

DESIGN AND IMPLEMENTATION OF ORTHOPEDIC TRAUMA SURGERY
REHABILITATION AND HEALTH MONITORING SYSTEM (OTSRS)

A Thesis
Presented to
The Faculty of the Graduate School
University of Missouri-Columbia

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science

By
Ailiyasijiang Yilihamujiang
Dr. Yi Shang, Thesis Supervisor

APRIL 2016

The undersigned, appointed by the dean of the Graduate School, have examined the
thesis entitled

**DESIGN AND IMPLEMENTATION OF ORTHOPEDIC TRAUMA SURGERY
REHABILITATION AND HEALTH MONITORING SYSTEM (OTSRS)**

Presented by Ailiyasijiang Yilihamujiang

A candidate for the degree of Master Science

And hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Yi Shang

Dr. Jeffrey Uhlmann

Dr. Tim Trull

ACKNOWLEDGEMENTS

At the beginning of the program I clearly remembered that it is going to be a very demanding job to finish master degree in Computer Science. Looking back to these two and half years, I can confirm that it was a hard journey, but I learned a lot not only from the courses, but also from the professors, students and lab members that I worked with.

Therefore, I would like to thank my advisor Dr. Yi Shang for his continuous support and guidance throughout my program. We have regular meeting each Friday afternoon, and I learned so much from Him. Without his support, I would not have completed my thesis. I would also like to thank Dr. Jeffrey Uhlmann and Dr. Tim Trull for reviewing my thesis and providing so many insightful and valuable comments and suggestions. I appreciate them taking precious time out of their busy schedule to serve on my thesis committee. I would also like to thank Dr. David Volgas for providing me this amazing thesis topic. Our meetings and discussions made this thesis possible.

I am also very grateful to my colleagues in lab and friends. When I was in trouble, their support and comments were really insightful. Last but not least, I would like to thank my parents and relatives. Without their support I probably would not have started my master program in Mizzou.

Contents

ACKNOWLEDGEMENTS.....	ii
Contents.....	iii
ABSTRACT.....	vii
1. Introduction.....	1
2. Related Work.....	4
3. Design Theory and Methodology.....	6
3.1 System Architecture.....	6
3.2 Introduction to Key Technology.....	7
3.2.1 <i>What is OAuth?</i>	7
3.2.2 <i>What are the functionality of OAuth 2?</i>	7
<i>How OAuth 2 work?</i>	8
3.2.3 <i>Authorization Grant</i>	9
3.3 Android System.....	10
3.3.1 <i>Android System Architecture</i>	10
3.3.2 <i>Android Core Building Blocks</i>	12
4. System Design.....	14
4.1 Wearable sensors module.....	15
4.2 OAuth2 authentication module.....	16
4.3 Mobile computation module.....	16
4.3.1 <i>Background Task Module</i>	16
4.3.2 <i>Networking Module</i>	17
4.3.3 <i>Fitbit OAuth2 Module</i>	17
4.3.4 <i>iHealth OAuth2 Module</i>	18
4.3.5 <i>Dashboard Module</i>	18
4.3.6 <i>Exercises Module</i>	19
4.3.7 <i>Physician Module</i>	20
4.3.8 <i>In-app messaging Module</i>	21
4.3.9 <i>Account management Module</i>	21
4.3.10 <i>Progress Module</i>	21

4.3.11	<i>Link Account Module</i>	22
4.3.12	<i>Utility Module</i>	22
4.4	Web Server Module	22
4.4.1	<i>Account Management Module</i>	23
4.4.2	<i>Data Processing Module</i>	23
4.4.3	<i>Data Visualization Module</i>	23
4.4.4	<i>Protocol Management Module</i>	24
4.4.5	<i>Health Monitoring Module</i>	24
5.	System Implementation	25
5.1	OAuth2 authentication module	25
5.1.1	<i>Fitbit OAuth2 authentication</i>	25
5.1.2	<i>iHealth OAuth2 authentication</i>	26
5.2	Mobile computation module	27
5.2.1	<i>Background Task Module</i>	29
5.2.2	<i>Networking Module</i>	30
5.2.3	<i>Fitbit OAuth2 Module</i>	31
5.2.4	<i>iHealth OAuth2 Module</i>	33
5.2.5	<i>Dashboard Module</i>	35
5.2.6	<i>Exercises Module</i>	37
5.2.7	<i>Physician Module</i>	39
5.2.8	<i>In-app messaging Module</i>	41
5.2.9	<i>Account management Module</i>	42
5.2.10	<i>Progress Module</i>	44
5.2.11	<i>Link Account Module</i>	45
5.2.12	<i>Utility Module</i>	46
5.3	Web Server Module	47
5.3.1	<i>Account Management Module</i>	48
5.3.2	<i>Data Processing Module</i>	49
5.3.3	<i>Data Visualization Module</i>	51
5.3.4	<i>Protocol Management Module</i>	52
5.3.5	<i>Health Monitoring Module</i>	54
6.	Summary and Future Work	56
	References	58

LIST OF FIGURES

<i>Figure 1.1: Worldwide Smartphone OS Market Share from 2012 to 2015</i>	3
Figure 1.2: Percentage of website using server-side programming language	3
Figure 3.1 A typical collaboration of the MVC components [9]	6
Figure 3.2 OAuth 2 overall workflow	9
Figure 3.3 OAuth2 authorization code flow	10
Figure 3.4 Android System Architecture.....	11
Figure 3.5 Android Core Building Blocks.....	12
Figure 4.1 Overall System Architecture	14
Figure 4.2 Kinds of wearable sensors that the proposed system supports	15
Figure 4.3 ACL Reconstruction Rehabilitation Protocol.....	19
Figure 4.4 Rotator Cuff Repair Rehabilitation Protocol	20
Figure 5.1 Fitbit Application Credentials	26
Figure 5.2 Fitbit OAuth2 Configuration File	26
Figure 5.3 iHealth API credentials.....	27
Figure 5.4 Login Page	28
Figure 5.5 Register and Surgery Date Selection Page.....	28
Figure 5.6 AsyncTask Upload Image Class	29
Figure 5.7 Anatomy of an HTTP request.....	30
Figure 5.8 Fitbit authentication page	31
Figure 5.9 Fitbit OAuth 2.0 authentication flow	32

Figure 5.10 iHealth OAuth 2 authentication Flow	34
Figure 5.11 iHealth authentication page	34
Figure 5.12 Dashboard Page	36
Figure 5.13 Exercise details page.....	38
Figure 5.14 Survey page.....	39
Figure 5.15 Physicians page	40
Figure 5.16 In-app messaging page	41
Figure 5.17 Account information page	43
Figure 5.18 Forget password page.....	43
Figure 5.19 Progress page.....	44
Figure 5.20 Example of an XML layout	45
Figure 5.21 Link account page	46
Figure 5.22 OTSRS system admin website login page	47
Figure 5.23 OTSRS server update profile, register and forget password page	49
Figure 5.24 RIJNDAEL_128 cipher Block Chaining mode encryption	50
Figure 5.25 RIJNDAEL_128 cipher Block Chaining mode decryption	51
Figure 5.26 Dashboard patients' overall progress.....	52
Figure 5.27 OTSRS website patients' region map.....	52
Figure 5.28 Add protocol page	53
Figure 5.29 Review protocol page	54
Figure 5.30 Physician sending email page	55

ABSTRACT

A very common problem relating to rehabilitation of orthopedic trauma surgery is that patients do not have access, whether for financial reasons or physical location, to physical therapy and rehabilitation services after the surgery. In many cases, the exercises that physicians ask patients to perform are well understanding and can be performed at home; besides, some patients cannot remember what they were supposed to do and often lost the papers which showed them what to do and how to do the exercises. Patients also do not remember to do their exercise every day and physicians have no measure of their compliance.

This thesis presents an Android and Web server combined Rehabilitation System, with support of two large health monitoring wearable sensor company APIs (Fitbit and iHealth), for orthopedic trauma surgery, especially for rotator cuff repair and ACL reconstruction, to improve current methods and provide more convenient, reliable, secure approach not only for the patients to recover quickly but also for the physicians to monitor the patients' progress between office visits and home to determine whether they are progressing as expected.

Wearable devices combine with smartphones provide much more powerful health monitoring approach for physicians. This system consists of four major parts: wearable sensors module (Fitbit and iHealth) that collect personal health data, OAuth2

authentication module, mobile computation module and web server module. The android application is responsible for setting up API connections, collecting and showing patients' health data from the wireless wearable devices, interacting with the server to display each day exercises and generate survey, getting in-app message from the physician, comparing patients' progress with other patients and uploading data to the web server. The OTSRS web server is responsible for data receiving, handling, visualization, sending emails, logging surgery protocols, sending in-system messages, monitor patients health and rehabilitation progress. The system has been developed and waiting to be deployed in the field of study.

1. Introduction

Transparently embedded remote health care can become a new cost effective paradigm, which can solve most of the problems primarily centralized Health Care systems have [1]. Health rehabilitation and monitoring systems are gaining their significance as the fast-growing patient population increases demands for better health care. In many cases patients released from the hospitals still needs to access to either physical therapy or rehabilitation services and strongly advised to be under observation some period of time.

However, most of the methods in clinical surgery rehabilitation primarily rely on questionnaires, protocols and interviews with physicians, which have no capability of collecting and monitoring real-life personal health data, and physicians have no measure of patients' compliance of each days' exercises. Wearable and health devices are different innovative technologies such as smart glasses, activity trackers, smartwatches, blood pressure monitors, wireless scales, fitness devices, wireless glucometers and more. Essentially they include any smart devices that people can wear and use to monitor personal health. Monitoring and recording of different physiological parameters of patients in outside clinical environment is becoming increasingly important for physicians in order to provide patients with better health care [2]. By combining real-life health data from all kinds of wearable and health devices with self-reported feedbacks, physicians can then monitor the patients' progress between office visits and determine whether the patient is progressing as expected. This system have many advantages. First of all, it is a

convenient tool to keep patients engaged in ongoing rehabilitation plans. Patients have real-time access to health data, exercise instructions, and much more. They save time by not having to re-visit the office to pick up lost instructions papers or feedbacks from their physicians. Also, it provides efficiency and time-savings. Through the system physicians have immediate access to patient health information. Last but not least, it can reduce health care cost. Also, by extension, a system like this could be used for smoking cessation, behavioral therapy, diabetes control, etc.

This paper presents a rehabilitation and health monitoring system designed for orthopedic trauma surgery, especially for rotator cuff repair and ACL reconstruction, to improve current methods and provide more convenient, reliable, secure approach not only for the patients to recover quickly but also for the physicians to monitor the patients' progress between office visits and home to determine whether they are progressing as expected. The mobile side of the system is developed for android smartphone, which is responsible for collecting health data through the intergraded API connections, displaying connected protocol details, interacting with the patients and generating survey, assessing patients performance and grading their each day exercises, comparing particular patient's progress with other patients, in-app messaging between physicians, communicating with OTSRS server for uploading encrypted data and receiving instructions. In order to achieve better health monitoring, two APIs are intergraded using Oauth2.0 protocol, and user-friendly UIs, classes and functions are implemented with satisfying performance. Data

collected by the OTSRS application is uploaded to the server once it is triggered. The server will decrypt the health data and prepare the data for plotting.

As it can be inferred from the Figure 1.1 [3], Android dominated the smartphone market with a share of 82.8% in 2015. Because of the highest market share and open source feature of Android Software Development Kit, it is chosen as the platform to develop and implement the mobile side of the rehabilitation and health monitoring system for the mobile device in this thesis. It can also be seen from figure 1.2 [4], PHP is the most popular server side programming language, so the server side is developed and implemented using PHP.

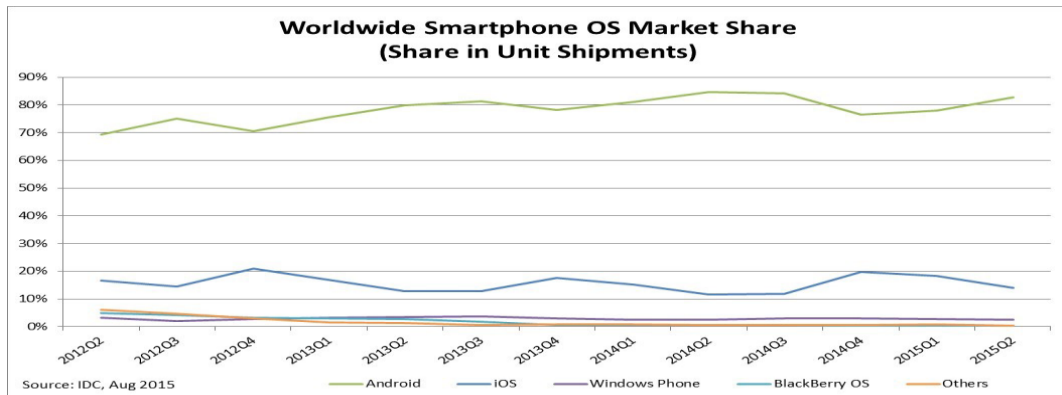


Figure 1.1: Worldwide Smartphone OS Market Share from 2012 to 2015

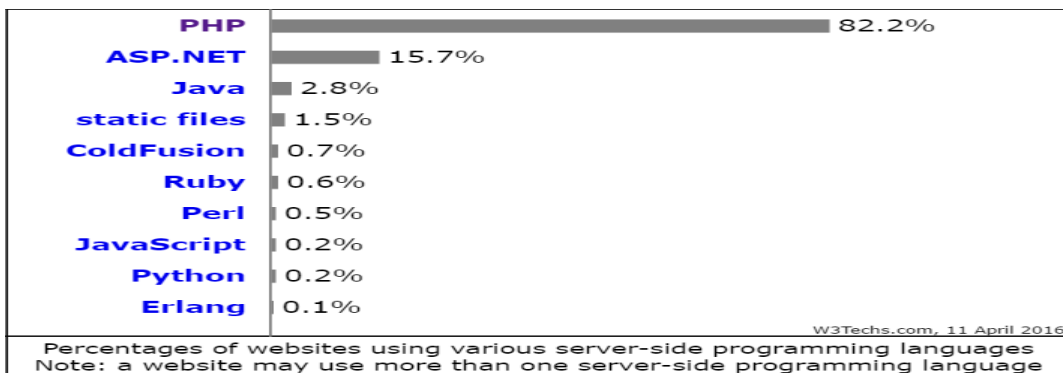


Figure 1.2: Percentage of website using server-side programming language

2. Related Work

Several systems have been developed in recent years to support out-clinic health monitoring and rehabilitation. Data acquisition on the personal mobile device enables both end-users and care givers to provide better and more effective health monitoring and facilitate prevention [1].

The system that proposed in this paper is closely related to the following three projects that presented in following papers: Android Based Control and Monitoring System for Leg Orthosis [5], Rehabilitation Exercises Feedback on Android Platform [6] and DroidGlove: An Android Based Application for Wrist Rehabilitation [7]. All three projects above use Android platform for their mobile side development, and they are related to mobile health monitoring and out-clinic rehabilitation.

In paper [5], an Android based control and monitoring system for leg orthosis, which able to help physical therapist and physician in monitoring control the orthosis during rehabilitation process, is successfully developed. In paper [6], a system called VITFIZ is implemented successfully on Android platform to provide a low cost solution that would make management of out-clinic exercise programs for patients undergoing rehabilitation more efficient. In paper [7], a game based rehabilitation system for wrist is developed on Android platform. This system has the innovative advantage of allowing ubiquitous game

based therapy; besides, this system also has possibility of providing patients' out-clinic exercise details to the physicians.

There are also other systems were developed to help people those who in dangerous situation. An Android based Emergency Alarm and Healthcare System [8] was implemented using Global System for Mobile Communications (GSM) and Global Positioning System (GPS) network. The system finds the location of the users when they are in trouble or danger, and triggers the alarm; then, their families and friends can immediately take action to rescue the user.

3. Design Theory and Methodology

System design and requirement analysis are the two main steps that can determine whether a system is going to be an expected one; besides, implement what kind of methods to make the system more user-friendly is also a very important factor.

3.1 System Architecture

The proposed system in this paper uses MVC pattern for both the mobile side and server side development. As it shown in Figure 3.1, Model–view–controller (MVC) is a software architectural pattern for implementing user interfaces on software. The pattern divides a given application into three related parts, so as to separate internal representations of information from the ways that information is presented to or accepted from the user [9].

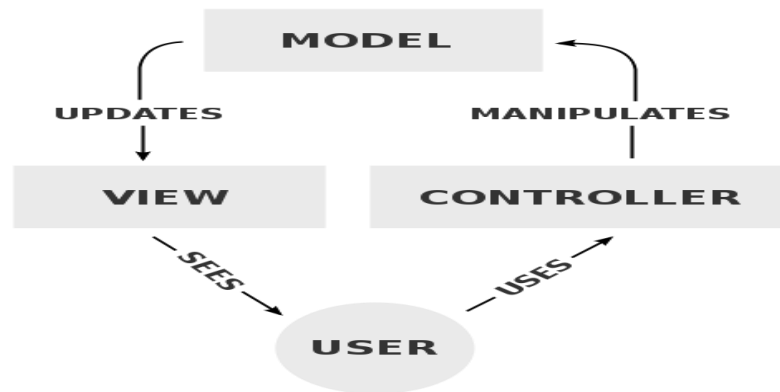


Figure 3.1 A typical collaboration of the MVC components [9]

Besides, divide the developing application into three components, the model-view-controller design also defines the interactions between three components. The main responsibility of a model is storing data which is retrieved according to the commands

that ordered from controller and displayed in the view. The view is responsible for generating an output presentation to the user based on updates in the model. On one hand, controller can send set of commands to the model to change the model's state; on the other hand, controller can also send commands to relating views to update the way that view is being presented.

3.2 Introduction to Key Technology

With the development of computer technologies, more and more applications, web services and systems have been developed, and the communication between different products are getting closer. However, there is a problem: how can one site get the information it needs from the other site without knowing the user's credentials? Before the introduction of OAuth, it was a hard question to answer.

3.2.1 What is OAuth?

OAuth is an open standard for authorization. It is a standard that applications can use to provide client applications with a 'secure delegated access'. OAuth works over HTTP and authorizes Devices, APIs, Servers and Applications with access tokens rather than credentials. There are two versions of OAuth: OAuth 1.0a and OAuth2. These specifications are completely different from one another, and cannot be used together. [10] The version that presented in this paper and implemented in the rehabilitation system is OAuth 2.

3.2.2 What are the functionality of OAuth 2?

OAuth is an authorization workflows which has the capability of providing third party applications to obtain limited, secure and faster access to user information stored on different server, such as Caspio, Ihealth, and Fitbit. First, OAuth represents the user authentication on the service that hosts the user account; then, it authorizes third-party applications to access the user account information. OAuth not only provides authorization workflows for web server applications, it also has covered all the mobile development platforms such as Android, IOS and Windows phone.

How OAuth 2 work?

As we can see from Figure 3.2, OAuth defines four roles in each of its application: resource owner, client, resource server, authorization server. Resource owner is the user who authorizes the client application to access their account. The access is limited by the "scope" that the authorization granted. The resource server is the host that protects the user account information. The authorization server verifies the identity of the user then issues access tokens to the client application. The client is the application that wants to access the user's account. [11] But in order to access, third-party application must be authorized by the user, and the authorization must be validated by the API from the service.



Figure 3.2 OAuth 2 overall workflow

3.2.3 Authorization Grant

In Figure 3.2, the main functionality of first four steps includes obtaining an authorization grant and access token. The authorization grant type depends on the method used by the client application. OAuth 2 defines four grant types: Authorization Code grant type is mostly used by server-side applications such as Fitbit and iHealth; Implicit grant type is mostly implemented by mobile applications and web applications; resource owner password credentials grant type is used by trusted applications, and client credentials grant type is used in order to get applications an API access. The grant types that implemented in the presented system is Authorization Code as shown in Figure 3.3.

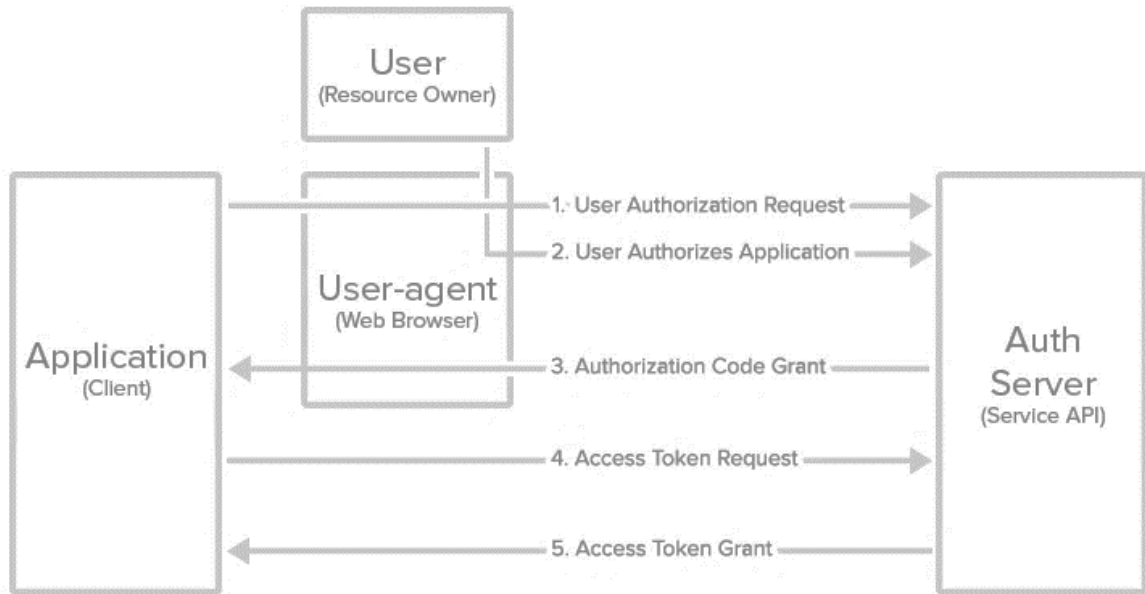


Figure 3.3 OAuth2 authorization code flow

3.3 Android System

Android is a mobile operating platform developed by Google. It is open source, and designed primarily for mobile devices such as smartphones and tablets. However, in recent years, Android is also applied on wearable devices, televisions, cars and laptops.

3.3.1 Android System Architecture

As it is shown in Figure 3.4, android architecture is categorized into five parts: Linux kernel, native libraries (middleware), Android Runtime, Application Framework, and Applications [12].

Linux kernel is the heart and most important part of android architecture that exists at the root. The main responsibilities of the kernel are manage device drivers, power management, memory management, device management and resource access.

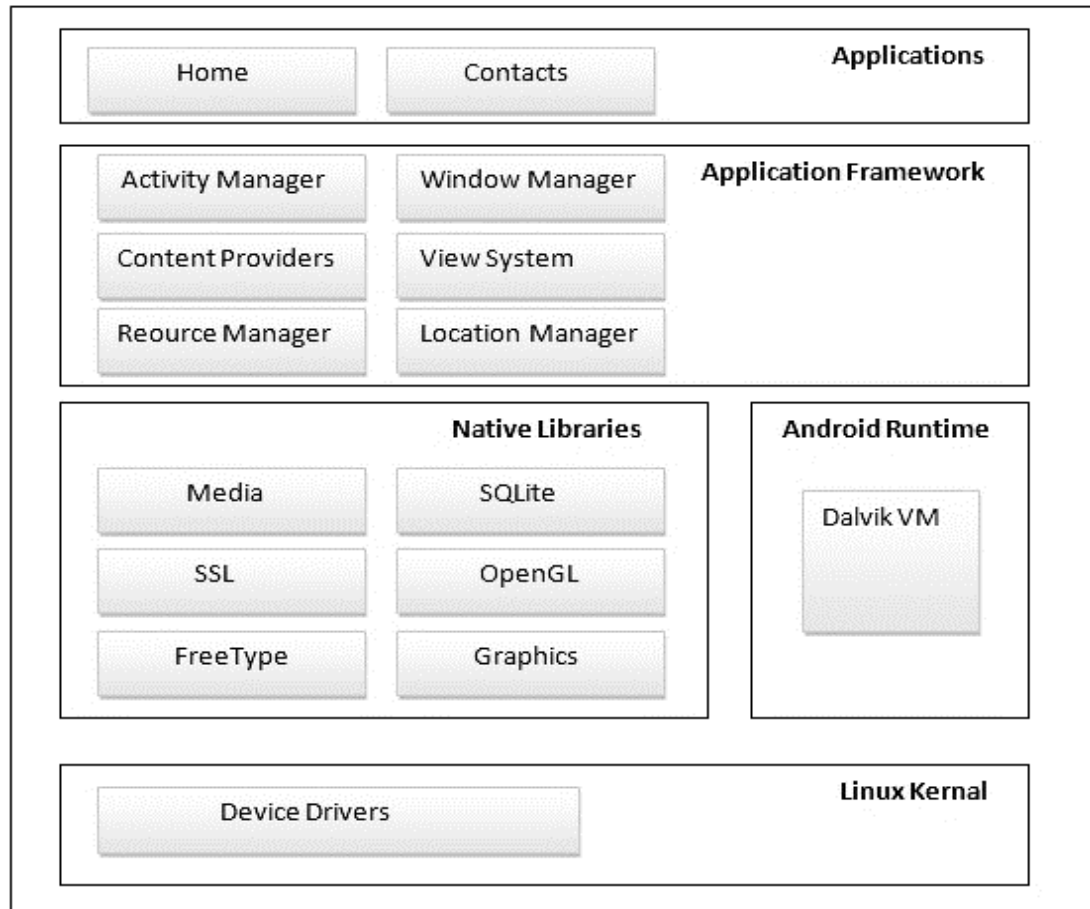


Figure 3.4 Android System Architecture

It can be seen from Figure 3.4 that on the top of Linux kernel are Native libraries such as the mobile side database system SQLite, Media storage system, c-libraries etc. The android runtime includes core libraries and Dalvik Virtual Machine which is responsible for running android application. Dalvik Virtual Machine consumes less memory and provides fast performance. Android operating system also includes APIs such as User Interface, resources, locations, Content Providers and package managers. Besides, it also provides a lot of classes and interfaces for android application development. The very top layer is applications layer, which includes native applications that come with the system

such as home, contact, settings, browsers etc. as well as third party applications such as the mobile side of the system (OTSRS app). They are all using android framework and libraries.

3.3.2 Android Core Building Blocks

Generally speaking, android component is a piece of code that has a well-defined life cycles such as Activity, Receiver, and Service. As it shown in Figure 3.5, the core building blocks of android are activities, views, intents, services, content providers, fragments and Android Manifest.

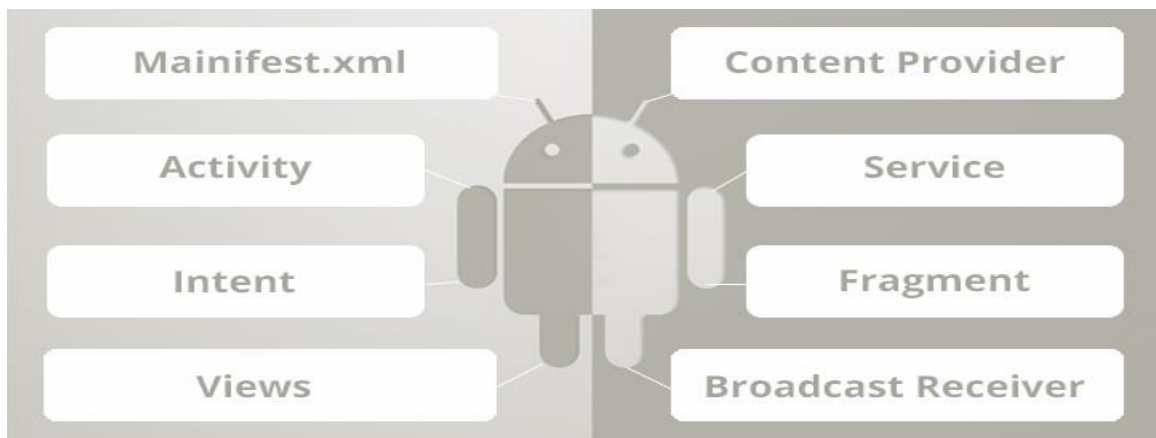


Figure 3.5 Android Core Building Blocks

Activity is a class that represents a single screen. It is like a Frame in AWT. A view is the user interface element such as button, label, text field, image view etc. Anything can be seen through the application is a view. Intent is used to invoke components. It is mainly used to: Start the service, Launch, Display, and Broadcast. Service is a background process that can run for a long time without user notice. There are two types of services: one is local service and the other one is remote service. Local services is referring to the service

that runs within the applications; whereas, the remote service means remote accessing the local services within the applications. Content providers is used for sharing data between different applications. Fragments is as part of the activities. An activity can display more than one fragments on the phone screen at the same time. Android Manifest contains information about activities, content providers, permissions etc.

The android side of the system presented in this paper mostly use fragments because fragments are more reusable than custom views.

4. System Design

As shown in Figure 4.1, the proposed orthopedic trauma surgery rehabilitation system in this paper consists of four major parts: Wearable sensors module, OAuth2 authentication module, Mobile computation module, and Web Server Module.

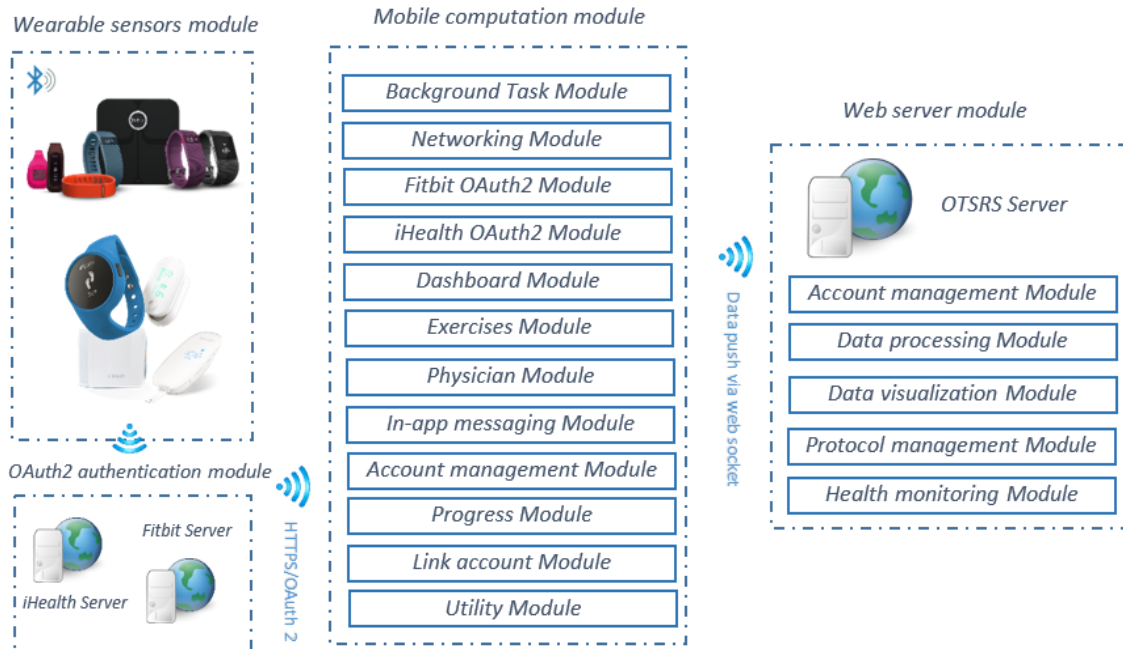


Figure 4.1 Overall System Architecture

The Mobile Computation Module consists of 12 sub-modules including Background Task Module, Networking Module, Fitbit OAuth2 Module, iHealth OAuth2 Module, Dashboard Module, Exercises Module, Physician Module, In-app messaging Module, Account management Module, Progress Module, Link Account Module and Utility Module.

The web server module includes account management module, data processing module, data visualization module, protocol management module, and health monitoring module; besides, database design is also part of the web server module.

4.1 Wearable sensors module

Wearable fitness sensors have gained popularity recently. Cheap devices measure general activity level through motion detection. More expensive devices measure more precise physiological data such as heart rate. In addition, these devices save the measurement results either in a phone or in the cloud. Fitbit is one of these manufacturers. They make portable fitness trackers and online scales.

Wearable sensors module includes health monitoring devices manufactured by Fitbit and iHealth, and they are responsible for recording and syncing the patient's physiological data to the oauth2 authentication module. Figure 4.2 shows some of the sensors that supported by the proposed system.



Figure 4.2 Kinds of wearable sensors that the proposed system supports

Wearable sensors have been widely used in medical sciences, sports and security. Wearable sensors can detect abnormal and unforeseen situations, and monitor physiological parameters and symptoms through these trackers. This technology has transformed healthcare by allowing continuous monitoring of patients without hospitalization. Medical monitoring of patients' body temperature, heart rate, brain

activity, muscle motion and other critical data can be delivered through these trackers. Moreover, in sports training there is an increasing demand for wearable sensors. [13]

4.2 OAuth2 authentication module

The OAuth2 authentication module mainly responsible for establishing API connections to mobile computation module. The rehabilitation system presented in this paper supports two wearable sensor companies' APIs, Fitbit and iHealth; both of the companies uses OAuth 2.0 for user authorization and API authentication. The OAuth 2.0 framework requires the application to obtain an Access Token when the users authorize the system to access their data. The Access Token is used for making HTTP request to the API.

4.3 Mobile computation module

The Mobile computation module mainly consists of an Android application called OTSRS, which stands for orthopedic trauma surgery rehabilitation system. OTSRS app is a context aware mobile application that is developed to perform on phones or tablets with Android Operating System. The OTSRS app includes 12 modules: Background Task Module, Networking Module, Fitbit OAuth2 Module, iHealth OAuth2 Module, Dashboard Module, Exercises Module, Physician Module, In-app messaging Module, Account management Module, Progress Module, Link Account Module and Utility Module. The main functionality of the modules talked above is explained in this section and detailed implementations of these modules are discussed in System Implementation section.

4.3.1 Background Task Module

The background task module mainly responsible for loading data and managing device status. It is designed to improve user interface performance, and minimize its drain on the battery by sending work to a service running in the background; besides, long-running I/O operations can also be passed to background task module in order to improve patients experience when switching different fragments.

4.3.2 Networking Module

Networking module is responsible for establishing cellular or Wi-Fi networking connection to the OTSRS app. After the cellular or Wi-Fi networking connection established, Networking Module communicate with web server module; then, transfer the encrypted data get to the web server using Hypertext Transfer Protocol (HTTP) protocol. The OTSRS system sends the all the data after checking the android device network connection state, so everything that sent to the web server will be received; otherwise, the server could not receive the patients data.

4.3.3 Fitbit OAuth2 Module

First of all, Fitbit OAuth2 module is responsible connecting Fitbit server with the rehabilitation system that presented in this paper. After the API connection established, the module communicates with the Fitbit server to get patient's health data (steps, floors, consumed calories, burned calories, waters, sleep etc.); then, send the retrieved data to data encryption method to encrypt; after the encryption completed, on one hand, the module sends the encrypted data to networking module, on the other hand, retrieved data is sent to dashboard module to display.

4.3.4 iHealth OAuth2 Module

The iHealth OAuth2 module allows the rehabilitation system to interact with patient's iHealth's data in OTSRS app and web server. The iHealth API allows for most of the read and write methods that the system needs to support OTSRS app. As it described in last section, the first step of the module is connecting OTSRS app with the iHealth server; after the API connection is authorized, the module communicates with iHealth sandbox to get patient's health data (heart rate, glucose, blood pressure etc.); then, send it both to encryption module to encrypt and dashboard module to display.

4.3.5 Dashboard Module

Dashboard module responsible for displaying patient's health data in user-friendly format. It has some features: view determination, date selection, refreshing, dashboard customization, and automatic data upload schedule; the dashboard module first trigger the view determination method when it clicked; if the patients have already linked any wearable device, the module displays the dashboard page; otherwise, the module will hand this request to Link account module.

In dashboard page, patients can select a particular date to see their health data that retrieved from OTSRS server. If the date is chosen as today's date, patients also can swipe down to refresh the data that shown in the dashboard page. Patients can also customize the dashboard as they want; they can delete the field they do not want to see. In order

to get the most completed data of the day, an auto uploading method is designed using alarm manager in android operating platform.

All of the auto uploading sections are under control of background task module; so, even if the OTSRS app is not activated, the data will be upload without the patients' notice.

4.3.6 Exercises Module

Rotator cuff repair and ACL reconstruction protocols for each physician is different, and for each particular surgery the type of the protocols is also different. As we can see from Figure 4.3 and 4.4, each protocol includes detailed exercises for each week sessions, so the main responsibility of exercise module is to show the patients today's exercises.


 MIZZOU.	
ACL Reconstruction Rehabilitation Protocol	
Seth L. Sherman, M.D. Tamara L Young, ATC, OTC, MEd Department of Sports Medicine	
Diagnosis: Right/Left ACL Reconstruction with BTB Autograft/Allograft, Hamstring Autograft	
<hr/>	
Date of Surgery: _____	
Frequency 2-3 times per week.	
	<ul style="list-style-type: none">➤ No open chain or isokinetic exercises➤ Provide patient with home exercise program per protocol <p>Weeks 2-6 - Period of protection</p> <ul style="list-style-type: none">➤ Weight bearing as tolerated without assist by post-op day 10.➤ BTB Autograft: Brace unlocked for ambulation when quad control is adequate. Discontinue brace no sooner than 4 – 6 weeks post operative at the discretion of the therapist. Brace locked in extension at night until full terminal extension is attained.➤ BTB Allograft: Brace unlocked for ambulation when quad control is adequate. Discontinue brace no sooner than 2 – 4 weeks post operative at the discretion of the therapist. Brace locked in extension at night until full terminal extension is attained.➤ Hamstring Autograft: Brace unlocked for ambulation when quad control is adequate. Discontinue brace no sooner than 2 - 4 weeks post operative at the discretion of the therapist. Brace locked in extension at night until full terminal extension is attained.

Figure 4.3 ACL Reconstruction Rehabilitation Protocol



Arthroscopic Rotator Cuff Repair Rehabilitation Program

**Seth L. Sherman, M.D.
Tamara L. Young, ATC, OTC, M.Ed**

Department of Sports Medicine

Diagnosis: Right / Left Rotator Cuff Repair

Date of Surgery:

Frequency 2-3 times per week.

Week 0-1:

- Wear shoulder sling at all times except when exercising and for hygiene
- Cryotherapy 8 times per day (20 minute sessions)
- Home Exercise Program 3 times daily (Codman's, Elbow ROM, Wrist ROM, Grip Strengthening)

Week 1-6:

- True PROM only! The rotator cuff tendon needs to heal back into the bone
- ROM goals: 140° FF/40° ER at side; ABD max 60-80° without rotation
- No resisted motions of shoulder until 12 weeks post-op
- Grip strengthening
- No canes/pulleys until 6 weeks post-op, because these are active-assist exercises
- Heat before PT, ice after PT

Weeks 6-12:

- The patient will discontinue sling
- Begin AAROM → AROM as tolerated
- Goals: Same as above, but can increase as tolerated
- Light passive stretching at end ranges
- Begin scapular exercises, Passive Resistance Exercises for large muscle groups (pecs, lats, etc)

Figure 4.4 Rotator Cuff Repair Rehabilitation Protocol

Each exercise item includes title, details, frequency, related video instructions, and a check box that indicates whether the patients finished the particular exercise; besides, exercise module also includes a survey page, which can tell the physicians in detail about their feeling, performance, and compliance. After patients finished each day exercises, all the information they filled about the exercise is sent to web server through networking module.

4.3.7 Physician Module

The physician module is mainly responsible for binding physicians with patients, adding corresponding protocols and showing the details of physician. If patients are newly registered user, they need to bind physicians and protocols in order to access exercise module. If they already linked to particular physicians, physicians' information is displayed; besides, patients can make a phone call by just clicking the phone number in physician module.

4.3.8 In-app messaging Module

In-app messaging module can be accessed from physician module; it is using background task module and network module to get instant messages from physicians. Because, for now, MU Health Care server and OTSRS server is running separately; considering EMR consistency, sending messages from app to server is prohibited in this system. The implementation of in-app messaging module, and how modules communicate with one another will be explained in detail in the next section of the thesis.

4.3.9 Account management Module

The OTSRS app is mainly used by patients, and the responsibilities of account management module are registering new user, logging in, resetting password and updating user information. Account management module cooperates with Networking Module to communicate with the OTSRS server to add, verify or update the patients' account information on the server.

4.3.10 Progress Module

In order to increase patients' interest of taking exercises every day, a game style competition page is designed into progress module. In this module, patients can see their overall progress versus others'; besides, they can also see how many days that they have finished from surgery date, and how many days left until the expected fully recovery date. This module on one hand can give patients some sort of stress to finish their exercises, on the other hand give them hope of getting better.

4.3.11 Link Account Module

The link account module is the interface of requesting API connections to third party server (Fitbit and iHealth). Patients can choose which sensor they want to link to OTSRS account. They can also choose which information that they want OTSRS system to get; if one day they do not want to share their personal data with OTSRS system, they can revoke access, and all the API credentials linked with this particular patient will be deleted from the server.

4.3.12 Utility Module

Utility Module includes sets of functions and methods that frequently be used by the system; it also includes a series of static initialed variables which never be updated during the whole system cycle; besides, it includes encryption, sending emails and database management classes. There are also some resource files placed in utility module, which include some contents of the system such as surgery type, states name and initials, API credentials, photos, colors and more.

4.4 Web Server Module

The web server module, which is built on a Linux server with support of MYSQL Database, is consists of five independent modules. They are account management module, data processing module, data visualization module, protocol management module, and health monitoring module. Web server module is the core of the whole OTSRS system; it is not

only responsible for data processing, but also in charge of storing all the data received from both front-end website and OTSRS app into the database.

4.4.1 Account Management Module

The OTSRS admin is mainly used by physicians to monitor and track patients' health progress. The responsibilities of account management module are registering new physician, logging in, resetting password and updating personal information. Account management module communicate with database to perform add, verify or update the physicians' account information. The logging in method uses email password pair to prevent unauthorized physician from accessing the patients' health data.

4.4.2 Data Processing Module

Data processing module is responsible for performing three tasks: data decrypt, data processing, and data storage. Firstly, the server receives encrypted data from OTSRS mobile app; then, the server will insert all the encrypted data into the database. When it is time to display the encrypted data on OTSRS admin website, the server will retrieve the data, and send to decryption method to perform decryption. Finally, the decrypted data will be send to data visualization module to be displayed in user-friendly format (chart and map).

4.4.3 Data Visualization Module

In order to display live data in user-friendly format on OTSRS website, real-time chart illustration is designed. In terms of chart illustration, two types of charts are available:

patients' origin map and patients' progress linear. All charts are illustrated in real-time and designed with a color scale to let the physicians have direct visual experience.

4.4.4 Protocol Management Module

The protocol management module is for physicians to add and update new protocols. It support review methods; so, physicians can update the wrong field that they accidentally inserted, and new protocol will be set as official one only after the review section completed. Health Monitoring Module

4.4.5 Health Monitoring Module

Health monitoring module is consist of four dynamic tables and two methods. All four tables have functionality of multiple searching and sorting features. They are patient information table, patient health data table, patient progress table and physician protocol table. In terms of two methods, the first one is progress visualization method, which uses data visualization protocol to illustrate particular patient's progress data in linear chart. The other one is sending message method. Physicians just need to add patients email address, and compose an email like message. After the message sending completed, the system will send an email to corresponding patient indicating there is a message in his/her OTSRS app.

5. System Implementation

This section is about detailed implementation of orthopedic trauma surgery rehabilitation and health monitoring system for rotator cuff repair and ACL construction. The development tools used to develop this system are Eclipse and Notepad++. The OAuth2 authentication is implemented using Java language. The mobile computation module is also implemented using Java language with Android SDK (software development kit) and XML (Extensible Markup Language). The web server module is implemented using PHP, JavaScript, HTML, CSS, and SQL languages.

5.1 OAuth2 authentication module

This section mainly presents the preparation work that needs to be finished before the implementation of Fitbit and iHealth OAuth2 authentication framework.

5.1.1 Fitbit OAuth2 authentication

The first step is to read through all the Fitbit API documents [14]; then, begin second step of authentication which is register an application to get an API client credentials. The credentials include Client ID, Client Secret, Authorization URI, and Token Request URI. Figure 5.1 shows the registered Fitbit application credentials for the presented system.

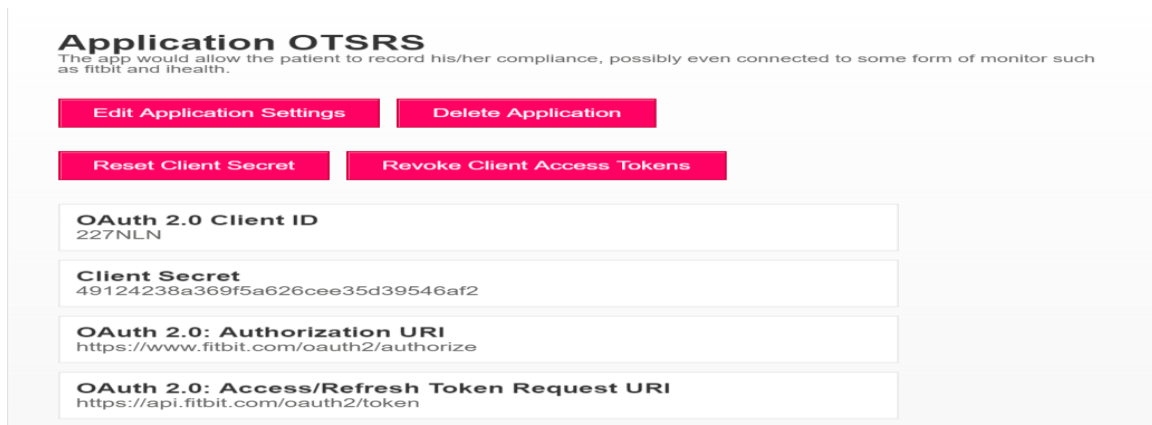


Figure 5.1 Fitbit Application Credentials

Because these credentials will not be changed during the whole system cycle, all the Fitbit application credentials are stored in a resource file in mobile computation module in order to be used by other classes. The format of the resource file is shown in Figure 5.2.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="ClientID">227NLN</string>
  <string name="ClientSecret">49124238a369f5a626cee35d39546af2</string>
  <string name="AuthorizationURI">https://www.fitbit.com/oauth2/authorize</string>
  <string name="AccessRefresh">https://api.fitbit.com/oauth2/token</string>
  <string name="ClientKey">9a11b07a910db4d2a12135b7ad0b8648</string>
  <string name="RedrictURI">http://otsrs.uytechs.com</string>
</resources>
```

Figure 5.2 Fitbit OAuth2 Configuration File

5.1.2 iHealth OAuth2 authentication

Firstly, read through all the iHealth API documents [15]; then, register an application to get an iHealth API credentials. The credentials not only includes Client ID, Client Secret, Authorization URI, and Token Request URI, but also includes API name, SC and SV of each health data, which is shown in Figure 5.3. Then, store all credentials in iHealth credentials resource file which is written in XML.

API Name	SC & SV
OpenApiActivity	SC: e970bba2788947a498ddfcbfb3bdb29a SV: 94aaeadc792a4cd1a098986020914530
OpenApiBG	SC: e970bba2788947a498ddfcbfb3bdb29a SV: bdf0dd95f5044679959f0a52a3a2b6f5
OpenApiBP	SC: e970bba2788947a498ddfcbfb3bdb29a SV: 0a64900887ef47c1b8918958198d177c
OpenApiFood	SC: e970bba2788947a498ddfcbfb3bdb29a SV: bba3fa6dff1c43d8820f776cac24a817
OpenApiSleep	SC: e970bba2788947a498ddfcbfb3bdb29a SV: f486cbcf3e2240a0b1d2de990ebcbae7
OpenApiSpO2	SC: e970bba2788947a498ddfcbfb3bdb29a SV: 5e78e45ce5604e55b4a64761ffe6ba8e
OpenApiSport	SC: e970bba2788947a498ddfcbfb3bdb29a SV: d16603ca8bac45b5a8173ad06800bfcb
OpenApiUserInfo	SC: e970bba2788947a498ddfcbfb3bdb29a SV: c55cf98b57814b2e91de0de9331e341e
OpenApiWeight	SC: e970bba2788947a498ddfcbfb3bdb29a SV: fe0efdb1f25348898c2ae8b26b730f98

Figure 5.3 iHealth API credentials

5.2 Mobile computation module

According to the system design, an Android application called OTSRS, which stands for orthopedic trauma surgery rehabilitation system, is developed. Application is the substance that directly communicates with users. Whether a software application is a good application depends on many aspects, such as usability, stability, communication time between server and application, and layout of each view. A view is composed of three elements: text, image, and layout. In order to make the OTSRS app look user friendly on all Android devices, all the views are implemented using relative layout combining with other layouts such as scroll layout. This section the detailed implementations of each modules are discussed, and how classes, functions, and views communicated with each

other is presented. The login, register and surgery date selection pages are shown in Figure 5.4 and 5.5.

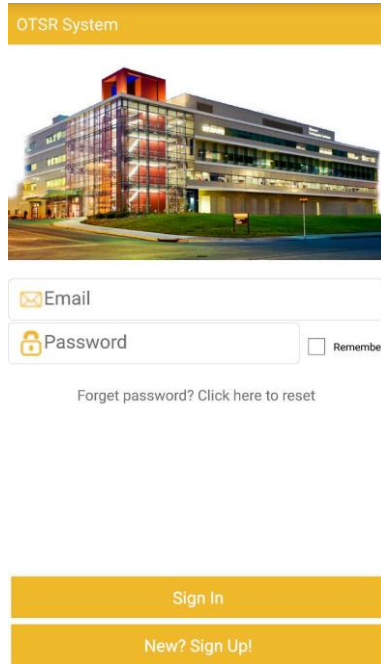


Figure 5.4 Login Page

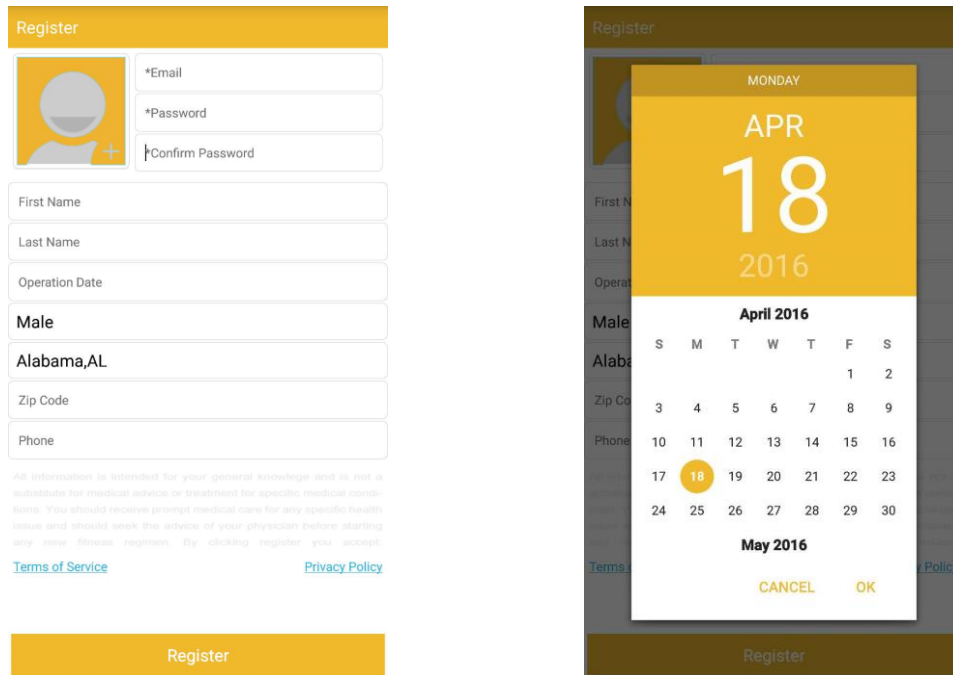


Figure 5.5 Register and Surgery Date Selection Page

5.2.1 Background Task Module

As it is said in the design section, the background task module mainly responsible for data communication and managing device status. In order to achieve data communication in background, all the classes that have load data functions are extended AsyncTask class. AsyncTask, which stands for asynchronous task, can achieve proper and easy use of the user interface thread. This class allows to perform background data loading and push results on the user interface thread without having to manipulate threads and/or handlers. AsyncTask is considered as a helper class around Thread and Handler.

```
class UploadImg extends AsyncTask<String, String, String> {
    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        progressDialog = new ProgressDialog(RegisterActivity.this);
        progressDialog.setMessage("Uploading...");
        progressDialog.setIndeterminate(false);
        progressDialog.setCancelable(false);
        progressDialog.show();
    }
    protected String doInBackground(String... args) {
        try {
            JSONObject json = jsonParser.makeHttpRequest(url_upimg, "POST", paramsimg);
            String message = json.getString(TAG_MESSAGE);
            return message;
        } catch (Exception e) {
            e.printStackTrace();
            return "";
        }
    }
    protected void onPostExecute(String message) {
        progressDialog.dismiss();
        Toast.makeText(getApplicationContext(), message, 1000).show();
        finish();
    }
}
```

Figure 5.6 AsyncTask Upload Image Class

An AsyncTask is defined by a computation that runs on a background thread. It includes three generic types, parameters, progress and result, and four functions, called onPreExecute{...}, doInBackground{...}, onProgressUpdate{...} and onPostExecute{...}. A

sample class, which is responsible for uploading user image to the server, is shown in Figure 5.4.

5.2.2 Networking Module

The networking module is responsible for making response and receiving results from web server. The functionality is implemented utilizing HTTP protocol. In most of the android applications it is essential that app may need to connect to internet and make some HTTP requests. HTTP Post is used in Java to request that a specific web server receive and store data submitted within a request form. The data is submitted and stored in name-value pairs. Anatomy of an HTTP request is shown in Figure 5.5.

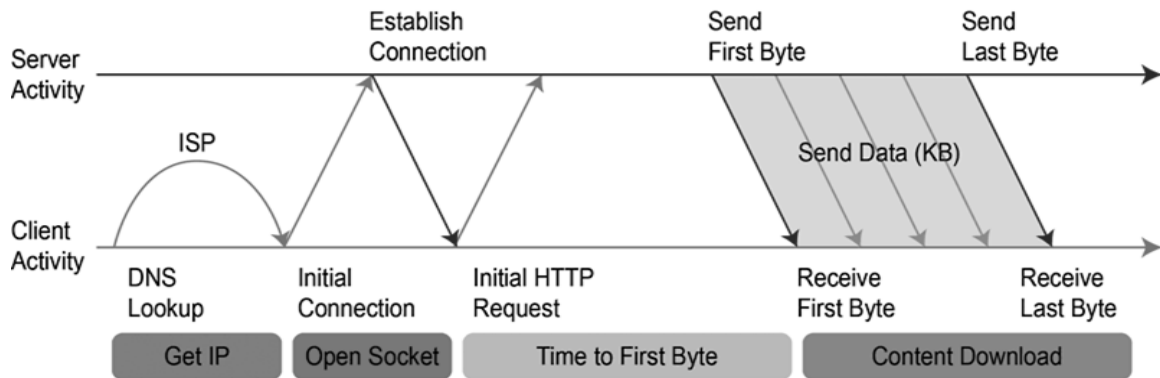


Figure 5.7 Anatomy of an HTTP request

The first step of making HTTP request is creating HTTP Client and HTTP Post; then, it needs to build POST data. Before making HTTP request it also needs to encode the post data in order to convert all string data into valid URL format. Then, execute the HTTP Post request using the HTTP Client thread created before. Last but not least, give the OTSRS application network permission in Android Manifest file.

5.2.3 Fitbit OAuth2 Module

For the purpose of integrating API connection to Fitbit server, two classes are created; they are Fitbit Auth Activity and Fitbit Token Request. The first class responsible asking the server to get authorization code that it is discussed in section two. The authorization code is later used to get access token, refresh token and client id. Fitbit authentication page is shown in Figure 5.8.

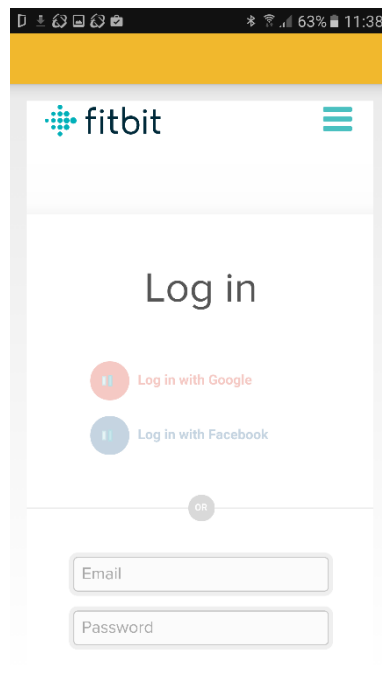


Figure 5.8 Fitbit authentication page

The first step of getting authorization code is to send the Fitbit server a HTTPS request; then, redirect the patient to the Fitbit OAuth 2.0 authorization page with the following parameters: client_id, response_type (code), scope (a list of the permissions that the OTSRS system requesting) and redirect_uri. A web view component is used in this system to load the Fitbit OAuth 2.0 authorization page. After the page loaded, patients enter the

Fitbit username and password; if they matches, they will be redirected to scope page where that can choose which data they want to share with the system, and click Accept; then, Fitbit server redirects the user to the redirect_uri including a “code” in HTML format; then, start Authentication method that implemented in Fitbit Auth Activity class catches the code and send it to Fitbit Token Request class.

After receiving the code, request Token method inside of Fitbit Token Request class makes another HTTPS request to get the access token; if the code is correct, Fitbit server will return five parameters: access_token, refresh_token, user_id, client_para and expires, in a JASON format; then, update User Info method will send these parameters to OTSRS server in order to store them in MYSQL data base; finally, an integration with Fitbit server is complete. The overall flow of requesting API connection to Fitbit server is shown in Figure 5.6.

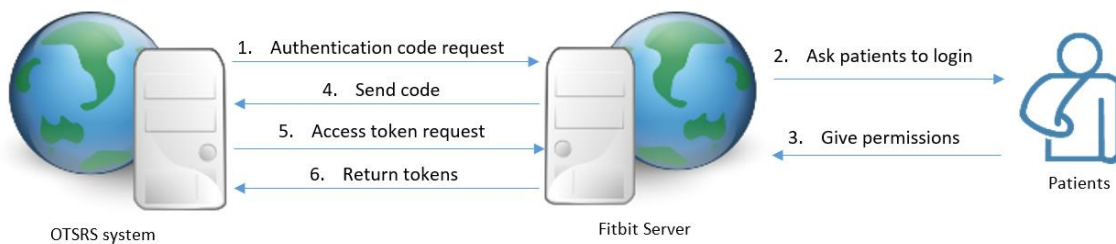


Figure 5.9 Fitbit OAuth 2.0 authentication flow

Besides getting authentication, Fitbit OAuth2 module also has responsibility to get health data from Fitbit data and send it to OTSRS server. In order to get patients’ authorized

health data, Fitbit Get User Info and Fitbit Upload Data classes are implemented using AsyncTask.

Fitbit Get User Info class executes only once when patients successfully linked to Fitbit server. Its main functions are `getUserInfo` and `uploadUserInfo`; `getUserInfo` makes an API call to Fitbit server to get user personal data such as name, gender, age, height, weight, date of birth and Fitbit user id; because responded data is in JSON format, it needs to be parsed; after getting the readable data, `uploadUserInfo` method will send those data to OTSRS web server. If the patients did not choose User Information permission when they connected to Fitbit, Fitbit Get User Info class will not be executed.

5.2.4 iHealth OAuth2 Module

In order to get iHealth OAuth authentication, `ihealth Auth Activity` and `ihealth Token Request` classes are created; as it is explained in last section, `ihealth Auth Activity` class is responsible for asking the server to get authorization code that is discussed in section two, whereas `ihealth Token Request` class is responsible for receiving the authorization code to get access token, refresh token and client id.

The overall flow of requesting iHealth OAuth2 authentication is similar to Fitbit authentication, but with different HTTPS request format; the scope of iHealth API request is a list containing AP Name combining with API SV; after completing the request URL, and authentication request will be sent out; then, iHealth server asks the patients to login and

choose the permission that they want to give to OTSRS system; then, an authentication code will be returned to OTSRS app using URL format followed by `redirect_uri`, which is also different from Fitbit authentication; at last, the code will be sent to iHealth Auth Activity to require access token. The flow is shown in Figure 5.7.

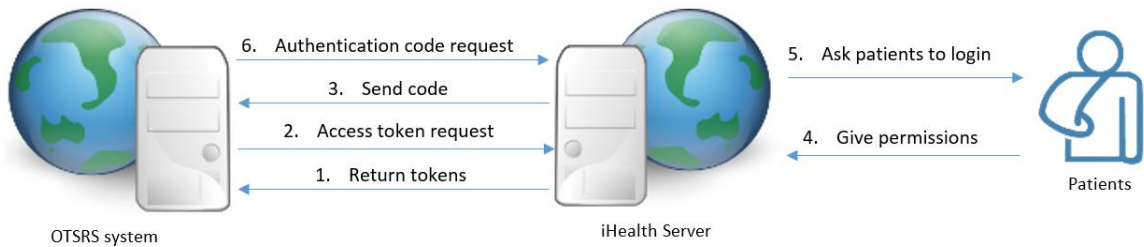


Figure 5.10 iHealth OAuth 2 authentication Flow

This module also has implemented upload user info and health data functions; the first one executes only once, under the condition of being permitted to get user personal information; the second class responsible for uploading health data to OTSRS server through HTTP request. iHealth authentication page is shown in Figure 5.11.

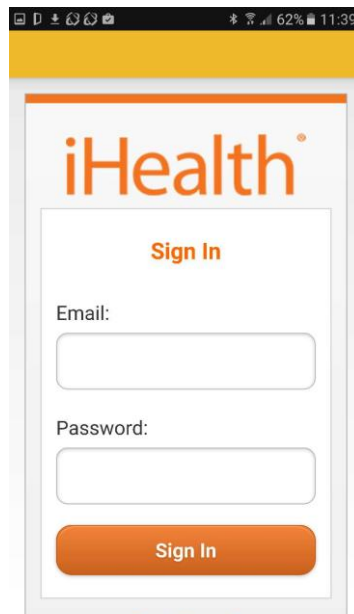


Figure 5.11 iHealth authentication page

5.2.5 Dashboard Module

The dashboard module is responsible for displaying patients' health data in text-image format. It includes six methods: viewDetermination, dateSelection, refreshView, updateData, checkAPIConnection and changeMData; updateData and refreshView are implemented using AsyncTask, and they are running in background.

When patients first login to the dashboard module, checkAPIConnection method send a POST request to OTSRS webserver to check whether patients have already linked to either Fitbit or iHealth API; the returned JSON object will be parsed, and the value will be assigned to a parameter called viewDet; then, viewDet will be passed to viewDetermination method; if patients have linked to at least one API, the view will be changed to dashboard page with all features; otherwise, patients will be directed to link account module by viewDetermination method. In dashboard page, patients can select a particular date to see their health data that retrieved from OTSRS server, which is implemented using dateSelection method. When they choose a date updateData method will send a POST request to web server; then, all the encrypted health data will be respond with JSON format; updateData method will send these encrypted data to utility module to decrypt, then use changeMData method to display it on dashboard page.

As it is said in design section, patients can customize the dashboard as they want. For customization, a component called Sharedpreference is implemented. Basically, the

positions and condition of each field is remembered in sharedpreference; if the field is deleted last time, it will not be shown next time when the patient login; but, if the application is reinstalled, patients need to customize it again. Sharedpreference is one of the Android operating system component used to save and retrieve data in the form of key-value pair. In OTSRS system, patients' login information, login sessions and customize-dashboard values are stored in Sharedpreference. Figure 5.12 shows the design of Dashboard Module and Menu.

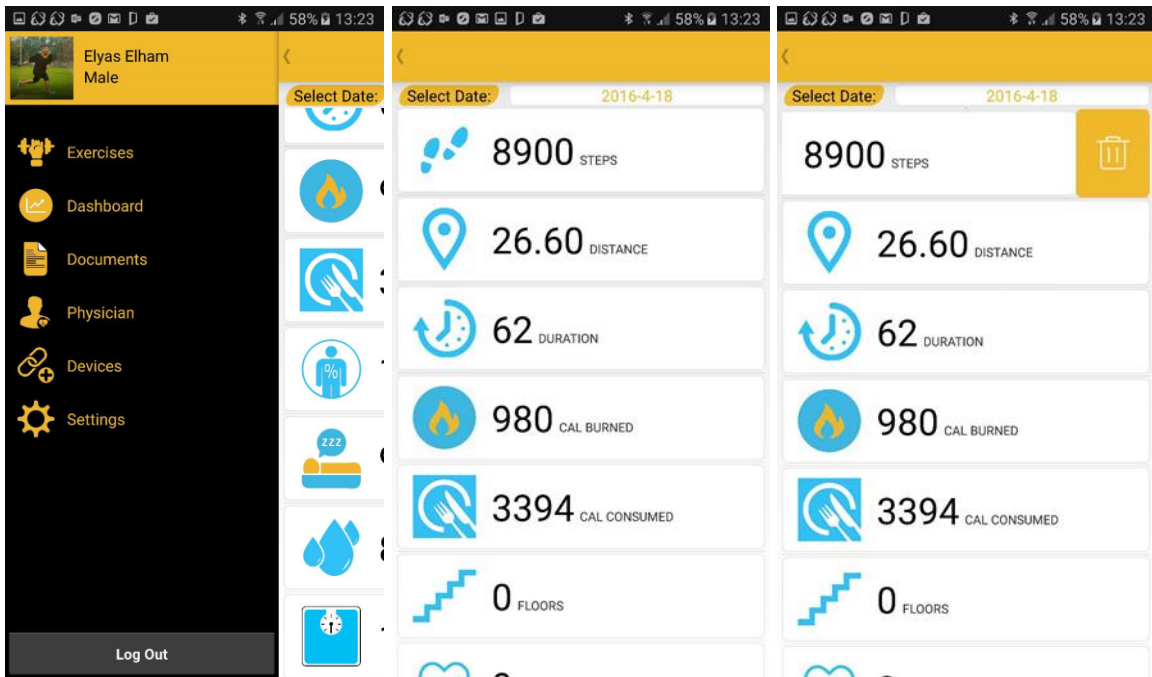


Figure 5.12 Dashboard Page

The last function that dashboard module has is auto uploading which is implemented using alarm manager. AlarmManager class provides access to the system alarm services. These allow the application to be run at some point in the future. When an alarm goes

off, the Intent that had been registered for it is broadcast by the system, automatically starting the target application if it is not already running. Registered alarms are retained while the device is asleep (and can optionally wake the device up if they go off during that time), but will be cleared if it is turned off and rebooted. [16] The Alarm Manager holds a CPU wake lock as long as the alarm receiver's `onReceive()` method is executing. This guarantees that the phone will not sleep until system have finished handling the broadcast. Once `onReceive()` returns, the Alarm Manager releases this wake lock.

5.2.6 Exercises Module

The exercise module is mainly responsible for display the details of each day exercises and submitting survey questions according to their performance during the exercises. These functionalities are implemented by four methods: `checkProtocolINPhysician`, `calWeekSession`, `loadVideo`, `assessProgress` and `uploadData`. `loadVideo` and `uploadData` methods are running in background task module.

When the patients first register and login the OTSRS app, they will be directed to exercises module; then, `checkProtocolINPhysician` method will send a HTTP POST request to the web server to check whether the patient is connected to physician; if patients are not connected to any physicians and relating protocols, they do not have access to exercise module, and they will see an icon with text saying “click to add physician”; after they click the icon, they will be redirected to physician module to handle the add physician task. After entering the physician id, type of surgery and protocol id, and if they match,

they will be granted access to exercises module. Then, calWeekSession will calculate which particular week session that patients fall in, and display the according exercises list to patients. The exercises list is implemented by SwipeMenuListView. In order to utilize SwipeMenuListView in the application, the first step is add SwipeMenuListView in layout XML; then, create a SwipeMenuCreator to add items; at last, listen to the item click event.

Each exercise element may contains an instruction video; if so, the URL of the video that get from web server will be send to loadVideo method; the URL will be wrapped by google video play service and play in the application. The page design of exercises module is shown in Figure 5.13

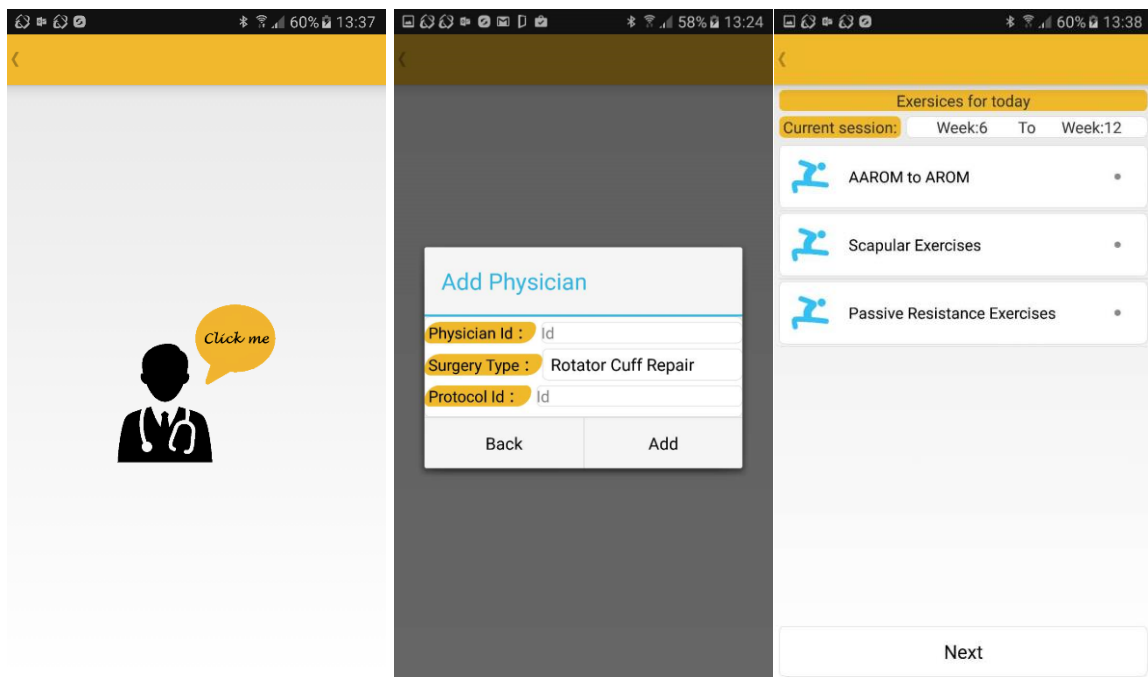


Figure 5.13 Exercise details page

After finishing the exercises, patients can click next button to go to the survey page. On survey page there are there questions, the first one is an animated button to ask patients' feeling during exercises; the second one is how they rate their performance; the last one is overall feedback of today's exercises. After they click submit, all the information they entered will be sent to web server using AsyncTask. The design of survey page and activity details dialog are shown in Figure 5.14.

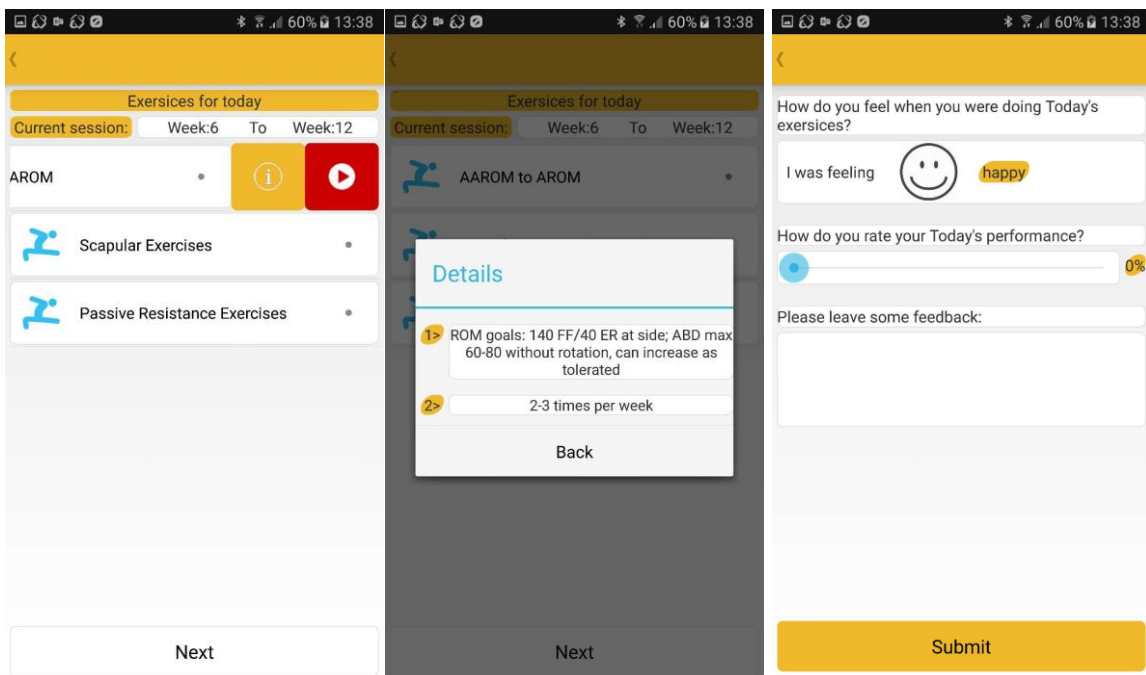


Figure 5.14 Survey page

5.2.7 Physician Module

Physician module is mainly responsible for displaying physicians' details and making calls. When the user first login to the app, they will be directed physicians module from exercise module to add protocols and physicians id; then, a method called checkPhysicianNProtocol will send request to the server to check whether patients and

physicians are connected; if not, patients are asked to enter the physician id and protocol id; if they do not have one, they should ask their physicians to provide the ids to them. The two ids should match in order to access to physician and exercises module; by adding physicians, patients are agreed to sharing their personal health data with their physicians.

After physicians added correctly, physicians' information will be displayed in physician page; patients can make direct call from the app just by clicking the phone number. They do not need to open phone app and re-enter the number again. This feature is implemented by creating a new intent. The design of physician page is shown in Figure 5.15.

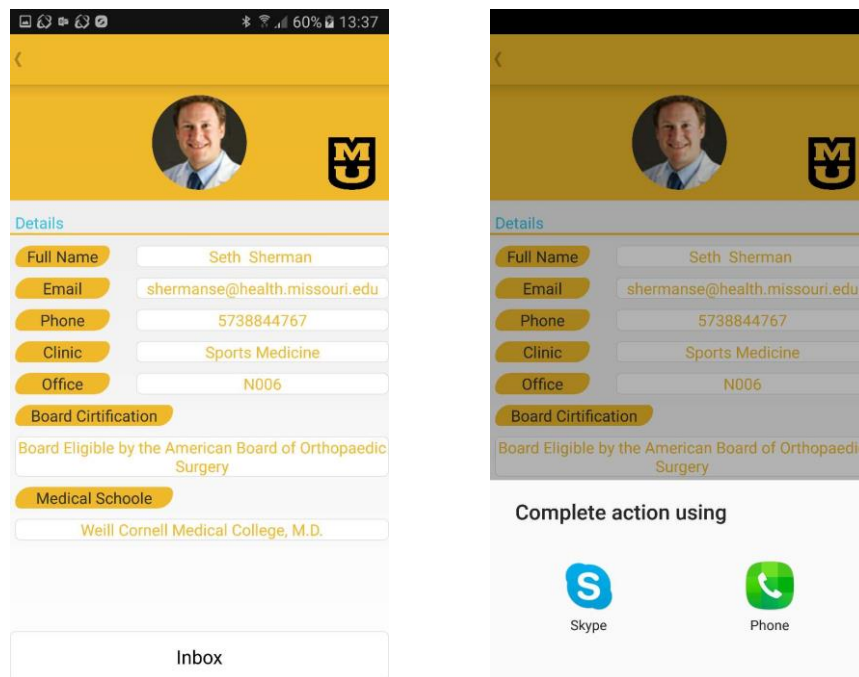


Figure 5.15 Physicians page

5.2.8 In-app messaging Module

In-app messaging module is implemented inside physician module. Patients can access to it by clicking Inbox button. The first page of in-app messaging module is message lists, which is implemented using list view. Each message item shows the title, subject and condition of the message. Only the patients' physician can send message to them. It is controlled by web server. The design of in-app messaging module is shown in Figure 5.16.

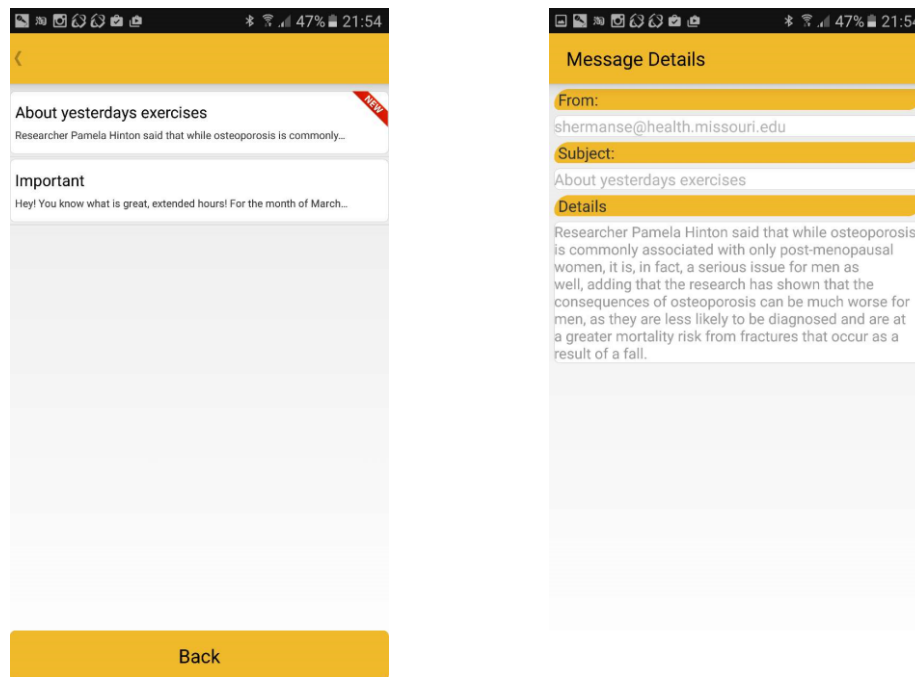


Figure 5.16 In-app messaging page

First, loadMessage function communicates with networking module to send HTTP POST request to OTSRS web server to download all the messages that physicians send to patients. Then, the networking module will use JSON parse function to parse the body of returned message to get each of the message details and store them into arraylist. ArrayList class uses a dynamic array for storing the elements. It extends AbstractList class

and implements List interface. It can contain duplicate elements, and also maintains insertion order. However it is non-synchronized. ArrayList allows random access because array works at the index basis. Finally, the values stored in array list shared between pages using intent. After the message being read, updateMessageReadCondition method sends another POST request to change the read-condition of message.

5.2.9 Account management Module

With the Account Management module of OTSRS app patient can set up, monitor, track, and change critical account information at their convenience. From registering new account to receiving email notification about account credentials, patients can access key information at any point in the process. In addition, the Account Management module enables patients to access and update a wide range of account information, including phone number, zip code, and password.

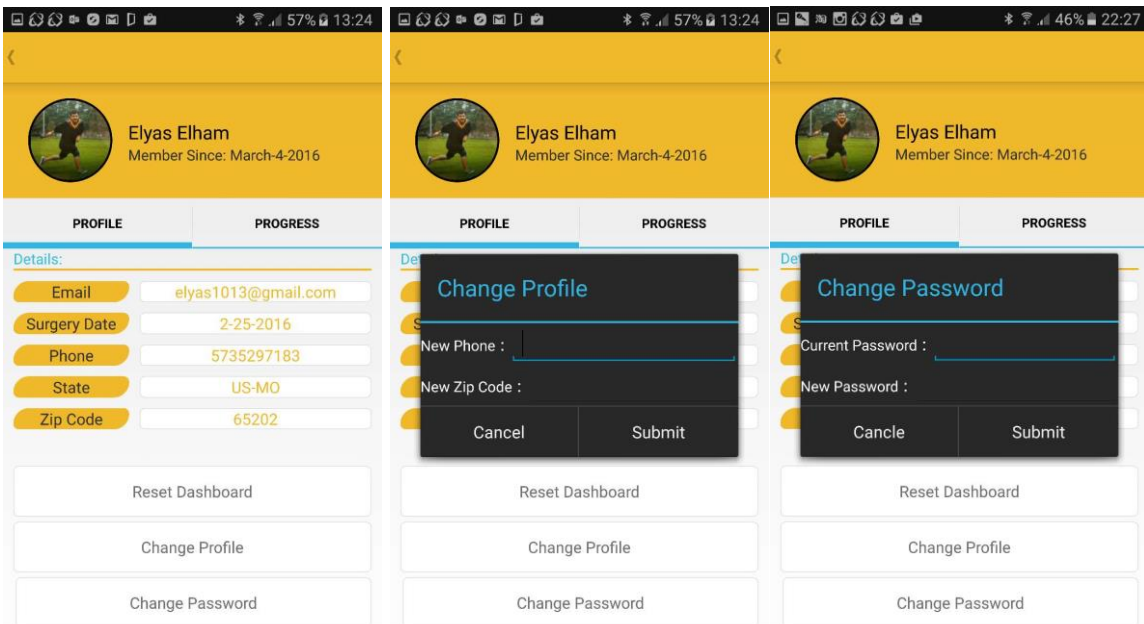


Figure 5.17 Account information page

If patients want to update information, they can go into setting page from menu, and click on relating buttons; after they entered the correct format of data, system will send a HTTP request to server to update it. The account information page is shown in Figure 5.17.

If patients forget their password, they can open the forget password page and enter the email address that they used when they register. If email address exists in web server, a password reset instructions will be send to user email address. If not, they will be noticed by Toast message. Toast is an object in Android framework, it is used to display information for the short period of time. A toast contains message to be displayed quickly and disappears after sometime. The android.widget.Toast class is the subclass of java.lang.Object class. The forget password page is shown in Figure 5.18.

Retrive password

Please enter the email address you used when you registered:

Email

Submit

Figure 5.18 Forget password page

5.2.10 Progress Module

Progress module is designed to show patients overall progress and days of exercises that finished after surgery. It is only accessible when the user is connected to physicians and protocols. Otherwise it shows a message, direct patients to connect physicians. If the patients have already authorized physicians, a method called loadProgress will send POST request to the server; then, JASON format data will be responded; after parsing the data by JASONParse class, the patients' average progress and other patients' average progress will be calculated; then, an animated circle shows the patients progress versus others. Progress page also shows the animated date finished circle to indicate the patients how many days that they have finished after the surgery. The design pf progress page is shown in Figure 5.19.

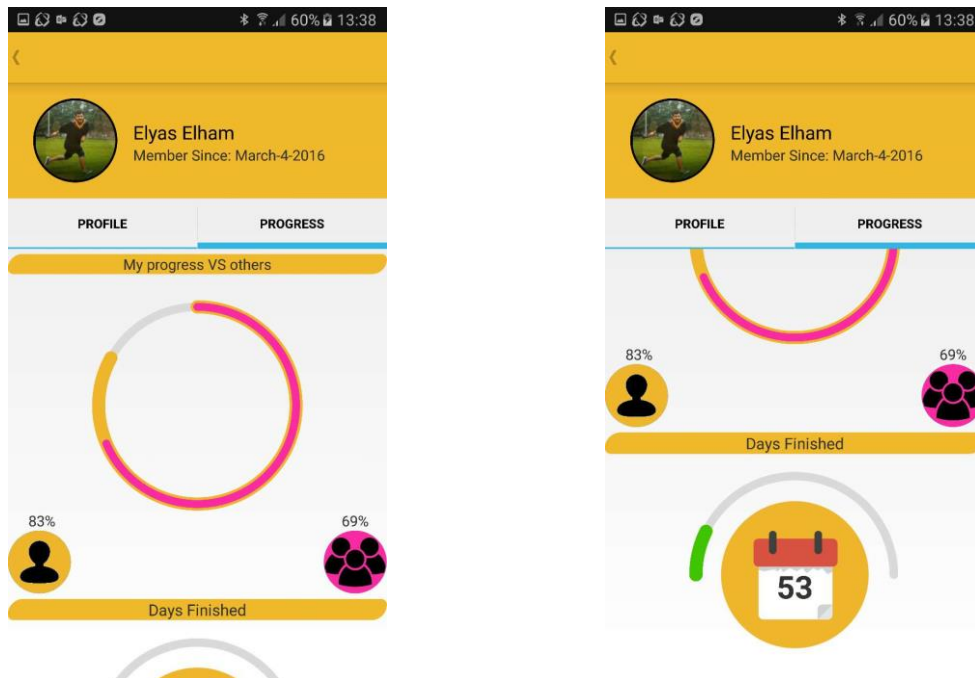


Figure 5.19 Progress page

The animated circle is implemented by using `hookedonplay.decoviewlib.DecoView` library. The first step of implementing animated charts is to Add DecoView to your xml layout as it showed in Figure 5.20; then, Configure DecoView data series in Android Java code; after setting the data series, it needs to add events to animate the data series. DecoView also can change the shape, gravity and orientation of each kinds of chart.

```
<com.hookedonplay.decoviewlib.DecoView
    xmlns:custom="http://schemas.android.com/apk/res-auto"
    android:id="@+id/dynamicArcView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="8dp"
    custom:dv_arc_gravity_horizontal="Fill"
    custom:dv_arc_gravity_vertical="Bottom" />
```

Figure 5.20 Example of an XML layout

XML is a language that's very similar to HTML. It's much more flexible than HTML because it allows developers to create their own custom tags. However, it's important to realize that XML is not just a language. XML is a meta-language: a language that allows developers to create or define other languages. For example, with XML so many other languages created, such as RSS, MathML (a mathematical markup language), and even tools like XSLT. The essence of XML is in its name: Extensible Markup Language. It lets define personal tags, the order in which they occur, and how they should be processed or displayed. The most recognizable feature of XML is its tags, or elements. In fact, the elements that created in XML will be very similar to the elements that created in HTML documents.

5.2.11 Link Account Module

Link account module is implemented using list view combining with fragment. First it will read implemented API names and images from resource file, and store it in an ArrayList object with pairs. Then read it to list view using loop. The design of Link account module is shown in Figure 5.21.

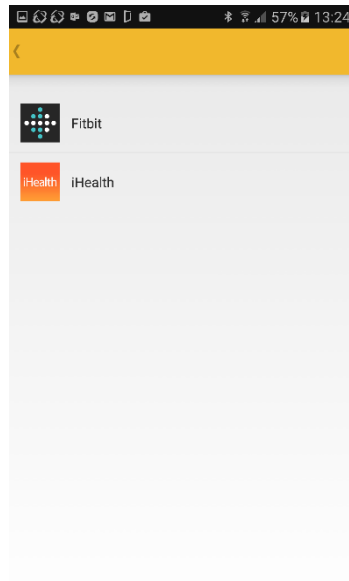


Figure 5.21 Link account page

5.2.12 Utility Module

The Utility Module includes several JAVA classes such as DatabaseUtile which include all the variables and methods that needs to implement SQLite database, MailSender which is used to send email from android phone, MCrpypt which is used to encrypt and decrypt the data that get from the web server, SessionManager which used to manage the session of the login and VideoEnabledWebView which is used to play video in exercises view.

Besides the utility module also manipulates the storage of resource files. All the layout files, which is written in XML language, are under resource/layout folder; all the images and app icons used in the OTSRS application are under resource/drawable folder; all the name-value pairs that used in application are defined in resource/values folder.

In addition, An AndroidManifest file is in the root directory of application folder. It controls the application-version, user permission, activity declaration and target platform version. The AndroidManifest is also written using XML language.

5.3 Web Server Module

Web server module is written in PHP and HTML languages and using MYSQL database. Over all web server is divided into two parts, one is just for maintaining communication between OTSRS application and server; the other one is to support the OTSRS admin website. This section mainly presents the detailed implementation of website and server of OTSRS system. Figure 5.22 shows the OTSRS system admin website.

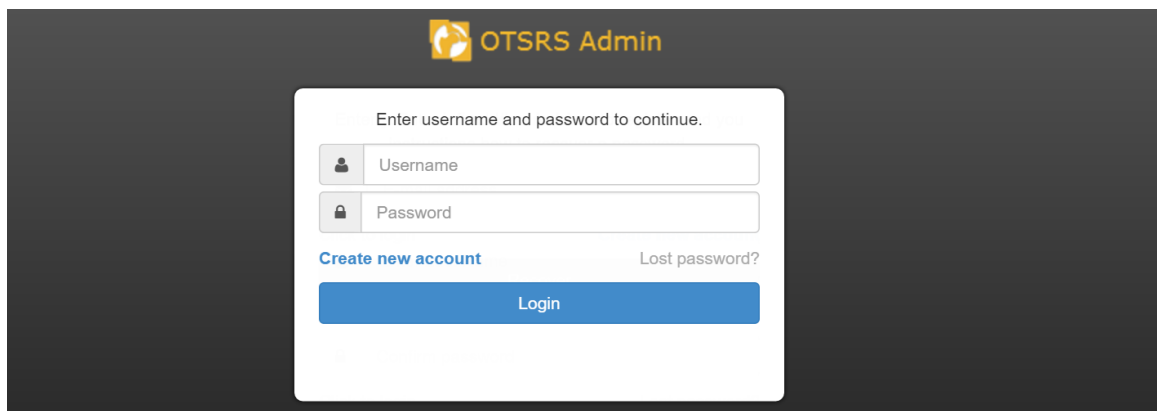
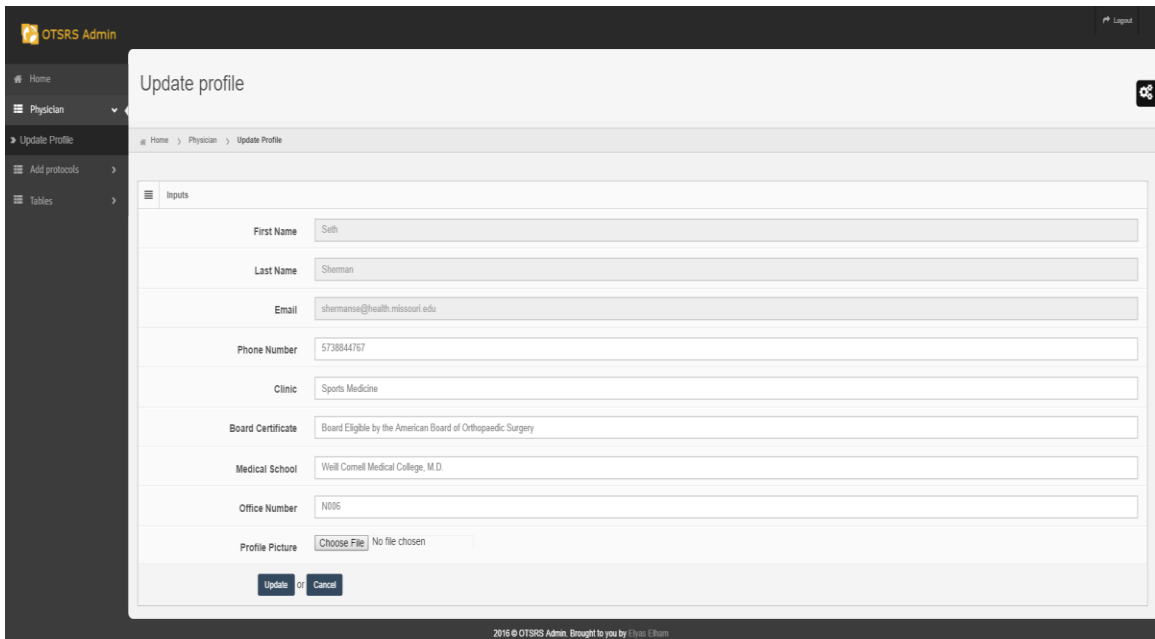


Figure 5.22 OTSRS system admin website login page

5.3.1 Account Management Module

Account management module in OTSRS server provides interface for physicians to register new user, login the website, retrieve forgot password, maintains and update the personal information by connecting with the MYSQL database. PHP, JavaScript, CSS, HTML and SQL languages are mainly used in this module.

Figure 5.22 shows the Physician update profile, register and forget password page. In the implementation, all the physicians' passwords are SHA encrypted. If the physicians forget their password, they only need to enter their email address, and will receive password retrieve email from the server. The two PHP files register.php and forgetpassword.php are responsible handling the request that created from these two pages. Physician email address verification, password format verification and email sender methods are implemented in these files.



The screenshot displays the 'Update profile' interface within the OTSRS Admin system. The page features a dark sidebar on the left with navigation options: Home, Physician, Update Profile, Add protocols, and Tables. The main content area is titled 'Update profile' and contains a form with the following fields:

- First Name: Sath
- Last Name: Sherman
- Email: shermans@health.missouri.edu
- Phone Number: 5738844767
- Clinic: Sports Medicine
- Board Certificate: Board Eligible by the American Board of Orthopaedic Surgery
- Medical School: Weill Cornell Medical College, M.D.
- Office Number: N005
- Profile Picture: Choose File (No file chosen)

At the bottom of the form, there are 'Update' and 'Cancel' buttons. The footer of the page reads '2016 © OTSRS Admin. Brought to you by Elvise Chan'.

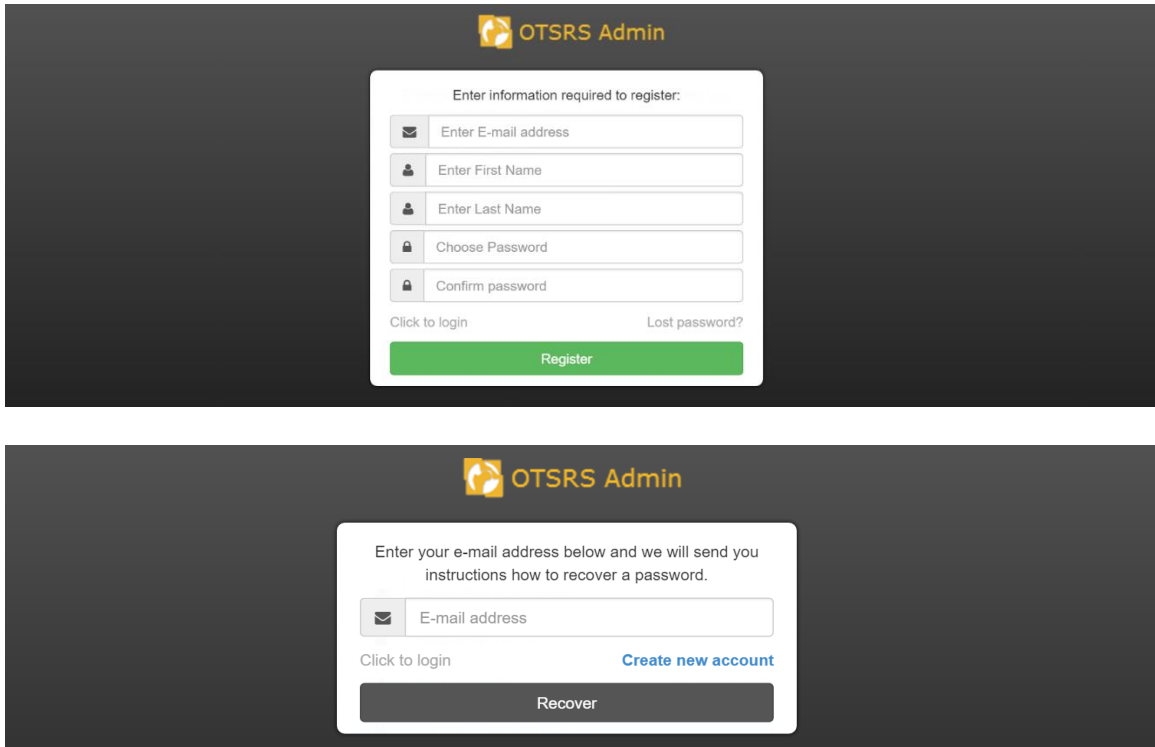


Figure 5.23 OTSRS server update profile, register and forget password page

5.3.2 Data Processing Module

Data processing module is responsible for both the parsing and decryption of data retrieved from the database; besides, it also handles the data coming from OTSRS application.

Data encryption and decryption is using PHP Mycrypt library, which supports a wide variety of block algorithms such as DES, TripleDES, Blowfish (default), 3-WAY, SAFER-SK64, SAFER-SK128, TWOFISH, TEA, RC2 and GOST in CBC, OFB, CFB and ECB cipher modes.

Additionally, it supports RC6 and IDEA which are considered "non-free". CFB/OFB are 8bit by default.

The encryption/decryption method used in this system is MCRYPT_RIJNDAEL_128 because it's AES-compliant, and used cipher Block Chaining mode. Rijndael (pronounced rain-dahl) is the algorithm that has been selected by the U.S. National Institute of Standards and Technology (NIST) as the candidate for the Advanced Encryption Standard (AES). [17] It is known that AES has a fixed block size of 128 bits and a key size of 128, 192, or 256 bits, whereas Rijndael can be specified with block and key sizes in any multiple of 32 bits, with a minimum of 128 bits and a maximum of 256 bits, and it is a symmetric key encryption algorithm. The encryption and decryption flow is shown separately in Figure 5.23 and Figure 5.24.

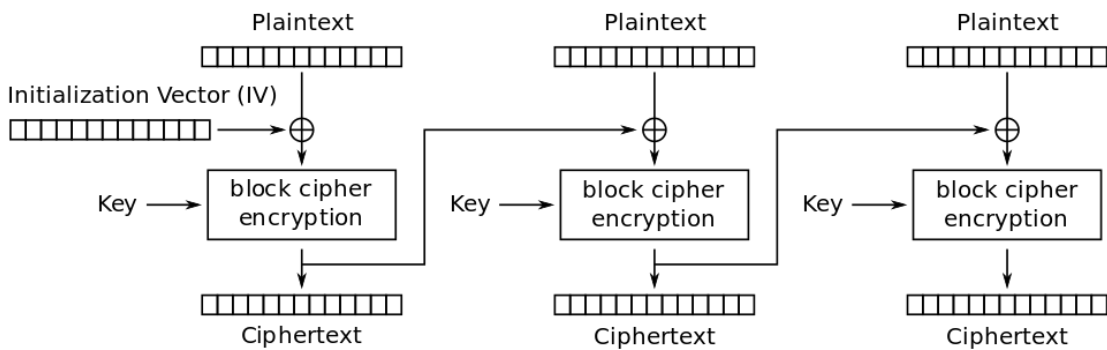


Figure 5.24 RIJNDAEL_128 cipher Block Chaining mode encryption

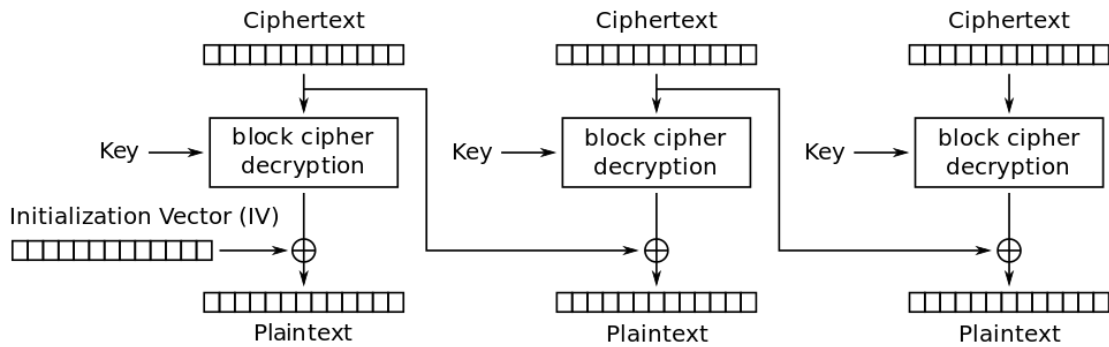


Figure 5.25 RIJNDAEL_128 cipher Block Chaining mode decryption

5.3.3 Data Visualization Module

The main responsibility of data visualization module is to show live charts in OTSRS web server. It is implemented by using google chart APIs. Google chart tools are powerful, simple to use, and free. It can be chosen from a variety of charts from simple scatter plots to hierarchical tree maps. It is highly customizable and has an extensive set of options that can be figured to perfectly match the look and feel of websites. It is also cross-browser compatibility (adopting VML for older IE versions) and cross-platform portability to iOS and new Android releases. Most importantly no plugins are needed. After the data processing step, data will be sent to a function called drawCharts, and using google chart API the data is illustrated. Figure 5.26 and 5.27 show the dashboard page of the OTSRS admin website. The first one indicates the patients' overall progress, and the second one shows the geographical regions that patients come from.

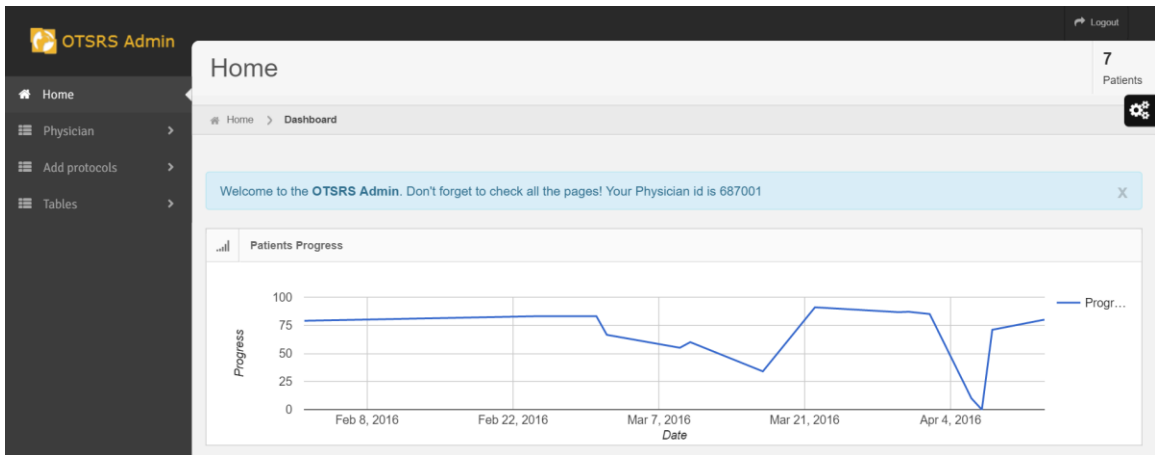


Figure 5.26 Dashboard patients' overall progress

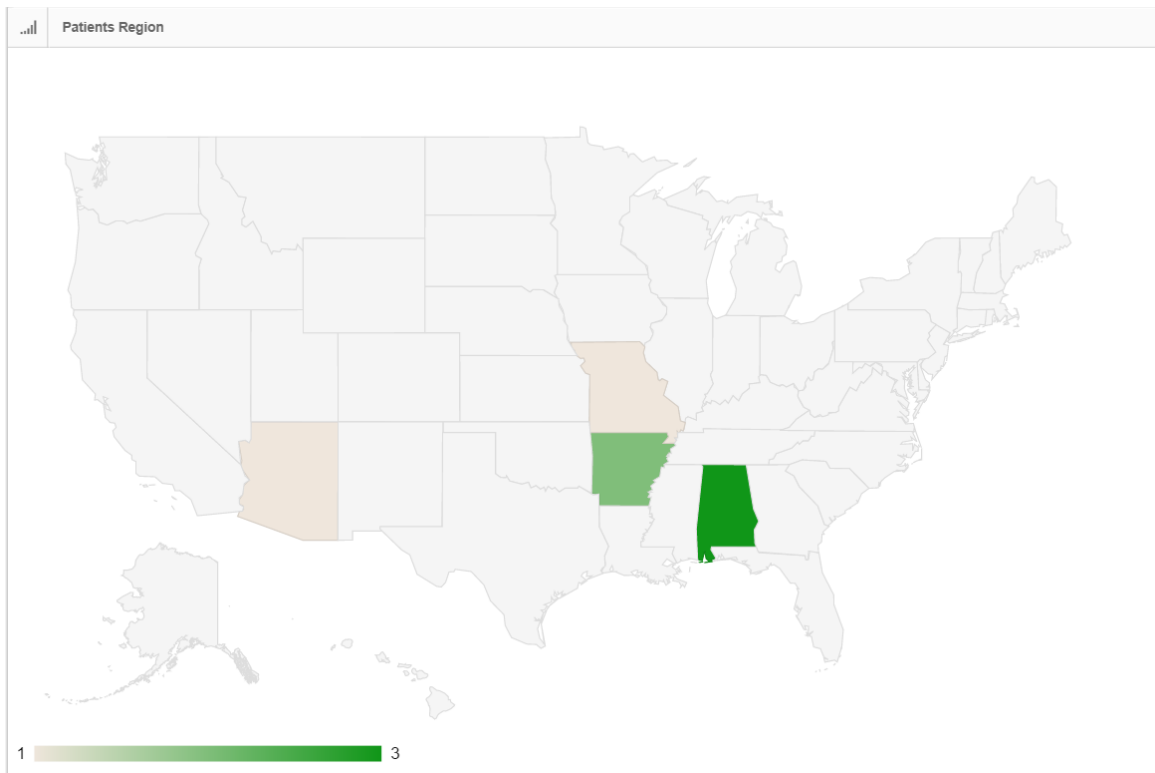


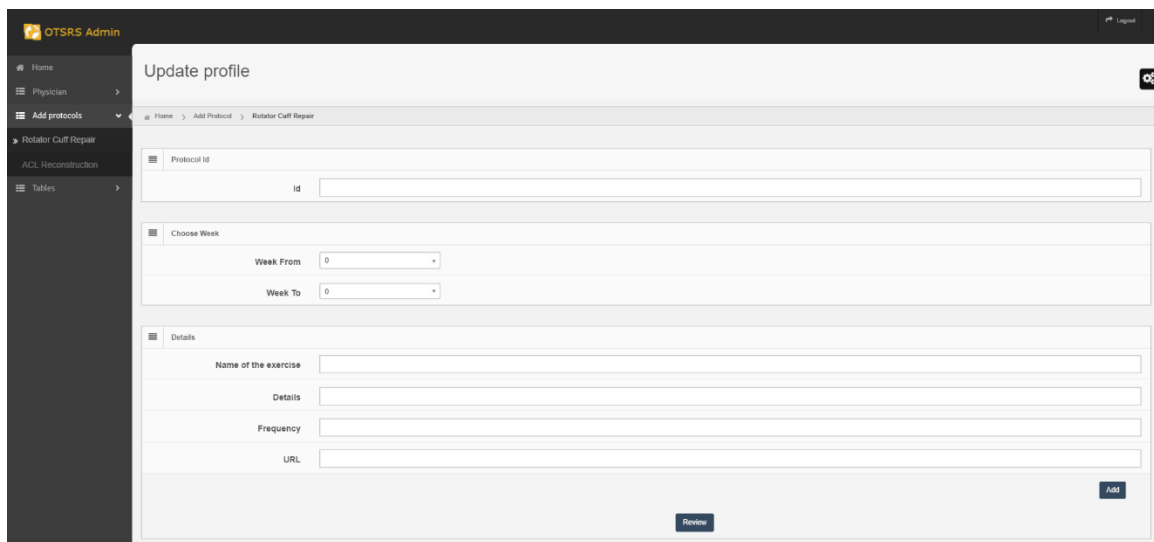
Figure 5.27 OTSRS website patients' region map

5.3.4 Protocol Management Module

The protocol management module implemented by adding, reviewing and updating functions. Physicians can customize the id of protocols. The first step is to set the week

from and week to variables; then, exercise name, details, frequency and instruction video URL can be filled. After completing each one exercise item, physicians need to click on the add button. Then, the addProtocol function starts to send request to server, and will insert a temporary field in database for reviewing purposes.

After completing all of the exercise details in a protocol, physicians can click on review button to check the values that they entered. If anything is accidentally entered wrong, physicians can click on edit button, and insert the line number to update the corresponding exercise details; If physicians feel confident about all the protocol, they can just click submit button, and all the fields in temporary table will be copied to official protocol table. Figure 5.28 and 5.29 shows the add protocol page and review protocol page of the OTSRS admin website.



The screenshot displays the 'Update profile' page in the OTSRS Admin interface. The page is titled 'Update profile' and includes a breadcrumb trail: Home > Add Protocol > Rotator Cuff Repair. The form is organized into several sections:

- Protocol Id:** A text input field with a label 'Protocol Id' and a sub-label 'Id'.
- Choose Week:** Two dropdown menus labeled 'Week From' and 'Week To', both currently set to '0'.
- Details:** A section containing four text input fields: 'Name of the exercise', 'Details', 'Frequency', and 'URL'.

At the bottom right of the form, there are two buttons: 'Review' and 'Add'.

Figure 5.28 Add protocol page

Protocol

Protocol Id #1001 Date: 2016-Apr-19

Line	Week From	Week To	Name	Details	Frequency	URL
1	0	1	Codmans	3 times daily, wear shoulder sling at all times except when exercising and for hygiene	2-3 times per week	0
2	0	1	Elbow ROM	3 times daily	2-3 times per week	https://www.youtube.com/embed/JoTnU7IGWU
3	0	1	Grip Strengthening	3 times daily	2-3 times per week	0
4	0	1	Wrist ROM	3 times daily	2-3 times per week	https://www.youtube.com/embed/VGrbAsSj3gk
5	1	6	Full ROM	tolerated with passive stretching at end ranges	2-3 times per week	https://www.youtube.com/embed/VGrbAsSj3gk
6	1	6	Grip Strengthening	3 times daily	2-3 times per week	0
7	6	12	AAROM to AROM	ROM goals: 140 FF/40 ER at side; ABD max 60-80 without rotation, can increase as tolerated	2-3 times per week	0
8	6	12	Scapular Exercises	for large muscle groups (pecs, lats, etc)	2-3 times per week	0
9	6	12	Passive Resistance Exercises	for large muscle groups (pecs, lats, etc)	2-3 times per week	0
10	12	48	Full ROM	tolerated with passive stretching at end ranges	2-3 times per week	0
11	12	48	Advance strengthening	isometrics->bands->light weights(1-5 lbs); 8-12 reps/2-3 sets per rotator cuff, deltoid, and scapular stabilizers	2-3 times per week	0
12	12	48	Eccentrically Resisted Motions	2 times daily	2-3 times per week	0
13	12	48	Phymetrics	weighted ball toss	2-3 times per week	0
14	12	48	Proprioception	body blade	2-3 times per week	0
15	12	48	Sports Related Rehab	four and half months, including advanced conditioning	2-3 times per week	0

Figure 5.29 Review protocol page

5.3.5 Health Monitoring Module

Health monitoring module consists of four dynamic tables, and it implements send in-app message functions as well. Physicians can only access to the authorized patients' data. In patient info page, physicians can get full access to patients' personal information including email address, name, gender, date-of-birth, phone number, state, and zip-code. They can also send in-app message by clicking Compose new button placed on top right corner, as it is shown in Figure 5.30. Since physicians enter the email address of the patients when they compose a message, after message being sent, patients will receive an email notification about their new message.

In patient health data page, physician can look at patients' physiological data including active heart rate, blood pressure, body fat, calories burned, distance, duration, floors,

resting heartrate, sleep, spo2, steps, weight, water, calories consumed and glucose. All the health data are encrypted not only during the transmission but also in database.

Patient progress page shows the feedback about patients' everyday exercises. It includes finished percentage, date, comments and feeling. Protocols page are designed for physicians to see all of the rotator cuff repair and acl reconstruction protocols. The sending email page is shown in Figure 5.30.

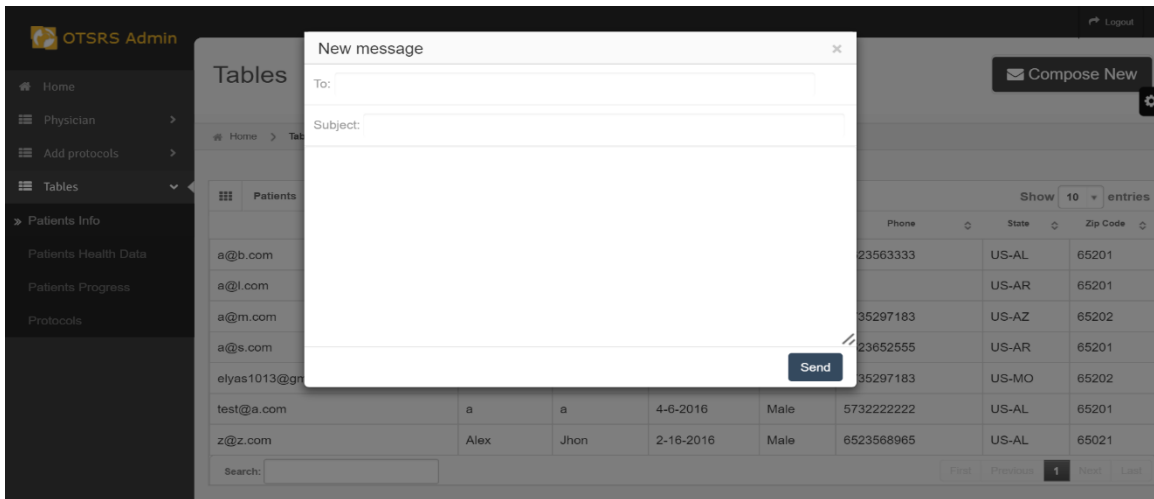


Figure 5.30 Physician sending email page

6. Summary and Future Work

In the future, for API authentication module, more and more wearable sensors APIs such as Garmin and mi Band will be intergraded. In this way, API documents of both sensors needs to be read; besides, new API integration, data collection and data uploading classes should be implemented for both the mobile computation module and server module. For more lasting experience, battery saving methods will be applied to OTSRS app to optimize the device battery life. OTSRS application dashboard module will be upgraded and added support to show bar chart of each field (seven days data).

New social networking module will be developed to support the communication between patients. Patients could be able to share their experience, stories and good methods to other patients. At the same time, progress module will also be upgraded accordingly to a new look. Patients not only can see the days that they have finished, they also can see the days that they have missed and days left till full recovery; moreover, that could be able to compete with other patients their contact list in the application.

For web server module, data visualization will be applied to health monitoring module, and more google chart API will be added to show bar chart of each patients' all-days health data. Most importantly, the OTSRS system is expendable. It could later be used to track the health status of diabetic patients and people with cardiovascular heart diseases. It could also be used as health assessment systems by implementing different surveys.

This thesis describes the design and implementation of an orthopedic surgery rehabilitation and health monitoring systems especially for patients who have done Rotator cuff repair and ACL reconstruction surgeries. As opposed to traditional method, where patients are needed either to go to physical therapy or to read through long instructional paper, and also required to submit their health status after the release of hospital manually. The presented rehabilitation system is much more efficient, money saving and real. Data doesn't lie. The presented rehabilitation and health monitoring system is implemented across the web service and has support of wearable sensors. Physicians can keep track of patients everyday exercises compliance and also can look at their real physiological data. In current stage, the prototype of the system is developed. Mobile computation module, the OTSRS application, has been deployed to different kinds of android devices, and tested successfully. All web service is running stable. API versions of both Fitbit and iHealth are upgraded to the latest version to get full support from wearable sensor.

References

1. Kozlovsky, Miklos, et al. "Personal health monitoring with Android based mobile devices." *Information & Communication Technology Electronics & Microelectronics (MIPRO)*, 2013 36th International Convention on. IEEE, 2013.
2. Kumar, Maradugu Anil, and Y. Ravi Sekhar. "Android based health care monitoring system." *Innovations in Information, Embedded and Communication Systems (ICIIECS)*, 2015 International Conference on. IEEE, 2015.
3. "IDC: Smartphone OS Market Share." [Www.idc.com](http://www.idc.com). IDC Analyze The Future, Aug. 2015. Web. 11 Apr. 2016.
4. "Usage of Server-side Programming Languages for Websites." *Usage Statistics and Market Share of Server-side Programming Languages for Websites*, April 2016. W3Techs, Jan. 2016. Web. 11 Apr. 2016.
5. Sabani, Azzizatul Huda, and R. Jailani. "Android based control and monitoring system for leg orthosis." *Signal Processing & Its Applications (CSPA)*, 2015 IEEE 11th International Colloquium on. IEEE, 2015.
6. Caulfield, Brian, et al. "Rehabilitation exercise feedback on Android platform." *Proceedings of the 2nd Conference on Wireless Health*. ACM, 2011.
7. Deponti, Dario, Dario Maggiorini, and Claudio E. Palazzi. "DroidGlove: an android-based application for wrist rehabilitation." *Ultra Modern Telecommunications & Workshops, 2009. ICUMT'09. International Conference on*. IEEE, 2009.
8. Du, Yuanyuan, et al. "An android-based emergency alarm and healthcare management system." *IT in Medicine and Education (ITME)*, 2011 International Symposium on. Vol. 1. IEEE, 2011.
9. "Model–view–controller." *Wikipedia*. Wikimedia Foundation. Web. 13 Apr. 2016.
10. Hardt, Dick. "The OAuth 2.0 authorization framework." (2012).
11. Boyd, Ryan. *Getting started with OAuth 2.0*. "O'Reilly Media, Inc.", 2012.

12. Burnette, Ed. Hello, Android: introducing Google's mobile development platform. Pragmatic Bookshelf, 2009.
13. Ermes, Miikka (January 2008). "Detection of Daily Activities and Sports With Wearable Sensors in Controlled and Uncontrolled Conditions". IEEE.
14. "Web API Documentation." - *Fitbit Web API Docs*. Web. 17 Feb. 2015.
15. "Put Your Health Front and Center with IHealth Mobile Products - IHealth." *Put Your Health Front and Center with IHealth Mobile Products - IHealth*. Web. 01 Feb. 2015.
16. "Android Developer Documents" *Android Developers*. Google. Web. 18 Apr. 2016.
17. "Advanced Encryption Standard." Wikipedia. Wikimedia Foundation. Web. 19 Feb.