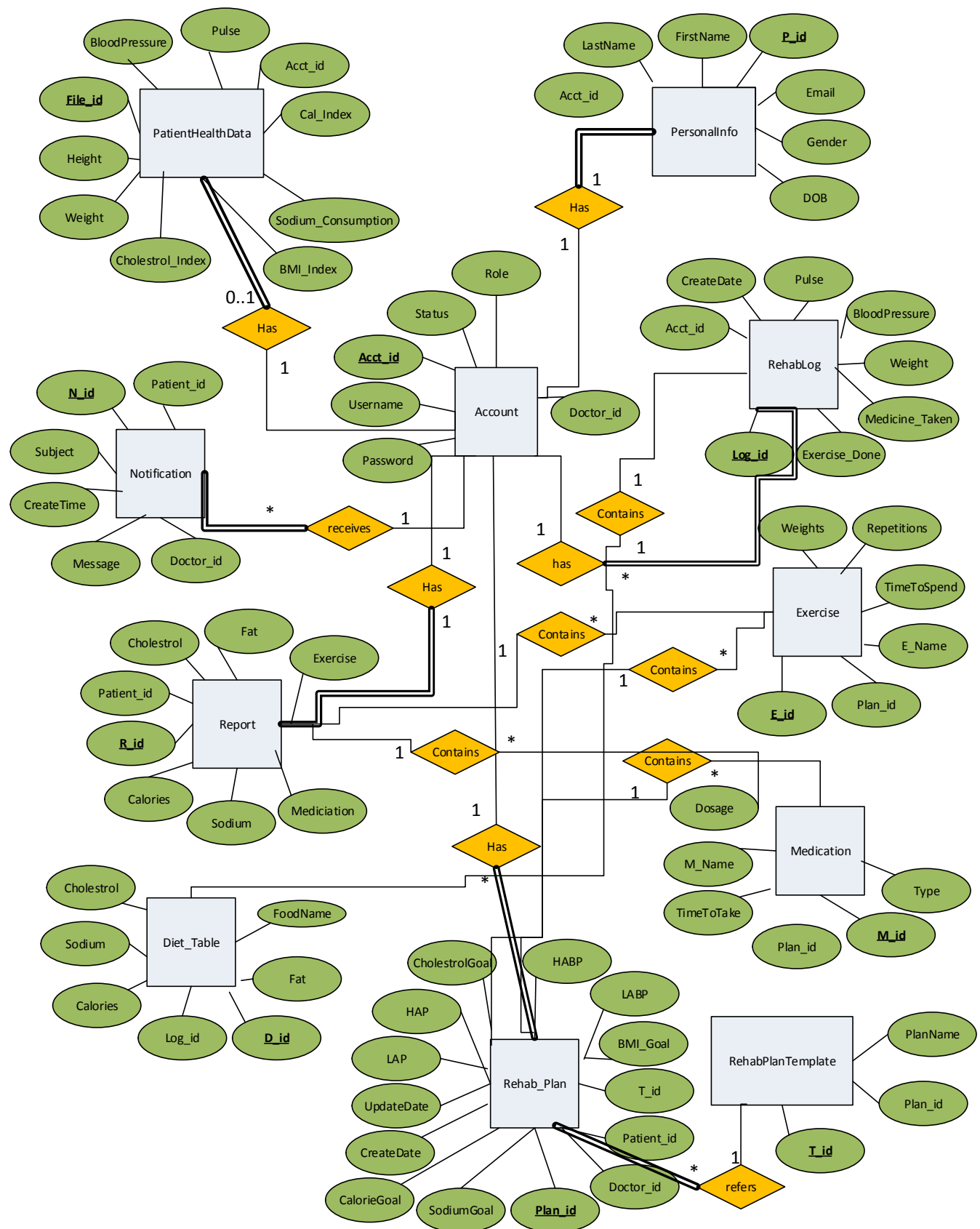
#### 6.4.2 DBConnection Subsystem

DBConnection subsystem creates a database connection which can be used to interact with database tables.

#### 6.4.3 EntityTables Subsystem

EntityTables subsystem consists of all entity tables that are required in application. This system have an Account table which stores account information of Medical Professional and Patients. It has one to one relationship with PersonInfo table which stores personal information about the user. Each record in PersonInfo table must be in the Account table. The subsystem also have Notification table which stores all the notifications for a user which will be displayed on users home page. Notification Table has many to one relationship with Account table where many notifications can be related to a single account. Each notification must also be related to an account in the account table. Account table also has one to one relationship with PatientHealthData which stores patient's previous medical records. Each account in account

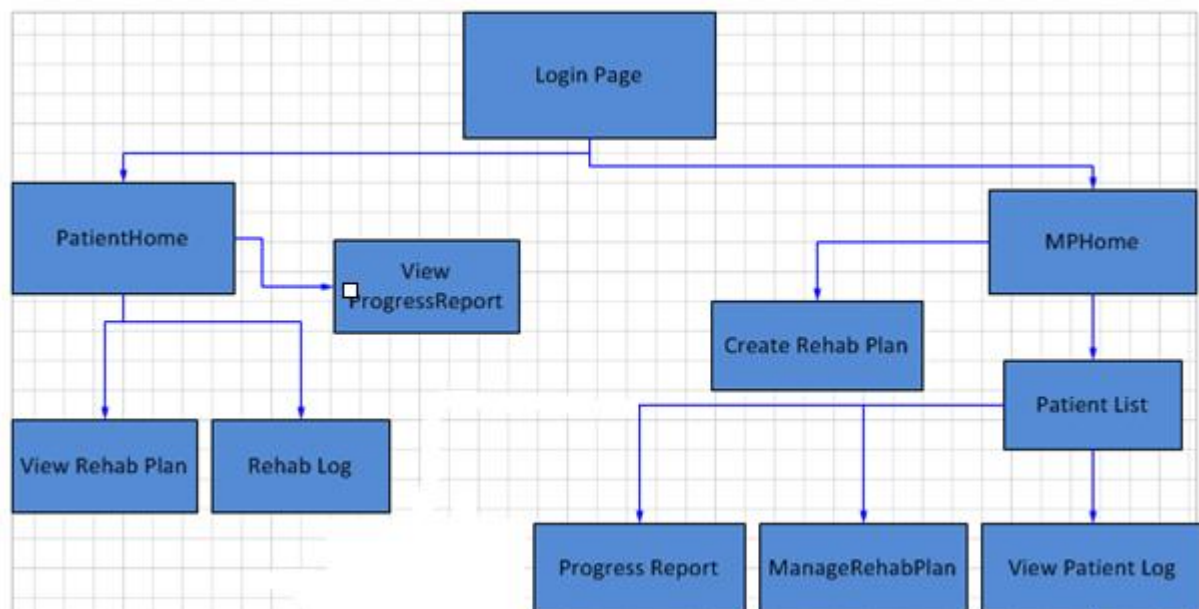
table may not have a record in Patient Health data table if the account is of doctor type. Each account in patient health data must be in the account table.



The subsystem also has Report table which has one to one relationship with Account Table. Each Patient Account will have a report. Patient Accounts stored in Account table will have one to many relationship with RehabLog table which stores log entries of patients. Each account can have many log entries. Every log in rehab log table will be related to a record in account table. Log Table also contains Diet Table in one to many relationship which stores diet taken by the patient. Rehab Plan Table is used to store rehab plan details. This table has one to many relationship with Exercise and Medication tables which stores required exercise and medication requirements respectively for a particular plan. The subsystem has Rehab Plan template table and has a plan in Rehab plan table which acts as a template to other plans. These template plans cannot be updated by user. Every plan in rehab plan table will be related to one of the templates in rehabplantemplate table.

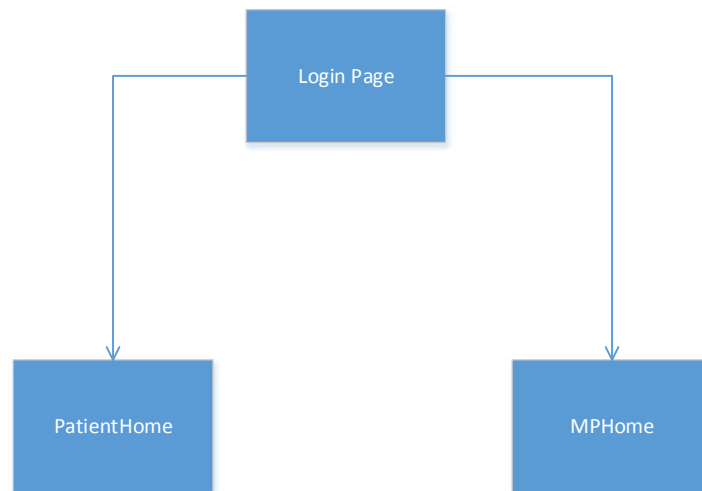
## 7. Use-Case Storyboards

Each use case here is presented from the actor's perspective. The overview interface diagram is given below



The above interface diagram helps describe the different interactions and pages that are a part of the application. This outlines the basic flow and how a user would navigate between pages.

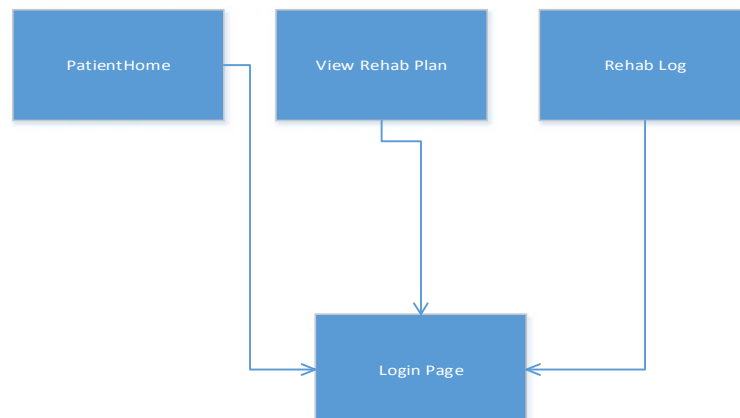
## 7.1 Login



1. User (Patient or Medical Professional) navigates to application by visiting URL and lands at the Login page.
2. User enters their credentials on the Login page and submits the information
3. User gets redirected to either MPHome or PatientHome page depending on their role if the User has given correct credentials
4. In case the User has not given correct credentials the Login Page displays a message saying so.
5. If the credentials are incorrect the splash page shows related message to the user.

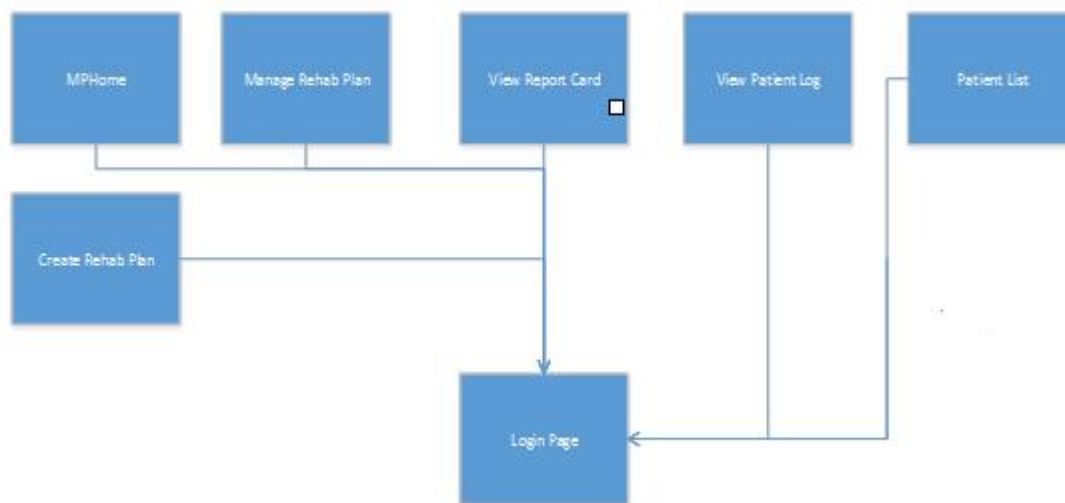
## 7.2 Logout

Patient:



Medical Professional:





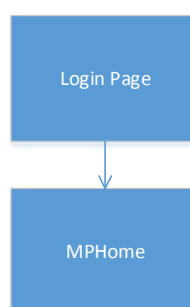
### Patient

1. The patient can be in one of the pages (i.e PatientHome, View Rehab Plan, Enter Daily Log) when he decides to log out and clicks on the logout button on the page.
2. Once the logout button is clicked, the application terminates user session
3. Redirected to Login Page for User

### Medical Professional

1. The medical professional can be in one of the pages (i.e MPHome, Manage Rehab Plan, Create Rehab Plan, View Report Card, View Patient Log, Patient List) when he decides to log out and clicks on the logout button on the page
2. Once the logout button is clicked, the application terminates user session
3. Redirected to Login Page for user to login

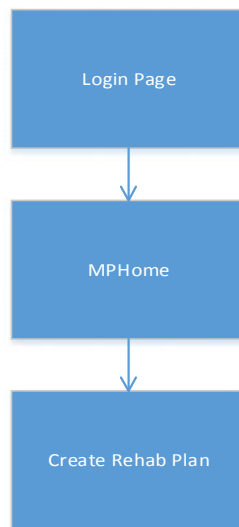
## 7.3 Notify User



1. The notification server sends the notifications as a result of the Medical Professional facing issues in generating a report, when the Patient enters vitals in the log which exceed the accepted range of values
2. The Medical Professional can login to the application using the Login page
3. After launching the home page, Medical Professional can view all the notifications sent by the notification server.

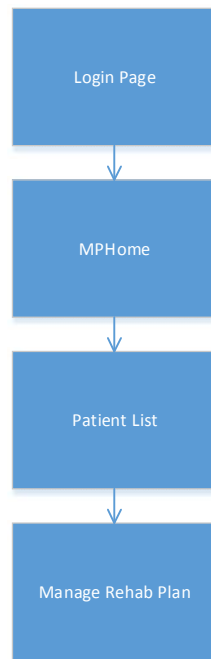


## 7.4 Generate Rehab Plan



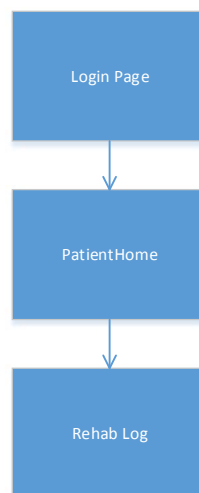
1. Medical Professional completes the “Login” use case from the Login page and lands on the MPHome page.
2. Medical Professional may have a notification on their home page that they have a new patient assigned to them.
3. Medical Professional clicks on “Create Rehab Plan” link on their new patient notification.
4. Medical Professional is redirected to the “Create Rehab Plan” page for the selected patient.
5. A rehab plan template along with the Patient’s Health data will be available to the Medical Professional.
6. Medical Professional reviews and completes the rehab plan by filling out all fields present on the page.
7. If additional exercises are required, Medical Professional can click “Add New Exercise” to be presented with another line of exercise fields.
8. If additional medications are required, Medical Professional can click “Add New Medicine” to be presented with another line of medication fields.
9. Once all fields are filled out, medical professional clicks “Save Plan” to save their changes.

## 7.5 Manage Rehab Plan



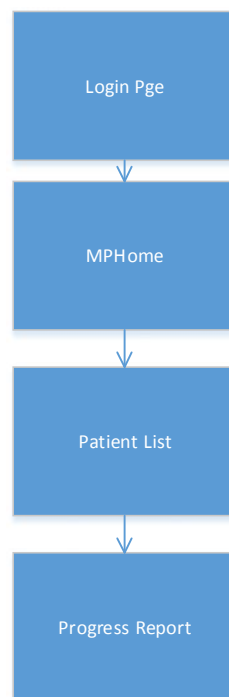
1. Medical Professional completes the “Login” use case and launches the MPHome page
2. Medical Professional clicks on “View Patient List” in the navigation menu.
3. Medical Professional is redirected to the Patient List.
4. Medical Professional finds the patient they are currently working with in the patient list.
5. Medical Professional clicks the “Manage Rehab Plan” link next to their chosen patient.
6. If the Plan does not already exist , the user is taken to the Create Rehab Plan page
7. If the plan already exists Medical professional is redirected to the “Manage Rehab Plan” page which is pre-filled with the existing rehab plan assigned to their chosen patient.
8. Medical Professional reviews and completes the rehab plan by filling out all fields present on the page.
9. If additional exercises are required, Medical Professional can click “Add New Exercise” to be presented with another line of exercise fields.
10. If additional medications are required, Medical Professional can click “Add New Medicine” to be presented with another line of medication fields.
11. Once all fields are filled out, medical professional clicks “Save Plan” to save their changes.

## 7.6 Manage Rehab Log



1. Patient completes the “Login” use case and lands on the PatientHome page.
2. Patient clicks on “Enter Daily Log” in the navigation menu
3. Patient is redirected to the “Rehab Log” page.
4. Patient fills out all fields to log their vital signs, diet, exercise, and medication.
5. If additional food items are required, Patient can click “Add New Food” to be presented with another line of dietary fields.
6. Once all fields are filled out, patient clicks “Save Log” to save their changes.
7. Once the log is saved, the application internally checks for any vitals that are not in acceptable range and sends a notification to the medical professional if any such vitals exist.

## 7.7 Generate Progress Report



1. Medical Professional completes the “Login” use case.
2. Medical Professional clicks on “View Patient List” in the navigation menu on the MPHome page

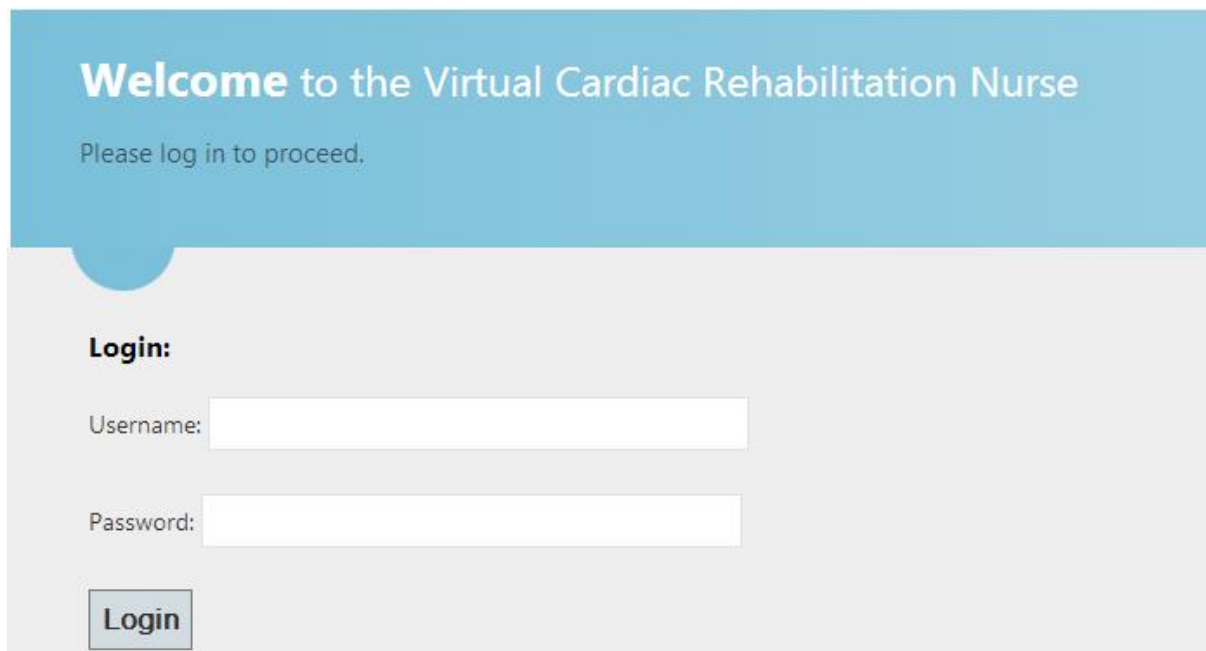
3. List of Patients are displayed on the Patient List Page
4. Medical Professional finds the patient they are currently working with in the patient list.
5. Medical Professional clicks the “View Report Card” link next to their chosen patient.
6. Medical Professional is redirected to the “Progress Report” page.
7. Medical Professional reviews the overview of the patient’s status that is presented to them on the Progress Report.

## 8. Prototype User Interfaces

The below screenshots and narratives provide an overview of the flow of our application. These will be modified and might appear different based on our final implementation and design.

### 8.1 Login Page

This is the splash page which is loaded when the user accesses the URL. The user will enter their credentials and hit login in order to access the homepage and navigate around the application. The homepage that is displayed will depend on the role of the user who logs in.



The screenshot shows a login interface for the 'Virtual Cardiac Rehabilitation Nurse'. It features a light blue header with the title 'Welcome to the Virtual Cardiac Rehabilitation Nurse' and a prompt 'Please log in to proceed.'. Below the header, on a light gray background, is a 'Login:' section. This section contains two input fields: 'Username:' and 'Password:'. A 'Login' button is positioned below the password field.

### 8.2 Patient Home Page

This view displays the home page for a patient. In the top right corner is a logout button which will take them back to the splash page. The home page provides a summary of the last time they updated their daily log, any alerts that might exist from the medical professional and several navigation links which will take them to other views or page.



### 8.3 Medical Professional Home Page

This view displays the home page for a medical professional. They can view a list of all patients that belong to them, create a new rehab plan, view patient report cards, and view any alerts the patient might have sent to them. This is where they can navigate to other pages.



### 8.4 Create Rehab Plan Page

This page is where the medical professional can create a new rehab plan. They will select which template to use and then put in information based on the patient's health data. This plan will be used by the patient to fill out daily logs and track their recovery process. There are fields to capture the vital signs and diet intakes as well as to document and capture exercises.



# Virtual Cardiac Rehabilitation Nurse

Logged in as ssmith. [Logout](#)

Select Rehab Plan Template To Use: Baseline Plan

## Create Rehab Plan For Patient John Smith

**Vitals**

Lowest Acceptable Pulse:

Highest Acceptable Pulse:

Lowest Acceptable Blood Pressure:

Highest Acceptable Blood Pressure:

**Diet**

Calorie Goal:

Fat Goal:

Sodium Goal:

Cholesterol Goal:

Exercise			
Exercise Name	Time To Spend	Weight	Repetitions
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

This view shows the three different rehab plan templates that can be selected for a patient. The values in the table will display based on the type of template selected. The template should be selected based on the patient's health data. If the rehab plan needs changing the medical professional can manage the rehab plan.



# Virtual Cardiac Rehabilitation Nurse

Logged in as ssmith. [Logout](#)

[Home](#) [View Patient List](#) [Create Rehab Plan Template](#)

Select Rehab Plan Template To Use: Baseline Plan

## Create Rehab Plan For Patient John Smith

**Vitals**

Lowest Acceptable Pulse:

Highest Acceptable Pulse:

Lowest Acceptable Blood Pressure:

Highest Acceptable Blood Pressure:

**Diet**

Calorie Goal:

Fat Goal:

Sodium Goal:

Cholesterol Goal:

Exercise			
Exercise Name	Time To Spend	Weight	Repetitions
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

This below screenshot shows the information that is being captured related to exercises and medicines that the patient should take. These will be saved to the rehab plan for the patient to follow. This is part of the overall rehab plan.

Exercise			
Exercise Name	Time To Spend	Weight	Repetitions
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<a href="#">Add New Exercise</a>			
Medicine			
Medicine Name	Time To Take	Dosage	Type
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<a href="#">Add New Medicine</a>			
<a href="#">Save Plan</a>			

## 8.5 Manage Rehab Plan Page

The medical professional can login and manage rehab plans once they have been created. They can alter the values or the rehab plan based on the patients daily logs or report cards. If any alerts are triggered to the doctor they can manage the rehab plan by clicking the “Manage Rehab Plan” link.



Virtual Cardiac Rehabilitation Nurse

Logged in as ssmith. [Logout](#)

[Home](#) [View Patient List](#) [Create Rehab Plan Template](#)

Current Patients			
Steve Smith	<a href="#">Manage Rehab Plan</a>	<a href="#">Send Notification</a>	<a href="#">View Report Card</a>
Joe Fakename	<a href="#">Manage Rehab Plan</a>	<a href="#">Send Notification</a>	<a href="#">View Report Card</a>
Mike Jones	<a href="#">Manage Rehab Plan</a>	<a href="#">Send Notification</a>	<a href="#">View Report Card</a>

© 2013 - Virtual Cardiac Rehabilitation Nurse



## Manage Rehab Plan For Patient John Smith

### Vitals

Lowest Acceptable Pulse:

Highest Acceptable Pulse:

Lowest Acceptable Blood Pressure:

Highest Acceptable Blood Pressure:

### Diet

Calorie Goal:

Fat Goal:

Sodium Goal:

Cholesterol Goal:

### Exercise

Exercise Name	Time To Spend	Weight	Repetitions
<input type="text" value="Walking"/>	<input type="text" value="30 minutes"/>	<input type="text" value="N/A"/>	<input type="text" value="N/A"/>
<input type="text" value="Yoga"/>	<input type="text" value="30 Minutes"/>	<input type="text" value="N/A"/>	<input type="text" value="N/A"/>

[Add New Exercise](#)

### Medicine

Medicine Name	Time To Take	Dosage	Type
<input type="text" value="Warfarin"/>	<input type="text" value="8am"/>	<input type="text" value="2.5mg"/>	<input type="text" value="Anticoagulant"/>
<input type="text" value="Propranolol"/>	<input type="text" value="8am, 2pm, 8pm"/>	<input type="text" value="10mg"/>	<input type="text" value="Antidysrhythmic"/>
<input type="text" value="Atorvastatin"/>	<input type="text" value="8am"/>	<input type="text" value="10mg"/>	<input type="text" value="Antilipidemic"/>

[Add New Medicine](#)

[Save Plan](#)

## 8.6 Patient Rehab Log Page

The Patient Rehab log is where the patient will go into daily to fill out their progress as they follow the rehab plan. This will track their pulse, blood pressure, and weight as well as track if they completed their exercises, took their medications, and diet intake. This information will be used to generate a progress report.





## Daily Log For Patient John Smith

### Vitals

Pulse:

Blood Pressure:

Weight:

### Food Intake

Food

Calories

Sodium

Cholesterol

Fat

[Add New Food](#)

Vital signs, diet, exercise, and medication module of the Rehab Log

### Exercise

Exercise Name

Time To Spend

Weight

Repetitions

Walking

30 minutes

N/A

N/A

Yoga

30 Minutes

N/A

N/A

### Medicine

Medicine Name

Type

Time(s) Taken

Dosage

Warfarin

Anticoagulant

8am

2.5mg

Propranolol

Antidysrhythmic

8am, 2pm, 8pm

10mg

Atorvastatin

Antilipidemic

8am

10mg

[Save Log](#)

Additional food items module of Rehab Log

### Food Intake

Food

Calories

Sodium

Cholesterol

Fat

Onion Bagel

400

300

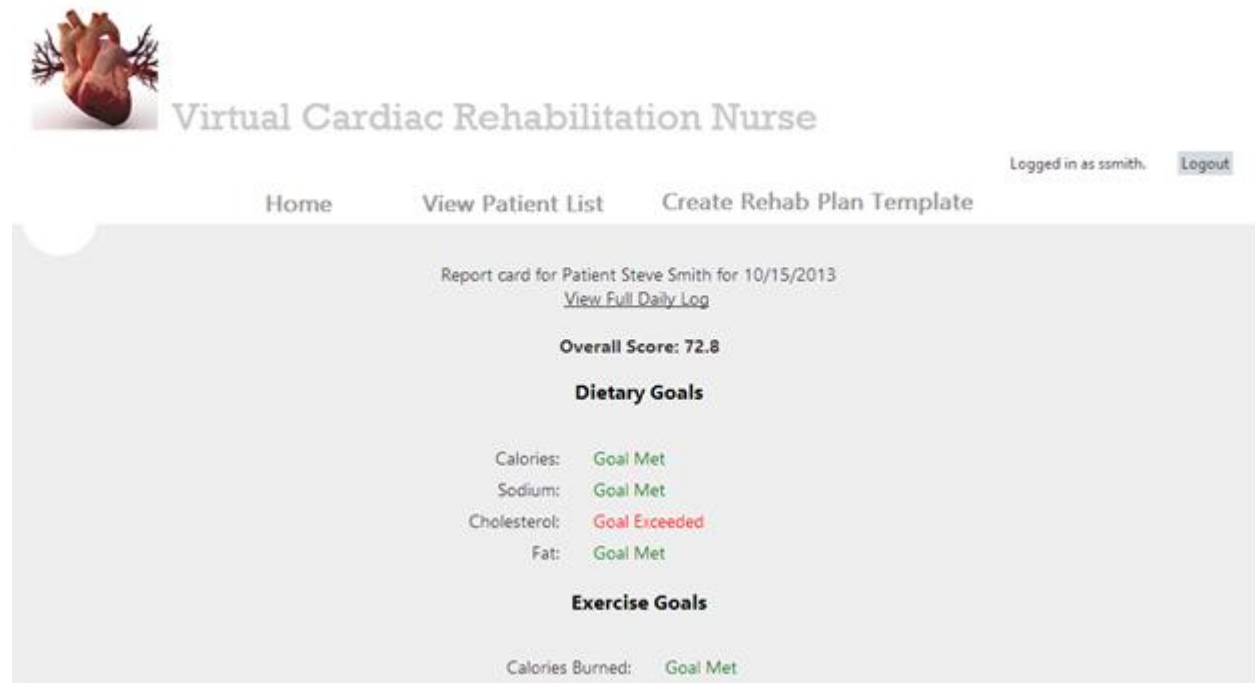
300

20

[Add New Food](#)

## 8.7 Progress Report Page

The progress report pulls from the information in the daily logs and provides a status as to how the patient is doing as they follow the rehab plan. It will display “Goal Met” or “Goal Not Met” based on the log the patient filled out.



## 9. System/Data Dependencies

These are the systems and data required to run the application.

### 9.1 Systems

Apache Tomcat Version 7.0 (Web and Application Server)  
Microsoft SQL Server 2005 (Database Server)  
Server - Windows 7  
Web Browsers (IE and Chrome capable of handling JavaScript , HTML 4.0 and CSS)  
JVM –Java version 1.6

### 9.2 Data

Patient Data  
User Personal Data

## 10. Appendices

### 10.1 Project Status

The “Design Specification 1 & 2” phase is complete. The project is on track and is 85% of the work is accomplished. The latest plan is uploaded as “Virtual Cardiac Rehabilitation Nurse Project Plan v1.9.mpp”