SDSU GOGGLES ANDROID APPLICATION

A Thesis

Presented to the

Faculty of

San Diego State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

by

Sunilkumar K. Lakhiyani

Spring 2012
SAN DIEGO STATE UNIVERSITY

The Undersigned Faculty Committee Approves the

Thesis of Sunilkumar Lakhiyani:

SDSU Goggles Android Application

Joseph Lewis, Chair
Department of Computer Science

Kris Stewart
Department of Computer Science

Carmelo Interlando
Department of Mathematics and Statistics

10-27-11
Approval Date
Copyright © 2012
by
Sunilkumar K. Lakhiyani
All Rights Reserved
DEDICATION

I dedicate this thesis work to my dear parents, my lord master and all the members of Universal Brotherhood Mission for their constant encouragement and their unconditional love which they showered upon me every time I needed.
This era is well known for various types of Smartphones, Tablets and IPods and IPads. Smartphones or Tablets helps you to stay organized, flexible and provides faster but secure communication. It also helps you to stay in touch with your friends and relatives easily. Smartphones run complete operating system software providing a platform for application developers. The SDSU Goggles is an android smartphone based application which provides the functionality of Rate and Review system. This functionality is capable of reviewing and rating professors’ teaching skills, assignments and his/her proficiency for particular subject or class. Similarly, user can rate and review particular subject. This paper describes an experiment which integrated the application core framework into java language using android operating system, a program that support UI and service layer of a smartphone application, stores all the information into database, finally ‘Data’ which flows between server and application in JSON format.

This application is user friendly. To use this application your smartphone must be connected with internet using Wi-Fi or 3G connection. After welcome screen user will see two options Search Professor or Search Subject. Using this functionality user can search professor by firstname or lastname. Similarly they can search subject by number. When user click on Search, application talks to server and finds all the professor or subject in to the list format. User need to decide which subject he/she wants to rate or review. After selection application shows them all the information of selection. Then user has 2 options review selection or rate selection. After rating and reviewing their selection application redirects them to information page again, where they can see their review and average rating. An interesting part of this application is very light weight communication in communication bus. I have used JSON. The benefits of JSON are it is smaller than XML and faster to work with, very efficient parsing, and many more.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
</tr>
<tr>
<td>ABBREVIATIONS</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
</tr>
<tr>
<td>CHAPTER</td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
</tr>
<tr>
<td>1.1 Objective</td>
</tr>
<tr>
<td>1.2 Need for an Application</td>
</tr>
<tr>
<td>2 BACKGROUND RESEARCH AND TECHNOLOGY</td>
</tr>
<tr>
<td>2.1 Android Framework</td>
</tr>
<tr>
<td>2.1.1 Android Features</td>
</tr>
<tr>
<td>2.1.2 Libraries</td>
</tr>
<tr>
<td>2.1.3 Application Frame Work</td>
</tr>
<tr>
<td>2.1.4 Applications</td>
</tr>
<tr>
<td>2.1.5 Actual Interaction with Development Environment</td>
</tr>
<tr>
<td>2.2 Android Framework</td>
</tr>
<tr>
<td>2.2.1 Smartphone User</td>
</tr>
<tr>
<td>2.2.2 Smartphone Server</td>
</tr>
<tr>
<td>2.2.3 Android Resources</td>
</tr>
<tr>
<td>2.2.4 Resource Table</td>
</tr>
<tr>
<td>2.2.5 Android Source Layer</td>
</tr>
<tr>
<td>2.3 Tomcat</td>
</tr>
<tr>
<td>2.3.1 Platform</td>
</tr>
<tr>
<td>2.3.2 JSON Parsing Tool</td>
</tr>
<tr>
<td>2.4 Java Servlet</td>
</tr>
<tr>
<td>2.5 MySQL</td>
</tr>
</tbody>
</table>
6  FUTURE PLAN...........................................................................................................45
REFERENCES ........................................................................................................................46
LIST OF TABLES

Table 2.1. Directory Structure of Android App .................................................................8
Table 3.1. Template for Describing Requirements .........................................................13
Table 3.2. Search Professor ......................................................................................13
Table 3.3. Search Subject ......................................................................................14
Table 3.4. Rate My Professor ...............................................................................14
Table 3.5. Rate My Subject ..................................................................................14
Table 3.6. Review My Professor ........................................................................14
Table 3.7. Review My Subject ...........................................................................15
Table 3.8. Department .........................................................................................15
Table 3.9. Professor Comments Table .................................................................15
Table 3.10. Subject Comments Table .................................................................15
Table 3.11. Professor Rating Table .................................................................16
Table 3.12. Subject Rating Table .................................................................16
Table 3.13. Professor Information Table ..........................................................16
Table 3.14. Subject Information Table ..............................................................16
LIST OF FIGURES

PAGE

Figure 2.1. Android OS architecture..........................................................................................4
Figure 2.2. Actual interaction between smartphone server android framework and smartphone app..............................................................6
Figure 2.3. Basic client server communications........................................................................9
Figure 4.1. Android phone summary.....................................................................................29
Figure 4.2. Wi-Fi settings in android smartphone....................................................................30
Figure 4.3. SDSU goggles’ landing page..................................................................................33
Figure 4.4. Search subject screen..........................................................................................34
Figure 4.5. Search professor “Joseph”....................................................................................35
Figure 4.6. List of professor first name as Joseph....................................................................36
Figure 4.7. Professor detail page..........................................................................................37
Figure 4.8. Add review screen..............................................................................................38
Figure 4.9. Detail page with new comments..........................................................................39
Figure 4.10. Search subject....................................................................................................40
Figure 4.11. Subject detail screen........................................................................................41
Figure 4.12. Rate my class screen.........................................................................................42
Figure 4.13. Detail of class with new avg rate.........................................................................43
ABBREVIATIONS

API: Application Programming Interface
HTTP: Hyper Text Transfer Protocol
JSON: JavaScript Object Notation
OS: Operating System
REST: Representational State Transfer
SDK: Software Development Kit
UI: User Interface
URI: Uniform Resource Identifier
XML: Extensible Markup Language
GSM: Global System for Mobile Communication
Wi-Fi: Wireless Fidelity
GPS: Global Positioning System
JPEG: Joint photographic experts group
PNG: Portable network graphics
GIF: Graphics interchange format
ACKNOWLEDGEMENTS

I am heartily thankful to my advisers Dr. Joseph Lewis, Dr. kris Stewart, and Professor Interlando for their encouragement, support, guidance, and continued supervision. I have presented various approaches to Dr. Lewis to deal with various problems and difficulties which arise during application development. With his guidance and support we dealt successfully with various problems which appeared during the course of this thesis work. His constant encouragement and discussion with various approaches helped me a lot to develop this application.
CHAPTER 1

INTRODUCTION

1.1 OBJECTIVE

The main objective of this thesis is to develop an android based smartphone application which supports both Rating and Review of class as well as professor. Thus, before semester starts students come to know which subject is appropriate for them and they can take their appropriate subject by contacting respective professors. This application is not only just supports reviewing and rating system, but it does support search functionality, too. That means before rate or review any subject/class or professor user can search particular professor or particular subject. So as the result user can see list of professor or subject. In this way user can select particular professor or class and rate and/or review respective selection.

Since, the application developed for smartphones, it has to have effective and attractive user interface (UI). Most of the smartphone application is user friendly, to achieve this feature I have created attractive user interface with support of android user interface features.

Development and testing of this application performed on Android based platform. Android framework has tons of functionalities among those this application mainly uses HTTP communication with Tomcat server, JSON parsing using GSON, various layouts and many more to support this application.

1.2 NEED FOR AN APPLICATION

Advance technology makes possible many things. Today phones are getting smaller and more advance. The difference between smartphones and regular phones is that a smartphone offers more and more functionalities, e.g. email, internet browsing, ebook reader, and many more. The heart of smartphone is applications. To use these applications effortlessly all the factors together add up in single application. This decides if you need a smartphone or a regular phone. Applications come in different forms, e.g. Tourist buddy,
Calculator, Splitcost, Electrical Vehicle Applications and many more. These applications are giving you an almost the PC-like feeling, and features. Nowadays more and more applications being made for smartphones. [1]

This application is also based on the Android smartphone. Nowadays as I have already mentioned that smartphones are very common thing, and plays an important role in day to day life of student or more specifically life of human beings. At the beginning of the semester students don’t know which class is better and/or what is the teaching style of professor. This application gives them an idea about subject and the professor. This application allows them to read comments written by other student, too. So, atleast at the start of semester they have idea how is the subject and whether that subject is the one which they are looking for, or is that subject all about their interest? Another thing about this application is that, application doesn’t require any kind of information from the user so they feel free to write comments and reviews. This particular type of application offer free rating and reviews to help others find out what people are thinking and/or saying about the subject and professor. It will take few minutes or just one click to check out what others are saying about that subject and you can decide which subject you will take during the semester.
CHAPTER 2

BACKGROUND RESEARCH AND TECHNOLOGY

2.1 ANDROID FRAMEWORK

Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and API necessary to begin developing applications on the Android platform using the Java programming language. Using Android platform developer can generate UI/ layout via GUI tool supported by operating system and XL programming language. Android SDK comes with emulator for app testing. As a developer I need to first define or create virtual devices which match screen size and API version.

2.1.1 Android Features

1. Application framework - enabling reuse and replacement of components
2. Dalvik virtual machine - optimized for mobile devices
3. SQLite - for structured data storage
4. GSM Telephony (hardware dependent)
5. Bluetooth, EDGE, 3G, and WiFi (hardware dependent)
6. Camera, GPS, compass, and accelerometer (hardware dependent)
7. Rich development environment including a device emulator, tools for debugging, memory and performance profiling, and a plugin for the Eclipse IDE

The android SDK supports most of the java platform standard edition. But, it does not supports AWT abstract windows toolkit and java swing. For the UI android has provided UI framework. The android platform comes with everything you need. Surprisingly in one package, only. This package contains the OS, core libraries, Dalvik VM, java development environment and device drivers.

As android provides java development environment usually it compare with JAVAME. Java ME platform is a collection of technologies and specifications that can be combined to construct a complete Java runtime environment specifically to fit the requirements of a particular device or market. This offers a flexibility and co-existence for all
the players in the eco-system to seamlessly cooperate to offer the most appealing experience for the end-user [2].

So here is a comparison.

- J2ME has multiple UI models for different UI models, while android has only one model for multiple UI model.
- J2ME supports MIDlets, Xlets and AWT, while android has its own UI framework, so you can create UI easily. And it is most under stable to read it.
- The Dalvik VM is more optimized and more responsive than JVM.
- Android runs .dex byte-codes instead of java byte-codes
- Ease of development in android compare to J2ME

You can see major components of the Android operating system in Figure 2.1. Each section is described in more detail below. [3]

![Android OS architecture](image)

**Figure 2.1. Android OS architecture.**

The base of android platform is the Linux kernel. This is responsible for all the OS duties and other miscellaneous stuff. It also supplies the device drivers, for example Camera.
keypad, wifi, audio and more. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack.

Moving to next level, on the top of the kernel Android Runtime. It includes a bunch of core libraries that provides most of the functionality available in the core libraries of the Java programming language. Most of the application framework accesses these core libraries through the Dalvik virtual machine.

It can run multiple instances of VMs (Virtual Machines). Each application gets its own VM. The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management.

### 2.1.2 Libraries

It includes a set of C/C++ libraries. Various components of the Android use them. These libraries includes

1. **System C library** – Standard C Libraries. Which allow to use Linux based device.
2. **Media Libraries** – These libraries supports all the media features such as Audio, Video and Images.
3. **Surface Manager** – Supports 2D and 3D graphics composite layer.
4. **LibWebCore** – Supports modern web surfing and fast internet access.
5. **SQLite** - a powerful and lightweight relational database engine available to all applications

### 2.1.3 Application Frame Work

Android provides an open development platform. This is the reason for developer to build innovative applications. Developers are free to take advantage of device hardware, access local information, run background service, alarm, calling functions and many more. Developers have full access to the same framework APIs used by the core applications. This same mechanism allows components to be replaced by the user.

### 2.1.4 Applications

Android ships with many applications. Examples are Email, google talk, android market place, and more.
2.1.5 Actual Interaction with Development Environment

Figure 2.2 explains how actual interaction happens between android application and Android OS development environment.

![Diagram of actual interaction between smartphone server android framework and smartphone app.](image)

**Figure 2.2.** Actual interaction between smartphone server android framework and smartphone app.

2.2 ANDROID FRAMEWORK

Start with base of development. The base of android applications development is off course android framework. As explain above Android framework provides all the feature, development environment, core libraries and Android application framework to make innovative and efficient applications. For ease understanding I have divided the application layer into development environment.
2.2.1 Smartphone User

User of my application is the one who has Android smartphone with Android OS 2.1 and above.

2.2.2 Smartphone Server

Smartphone server is the server which provides the web-services to interact with Android smartphone. Smartphone application request and receive service from the centralized server. My application provides and interface to allow user to request particular service (add comments or add ratings) and display the result the server returns. Server waits for request and as soon as request arrives it start process and create response for the client and send back response in XML or JSON format to client. It provides an interface to clients so clients need not be worry about the specification of the system.

2.2.3 Android Resources

You should always externalize application resources such as images and strings from your code, so that you can maintain them independently. After externalize your application resources you can access them using resource IDs. This ids are stored in R.java file. You should place each type of resource in a subdirectory. The directory structure is defined in Table 2.1.

2.2.4 Resource Table

Following table shows how project structure will be setup for developer in Eclipse. This all directories are belongs to res folder. Android generates R.java file to use resources in java file. We can find resources using ids [4] Please see following resource table.

2.2.5 Android Source Layer

I divide this layer into 2 parts (i) Android UI Layer and (ii) Android Service Layer.

1. Android UI Layer : This layer contains Application specification Resource APIs. As I’ve mentioned earlier android application use JAVA programming language to create Activity, Intent, Services and AIDL interfaces. So this layer contains different java files which allows user to start an application and show them result. These views render the model into a form suitable for interaction, typically a user interface element. Multiple views can exist for a single model for different purposes. Also, User can navigate screens easily from one place to another.
Table 2.1. Directory Structure of Android App

<table>
<thead>
<tr>
<th>Directory</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>res/anim</td>
<td>Store XML files which supports twinning animation.</td>
</tr>
<tr>
<td>res/color</td>
<td>Color files which defines list of colors</td>
</tr>
<tr>
<td>res/drawable</td>
<td>Bitmaps or XML files those are complied into drawable directories.</td>
</tr>
<tr>
<td>res/layout</td>
<td>XML files that defines a user interface.</td>
</tr>
<tr>
<td>res/menu</td>
<td>XML files defines menu.</td>
</tr>
<tr>
<td>res/values</td>
<td>XML files that contain simple values such as string, color, arrays, styles. Supports Internationalization.</td>
</tr>
<tr>
<td>res/xml</td>
<td>Arbitrary XML files that can be read at runtime by calling Resources.getXML().</td>
</tr>
</tbody>
</table>

2. Android Service Layer: My application is client server application. For each and every functionality it ask user to get results from server and display on screen. In these type of cases or scenarios Android service layers comes into the picture. The service layer contains Smartphone Object Model also. Service layer has 3 parts (a) first part contains the service interface and implementation (b) second part contains smartphone object models and (c) last part contains configuration. Service interface provides the interface to get input from user and initiates request for server. The service implementation part transfer request to server and accept response from server. If we get valid response, then the smartphone object model manages the behavior and data of the application domain. Object model stores the data which UI layer wants. This all are background processes, User doesn’t know about this background process. If we do not get results I mean if we get exception or some garbage value from server then configuration part comes into a picture. It handles all the exception, all the error at client side and display appropriate messages to user.

Move on last but most important and difficult part of my application is to make smartphone server. My application is basically client server application. My client is SDSU Goggles Android Application which allows user to see the comments and rating of professor and subjects. So my application deals with lots of data, and public interaction. So I have decided to go with the server which provides me to write web services for client which requests and get response back. Client server describes the relationship between 2 programs.
In a network the client server architecture provides a suitable way to interconnect each other even if they are distributed in different locations.

Figure 2.3 [5] gives you basic idea about how client and server communicate to each other.


As you can see in the figure, Client is a smartphone device which interacts with server using HTTP protocol. Server communicates with MySQL to get data and make response. After making response it sends validate/invalid data or error message back to client. Client gets response and generates appropriate results for users. For SDSU Goggles I have used Tomcat server, MySQL as database storage, Java servlet and JSON data exchange format for lightweight and speedy communications. Detail information of how to implement client server architecture is covered in following chapters.
2.3 TOMCAT

Apache Tomcat server is one of the most popular open source web servers that implement the Java Servlet and the Java Server Pages (JSP) specifications from Sun Microsystems to provide the platform to run Java code on a web server provided by Apache Software Foundation. Tomcat adds tools of its own internal HTTP server and can also add tools for configuration and management. It can also be configured by editing configuration files usually written in XML format. Apache Tomcat is intended to be a collaboration of the best developers from around the world that supports only web components [6].

SDSU Goggles also provides web services, which is nothing but method of communication between two electronic devices over a network. Here two electronic devices are (i) an android smartphone where application client is installed and (ii) a PC or a machine where application server is installed. Tomcat is also called servlet container. The Java Servlet API provides web developers with a simple but very robust mechanism for extending functionality of web server. There are following reasons to choose tomcat server for SDSU Goggles.

1. Tomcat is very light weight application server.
2. It is easily configured with IIS and Apache.
3. It doen’t required more memory at startup.
4. It is an open source server.
5. (v) very stable on UNIX like system.

2.3.1 Platform

The basic web services platform for this project is JSON+HTTP. HTTP stands for Hypertext Transfer Protocol. It provides functionality of request-response in client-server communication model. Here again I need to mentioned that our client i.e. SDSU Goggles submits an HTTP request message to the server i.e. Smartphone Server. The server, which communicates to database to get data, provides resources, performs necessary function behalf of client, does some calculation if required and replies back with response to the client. A response contains status information about the request and message in message body. HTTP protocol is designed in such a way that it permits intermediate network elements for communication. HTTP is an Application Layer protocol. Its resources are identified and located on the network by URLs or URIs.
### 2.3.2 JSON Parsing Tool

JSON stands for JavaScript Object Notification is a lightweight data-interchange format, which can be used between different platforms and programming languages and still express complex message and functions. JSON is a syntax for passing around object that contains name/value pairs, arrays and other objects.

1. Similarity between XML and JSON
2. JSON and XML both have values within values.
3. JSON and XML both can be parsed and used by many programming languages use
4. JSON and XML both can work efficiently with HTTP framework.
5. JSON and XML both are self describing, both have name value pairs, thus Human readable.

### 2.4 JAVA SERVLET

Early in the internet world the CGI, the common gateway interface was defined to allow web servers to process user input and serve dynamic content. Developers can write CGI scripting in any language for example Perl, PHP, and many more. But there are some drawbacks such as Performance and scalability. New process is created for each request, so its very difficult to server to response quickly. Database connection between multiple calls to
the same scripts is far from trivial. It is something like repeated execution of operations. There is another drawback called Security. Most of scripts use the command shell to execute operation system commands with user input, data and more information. This opens many chances for a person who can hack and explodes malfunction into CGI scripts [7].

On the other hand the servlet API was developed to force the advantages of the JAVA platform to solve the issues of CGI scripts. All web servers supports servlet API. Instead of creating a process like CGI scripting, servlet creates a thread for each request. Servlet can share resources. Security is improved, for example hardly ever need to allow a shell execute commands with user-supplied data, the reason behind this is Java API, which provides access to commonly used functions. Java's exception handling also makes a servlet more reliable. For example a divide by zero, Null pointer exception or Array index out of bound is reported as an error instead of crashing the Web server. Java servlet engine required to execute Java servlet class. This engine loads the servlet class the first time the servlet is requested, and then it is remains loaded to handle multiple request until servlet engine shuts down. Apache tomcat requires a servlet engine add-on module. This module catches all the requests for servlet, executes them and returns the response through the web server to the client.

2.5 MySQL

MySQL is currently very popular open source database server. There are 2 structure if database, (i) Flat database and (ii) relational database. Relational database is more structured and most oriented. MySQL is a relational database. In a relational structured database there are tables that store data. The columns define which kinds of information will be stored in the table. A row contains the actual values for these specified columns. Each row will have 1 value for each and every column.
CHAPTER 3

SERVER IMPLEMENTATION

3.1 REQUIREMENT

This application requires to store all the professor details in the database, rating and reviews of particular professor and also manipulation of data. Thus application server is required which provides web services and communicates with database efficiently and capable to do communication efficiently with client in our case android application. Database used to store all the information of professor.

Tables 3.1-3.7 gives you idea about requirement of this application.

Table 3.1. Template for Describing Requirements

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Reason for requirement.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>Input for requirement.</td>
</tr>
<tr>
<td>Processing</td>
<td>The processing needs that have to occur to fulfill the requirement.</td>
</tr>
<tr>
<td>Output</td>
<td>What results can be expected, including all output sources.</td>
</tr>
</tbody>
</table>

Table 3.2. Search Professor

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Retrieve Professor Information from Database.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>Professor’s First Name and/or Last Name.</td>
</tr>
<tr>
<td>Processing</td>
<td>Information of professor will be fetched from the database on the basis of the given user inputs.</td>
</tr>
<tr>
<td>Output</td>
<td>List of Professors’ information containing First name, Last name, Department, Photo, average rating, and comments, or an error message.</td>
</tr>
</tbody>
</table>

3.2 DATABASE DESIGNING

Database Designing: Tables 3.9-3.14 give you detail information about database designing used in the app.
Table 3.3. Search Subject

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Retrieve Subject Information from Database.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>Subject Number</td>
</tr>
<tr>
<td>Processing</td>
<td>Information of subject will be fetched from the database on the basis of the given user inputs.</td>
</tr>
<tr>
<td>Output</td>
<td>List of subjects’ information containing Subject Number, Subject Name, Department, average rating, and comments, or an error message.</td>
</tr>
</tbody>
</table>

Table 3.4. Rate My Professor

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Rate My Professor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>Rating between 1to 5 stars</td>
</tr>
<tr>
<td>Processing</td>
<td>When user click on Submit Button, the professor ID and users’ rating will be store to database and average rating will be calculated.</td>
</tr>
<tr>
<td>Output</td>
<td>Professor’s information containing professor ID and average ratings. Other information will be retrieved by stored list.</td>
</tr>
</tbody>
</table>

Table 3.5. Rate My Subject

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Rate My Subject</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>Rating between 1to 5 stars</td>
</tr>
<tr>
<td>Processing</td>
<td>When user click on Submit Button, the subject number and users’ rating will be store to database and average rating will be calculated.</td>
</tr>
<tr>
<td>Output</td>
<td>Subject’s information containing subject number and average ratings. Other information will be retrieved by stored list</td>
</tr>
</tbody>
</table>

Table 3.6. Review My Professor

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Write a review for my professor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>Review/ comments about professor.</td>
</tr>
<tr>
<td>Processing</td>
<td>When user click on Submit Button, the professor ID and users’ review will be store to database.</td>
</tr>
<tr>
<td>Output</td>
<td>Professor’s information containing Professor ID and comment. Other information will be retrieved by stored list.</td>
</tr>
</tbody>
</table>
Table 3.7. Review My Subject

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Write a review for my subject</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>Review/ comments about subject.</td>
</tr>
<tr>
<td>Processing</td>
<td>When user click on Submit Button, the subject number and users’ review will be store to database.</td>
</tr>
<tr>
<td>Output</td>
<td>Subject’s information containing subject number and comment. Other information will be retrieved by stored list</td>
</tr>
</tbody>
</table>

Table 3.8. Department

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Primary Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sdsugoggles_department_id</td>
<td>Varchar(10)</td>
<td>Yes</td>
<td>Department Id</td>
</tr>
<tr>
<td>sdsugoggles_department_name</td>
<td>Varchar(50)</td>
<td>Yes</td>
<td>Department Name</td>
</tr>
</tbody>
</table>

Table 3.9. Professor Comments Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Primary Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Int(10)</td>
<td>Yes</td>
<td>DB id of professors’ comment</td>
</tr>
<tr>
<td>sdsugoggles_prof_id</td>
<td>Varchar(40)</td>
<td>No</td>
<td>Professor Id</td>
</tr>
<tr>
<td>sdsugoggles_prof_comments</td>
<td>Longtext</td>
<td>No</td>
<td>Comments about professor</td>
</tr>
</tbody>
</table>

Table 3.10. Subject Comments Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Primary Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Int(10)</td>
<td>Yes</td>
<td>DB id of subjects’ comment</td>
</tr>
<tr>
<td>sdsugoggles_sub_number</td>
<td>Varchar(40)</td>
<td>No</td>
<td>Subject number</td>
</tr>
<tr>
<td>sdsugoggles_sub_comments</td>
<td>Longtext</td>
<td>No</td>
<td>Comments about subject</td>
</tr>
</tbody>
</table>
### Table 3.11. Professor Rating Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Primary Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Int(10)</td>
<td>Yes</td>
<td>DB id of professors’ rating</td>
</tr>
<tr>
<td>sdsugoggles_prof_id</td>
<td>Varchar(40)</td>
<td>No</td>
<td>Professor Id</td>
</tr>
<tr>
<td>sdsugoggles_prof_rate</td>
<td>Decimal(10,2)</td>
<td>No</td>
<td>Rating of professor</td>
</tr>
</tbody>
</table>

### Table 3.12. Subject Rating Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Primary Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Int(10)</td>
<td>Yes</td>
<td>DB id of subjects’ comment</td>
</tr>
<tr>
<td>sdsugoggles_sub_number</td>
<td>Varchar(40)</td>
<td>No</td>
<td>Subject number</td>
</tr>
<tr>
<td>sdsugoggles_sub_rate</td>
<td>Decimal(10,2)</td>
<td>No</td>
<td>Rating of subject</td>
</tr>
</tbody>
</table>

### Table 3.13. Professor Information Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Primary Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sdsugoggles_prof_id</td>
<td>Varchar(40)</td>
<td>Yes</td>
<td>Professor’s Id</td>
</tr>
<tr>
<td>sdsugoggles_prof_firstname</td>
<td>Varchar(50)</td>
<td>No</td>
<td>Professors’ First name</td>
</tr>
<tr>
<td>sdsugoggles_prof_lastname</td>
<td>Varchar(50)</td>
<td>No</td>
<td>Professors’ Last name</td>
</tr>
<tr>
<td>sdsugoggles_prof_deptid</td>
<td>Varchar(10)</td>
<td>No</td>
<td>Professors’ department</td>
</tr>
<tr>
<td>sdsugoggles_prof_pic</td>
<td>Longblob</td>
<td>No</td>
<td>72X72.png photo of professor.</td>
</tr>
</tbody>
</table>

### Table 3.14. Subject Information Table

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Primary Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sdsugoggles_sub_id</td>
<td>Varchar(40)</td>
<td>Yes</td>
<td>Subject Id</td>
</tr>
<tr>
<td>sdsugoggles_sub_name</td>
<td>Varchar(50)</td>
<td>No</td>
<td>Subjects’ Name</td>
</tr>
<tr>
<td>sdsugoggles_sub_deptid</td>
<td>Varchar(10)</td>
<td>No</td>
<td>Subject’s department Id</td>
</tr>
</tbody>
</table>
3.3 Server Design Architecture

The server implementation is divided into 3 packages. These packages are

1. `edu.sdsu.master.thesis.sdsugoggles.dbhelper`
2. `edu.sdsu.master.thesis.sdsugoggles.communicator`
3. `edu.sdsu.master.thesis.sdsugoggles.model`

3.3.1 Detail Information of all the Packages

(i) `edu.sdsu.master.thesis.sdsugoggles.dbhelper`: This package contains `SQLConnection` class which is mainly provides the connectivity between MySQL server and other model in server.

```java
public class SQLConnection {
    //Initialization of required fields
    private static final String url = "jdbc:mysql://localhost/"+dataBase;

    //open Connection.
    public Connection getConnection() throws Exception {
        Class.forName("com.mysql.jdbc.Driver").newInstance();
        liveConnection = DriverManager.getConnection(url, userName, password);
        return liveConnection;
    }

    //Create Statement
    public Statement getStatement() throws Exception {
        Connection c = getConnection();
        return c.createStatement();
    }

    //Close connection.
    public void closeConn(Connection ob) {
        if(ob != null) {
            try { ob.close(); } catch(Throwerable exception) {} 
        }
    }
}
```

3.3.2 Establish a Connection

Class `forName("com.mysql.jdbc.Driver")` causes the class named `com.mysql.jdbc.Driver` to be dynamically loaded at runtime. A call to `forName("com.mysql.jdbc.Driver")` causes the class named `com.mysql.jdbc.Driver` to be initialized.
`Class.forName("com.mysql.jdbc.Driver")` new `Instance()` returns the `Class` object associated with the "com.mysql.jdbc.Driver" class. `Class.forName("com.mysql.jdbc.Driver")` loads the class if it not already loaded. The JVM keeps track of all the classes that have been previously loaded. This method uses the `classloader` of the class that invokes it.

`DriverManager.getConnection()` attempts to establish a connection to the given database URL. The `DriverManager` attempts to select an appropriate driver from the set of registered JDBC drivers. `url` is nothing but a database URL while `user` is the username and password is the user’s password. If connection is Successful then its returns the connection to the URL. If error occurs then it throws `SQLException`.

### 3.3.3 Create Statement

A `Statement` object is used to send SQL statements to the DBMS, and should not to be confused with an SQL statement. A JDBC `Statement` object is associated with an open connection, and not any single SQL Statement. An active connection is needed to create a `Statement` object.

```java
Statement stmt = con.createStatement(); // which create statement to communicate with DBMS.
```

At this point, a `Statement` object exists, but it does not have an SQL statement to pass on to the DBMS.

### 3.3.4 Close Statement

Releases active Connection object's database and JDBC resources immediately instead of waiting for them to be automatically released. Calling the method close on a Connection object that is already closed is a no-op.

This is database connection client which provides basic functionality of database connection. Other classes can create which will use in other class to communicate with database.

(ii) `edu.sdsu.master.thesis.sdsugoggles.communicator`: This package contains all the classes which communicate with database using servlet API. All the classes in this package is extends the `HttpServlet` class.

A servlet is an extension to a server that improves server's functionality. The most common use for a servlet is to extend a web server by providing dynamic web content. HTTP
is the internet protocol which I have used to access web-services. **HTTPServlet** provides an abstract class to be subclassed to create an HTTP servlet suitable for a Web site.

### 3.4 SERVLET

```java
public class SearchProfessorCommunicator extends HttpServlet {
    /**
     * @author Hardevji
     */
    private static final long serialVersionUID = 1L;
    private static final String dataTable = "sdsugoggles";
    private Gson gson = new Gson();
    private SQLConnection db = null;
    private Connection conn = null;
    private ResultSet rs = null;
    private PreparedStatement st = null;

    @Override
    public void doPost(HttpServletRequest req, HttpServletResponse res)
        throws ServletException, IOException {
        doGet(req, res);
    }

    @Override
    public void doGet(HttpServletRequest req, HttpServletResponse res)
        throws ServletException, IOException {
        try {
            ProfessorListRequest professorListRequest = populateRequestParametersForList(req);
        } catch (Exception e) {
            throw new RuntimeException();
        }

        List<Professor> professorList = null;
        try {
            professorList = getListofProfessor(professorListRequest);
        } catch (Exception e1) {
            e1.printStackTrace();
        }

        ProfessorListResponse professorListResponse = new ProfessorListResponse();
        professorListResponse.setProfessorList(professorList);
        createResponseForResponse(professorListResponse, res);
        displayResponse(professorList);
    }

    private void displayResponse(List<Professor> professorList) {
        for (Professor professor : professorList) {
            professor.toString();
        }
    }

    private void createResponseForResponse(
```
ProfessorListResponse professorListResponse,
HttpServletResponse response) {
    // convert to JSON
    response.setContentType("application/json");
    PrintWriter out;
    try {
        out = response.getWriter();
        out.print(gson.toJson(professorListResponse));
    } catch (IOException e) {
        e.printStackTrace();
    }
}

private ProfessorListRequest populateRequestParametersForList(
    HttpServletRequest request) {
    BufferedReader reader = null;
    StringBuffer jsonRequest = new StringBuffer();
    try {
        reader = new BufferedReader(new InputStreamReader(request
            .getInputStream()));
        String thisLine;
        while ((thisLine = reader.readLine()) != null) {
            jsonRequest.append(thisLine);
        }
    } catch (IOException e) {
        throw new RuntimeException(e);
    }

    ProfessorListRequest professorListRequest =
        gson.fromJson(jsonRequest
            .toString(), ProfessorListRequest.class);
    return professorListRequest;
}

private List<Professor> getListofProfessor(
    ProfessorListRequest professorListRequest) throws Exception {
    List<Professor> professorList = populateProfessorListFromDatabase(
        professorListRequest.getFirstName(),
        professorListRequest
            .getLastName());
    return professorList;
}

private List<Professor> populateProfessorListFromDatabase(String
    firstName,
    String lastName) {
    List<Professor> professorList = new ArrayList<Professor>();
    Professor professor = null;
try {
    // Create new instance of Database
    db = new SQLConnection(dataTable);
    conn = db.getConnection();
    String queryString = "SELECT * FROM sdsugoggles_search_prof prof, " + "sdsugoggles_department dept " + "where prof.sdsugoggles_prof_deptid=dept.sdsugoggles_department_id AND " + "prof.sdsugoggles_prof_firstname like ? AND " + "prof.sdsugoggles_prof_lastname like ?";
    st = conn.prepareStatement(queryString);
    st.setString(1, firstName + "");
    st.setString(2, lastName + "");
    rs = st.executeQuery();

    while (rs.next()) {
        professor = new Professor();
        professor.setId(rs.getString(1));
        professor.setFirstname(rs.getString(2));
        professor.setLastname(rs.getString(3));
        professor.setDeptId(rs.getString(4));
        String imgLen = rs.getString(5);
        byte[] imageByeArray = imgLen.getBytes();
        professor.setPic(imageByeArray);
        professor.setDept(rs.getString(7));
        professorList.add(professor);
    }
}

} catch (Exception e) {
    e.printStackTrace();
} finally {
    db.closeConn(conn);
}

return professorList;

SearchProfessorCommunicator, a class that extends HttpServlet and override doGet or doPost (in my case both), depending on whether the data is being sent by GET or by POST. These methods take two arguments: an HttpServletRequest and an HttpServletResponse. The HttpServletRequest has methods that let you find out about incoming information. The HttpServletResponse has methods that lets you specify the HTTP response line, header and, most importantly, lets you obtain a PrintWriter used to send output back to the client.

In this application, I have used HTTP over JSON rest services. I have used 2 techniques first is To sent JSON request and another one is to populate request from JSON string. To implement this techniques in simple way I have used GSON. GSON is a Java
library that can be used to convert Java Objects into their JSON representation. It can also be used to convert a JSON string to an equivalent Java object.

### 3.4.1 Features of Gson

- Provide simple toJson() and fromJson() methods to convert Java objects to JSON and vice-versa
- Allow pre-existing unmodifiable objects to be converted to and from JSON
- Extensive support of Java Generics
- Allow custom representations for objects
- Support arbitrarily complex objects (with deep inheritance hierarchies and extensive use of generic types)

So using GSON it is very easy to convert request/response to/to from JSON. Client sent the request to search professor [8].

### 3.4.2 Init()

After servlet class declaration init() method will be called. Next lines of code declares the init() method. Init() method is a Servlet life cycle method and defined in javax.servlet.Servlet interface.

```java
public void init(ServletConfig config) throws ServletException { super.init(config); }
```

Init() method is used to initialize the servlet. In my application servlet, init() method doesn’t do anything, it just calls the super class version of the method.

### 3.4.3 HTTP POST Request

Next lines of code declares the doPost() method. doPost() method is defined in HttpServlet class. doPost() method is used to handle the HTTP POST request. This method takes HttpServletRequest and HttpServletResponse objects as arguments. HttpServletRequest object encapsulates the information contained in the request. HttpServletResponse object encapsulates the HTTP response.

### 3.4.4 Populate Request Statement

Next line of code calls the private method populateRequestParametersForList(req); This method takes HttpServletRequest as an argument. This method returns ProfessorListRequest object. In this method, we read the request from the client which is
nothing but a JSON request string. So I have used BufferedReader Object, which provides a more efficient way to read string. BufferedReader objects use a buffer to store input from an associated Reader. Clearly BufferedReader is more efficient than a regular Reader. All reading is done directly from the buffer. So by using defined readLine() function we read the string. This readLine() returns a String containing the contents of the line, here it is a JSON Request from Client. This JSON request is in the JSON format, servlet can’t understand JSON String. So I have used fromJSON(Request, Class) method. This method deserializes the specified Json into an object of the specified class. This method works fine if the any of the fields of the specified object are generics, just the object itself should not be a generic type. So finally we have ProfessorListRequest object.

Metadata for Professor List Request.

```java
public class ProfessorListRequest implements Serializable {

    private static final long serialVersionUID = 1L;
    private String firstName;
    private String lastName;

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    @Override
    public String toString() {
        return "ProfessorListRequest [firstName=" + firstName + ",
        lastName=" + lastName + "]";
    }
}
```

3.4.5 Retrieve Data from Database Based on Request

Next line of code populate data according to client’s request and returns back with data from database. getListofProfessor(ProfessorListRequest professorListRequest); This method take client’s request as an argument. This request has been generated before the
method call. For example professorListRequest. Before populate data from database we need to know what information we want from database. As we know here user has search for particular professor. User has sent us firstname and/or lastname of the professor. But, how to get information back? For that we have request object, so using request object and get methods or model class (professorListRequest class for our example) we can get actual value which user has entered. For our example we have professors firstname and/or lastname. Now pass this actual value to database by writing a query.

3.4.6 Create JDBC Prepare Statement

The efficient and more convenient way for sending SQL statements to the DBMS is to use a PreparedStatement object. The main feature is that, it is given an SQL statement right when it is created. Then create SQL statement is sent to DBMS right away, where it is compiled. In this way, a PreparedStatement is connected as a association with a connection and a compiled SQL statement. Advantage is that if we need to use the same, query with different parameters several times, the statement can be compiled and optimized by the DBMS just once.

PreparedStatements are also created with a Connection method. Before execute a PreparedStatement, we need to supply actual values for the parameters. This values in our example is firstname and/or lastname. This can be done by calling one of the “set” method defined in the class PreparedStatement.

String queryString = "SELECT * FROM sdsugoggles_search_prof prof, " + "sdsugoggles_department dept " + "where prof.sdsugoggles_prof_deptid=dept.sdsugoggles_department_id AND " + "prof.sdsugoggles_prof_firstname like ? AND " + "prof.sdsugoggles_prof_lastname like ?";

st = conn.prepareStatement(queryString);
st.setString(1, "%" + firstName + "%");
st.setString(2, "%" + lastName + "%");
rs = st.executeQuery();

As you see above first I have created queryString which nothing but the SQL query, then I have pass the SQL statement or query to DBMS using prepareStatement and I have used setString method to pass the parameter to DBMS with SQL query. And finally I have generated resultSet.
3.4.7 Prepare Raw Material for Response

```java
while (rs.next()) {
    professor = new Professor();
    professor.setId(rs.getString(1));
    professor.setFirstname(rs.getString(2));
    professor.setLastname(rs.getString(3));
    professor.setDeptId(rs.getString(4));
    String imgLen = rs.getString(5);
    byte[] imageByeArray = imgLen.getBytes();
    professor.setPic(imageByeArray);
    professor.setDept(rs.getString(7));
    professorList.add(professor);
}
```

So finally, we have data from database. Now, next step is to prepare response for client. Here user wants list of professor with professor information. Using “get” methods defined in ResultSet class I have set all the information in professor class and made a professorList and return that list. After doing all this stuff we need to prevent database leak so whether the operation is successful or not we need to close database so in finally block I have used `db.closeConn(conn)`; which close the conn after each and every operation.

3.4.8 Create Response

Now, we have list of professor. Next step is to generate response and reply back to client.

```java
ProfessorListResponse professorListResponse = new ProfessorListResponse();
professorListResponse.setProfessorList(professorList);
```

Here, ProfessorListResponse is another model whose metadata is as below.

```java
public class ProfessorListResponse implements Serializable {

    /**
     * @author Hardevji
     */
    private static final long serialVersionUID = 1L;

    private List<Professor> professorList;

    public ProfessorListResponse() {
        professorList = new ArrayList<Professor>();
    }

    public List<Professor> getProfessorList() {
    }
```
private void createResponseForResponse(
    ProfessorListResponse professorListResponse,
    HttpServletResponse response) {
    // convert to JSON
    response.setContentType("application/json");
    PrintWriter out;
    try {
        out = response.getWriter();
        out.print(gson.toJson(professorListResponse));
    } catch (IOException e) {
        e.printStackTrace();
    }
}

So finally we create response using above method and as we all know that we are using HTTP+JSON api. For that the content type is ("application/json"); Response can be returned to the client by writing to the java.io.PrintWriter object associated with ServletResponse. Following line obtains the PrintWriter object from HttpServletResponse object. We need to send whatever response is in the JSON format. So to convert into JSON, I have used gson.toJson() method. When I call toJson(obj), Gson calls obj.getClass() to get information on the fields to serialize. Finally server replies back to client.

3.5 WEB.XML DESCRIPTOR FIL

Web.xml file is called the deployment descriptor, it includes the configuration of our web application. The first line in the deployment descriptor specifies the version of the XML and encoding used. Next is the servlet definition. Every Servlets in a web application must be defined in the web.xml file using <servlet> tag.

<servlet>
    <servlet-name>SDSUSSearchProfessor</servlet-name>
<servlet-class>
edu.sdsu.master.thesis.sdsugoggles.communicator.SearchProfessorCommunicator</servlet-class>
</servlet>

Servlet-name tag specifies the name of the servlet, this name is used later in servlet-mapping. Servlet-class specifies the fully qualified class name of the servlet.

Next is servlet mapping. Servlet mapping is used to map the URLs to a servlet. The following servlet mapping definition specifies that the request for the URL /master/thesis/professor/list should be handled by SDSUSearchProfessor.

<servlet-mapping>
  <servlet-name>SDSUSearchProfessor</servlet-name>
  <url-pattern>/master/thesis/professor/list</url-pattern>
</servlet-mapping>

Next line ends the <web-app> tag.
</web-app>

In this way whole server implementation is done.
CHAPTER 4

CLIENT IMPLEMENTATION

4.1 REQUIREMENT

The main objective of developing SDSU Goggles application is to show rating and review of professor and subject. The requirements gathered are enlisted below.

The language used in Android OS is Java. Hence, Java SDK is also required to develop software on Android OS.

All mobile devices having Android OS installed comes with internet connection. Thus we can connect our mobile with smartphone server and get all the requested data.

4.2 HARDWARE REQUIREMENT

Any device installed with Android OS 2.1 or later version which would be able to run this application. Currently, Application screen shots shown in this chapter is run on Android OS 2.3.4 (froyo ) Model number Droid3 as shown in see Figure 4.1.

In order to run this application we require Wi-Fi connection. All android phones has built in Wi-Fi Manager that is available to connect with available Wi-Fi hot spot and thus with Internet. As show in below how we can connect to Wi-Fi browsing through Menu- >Settings->Wireless & networks ->Wi-Fi Settings (see Figure 4.2).

4.3 APPLICATION ON HOME SCREEN

To start an Application go to home screen and select application icon. Application name is SDSU Goggles. As show below an Icon and name of an Application on Home Screen of the phone.
// phone image with app icon.

4.4 SMARTPHONE APPLICATION DESIGN ARCHITECTURE

The client implementation is divided into 7 packages. These packages are

1. edu.sdsu.master.thesis.sdsugoggles.exception
2. edu.sdsu.master.thesis.sdsugoggles.model
<table>
<thead>
<tr>
<th>Legal information</th>
</tr>
</thead>
<tbody>
<tr>
<td>System version</td>
</tr>
<tr>
<td>Verizon.en.US</td>
</tr>
<tr>
<td>Model number</td>
</tr>
<tr>
<td>DROID3</td>
</tr>
<tr>
<td>Android version</td>
</tr>
<tr>
<td>2.3.4</td>
</tr>
<tr>
<td>Baseband version</td>
</tr>
<tr>
<td>Kernel version</td>
</tr>
<tr>
<td>Build number</td>
</tr>
<tr>
<td>ERI version</td>
</tr>
<tr>
<td>PRL version</td>
</tr>
</tbody>
</table>

**Figure 4.1.** Android phone summary.
Figure 4.2. Wi-Fi settings in android smartphone.
3. edu.sdsu.master.thesis.sdsugoggles.parser
4. edu.sdsu.master.thesis.sdsugoggles.services
5. edu.sdsu.master.thesis.sdsugoggles.services.impl
6. edu.sdsu.master.thesis.sdsugoggles.ui
7. edu.sdsu.master.thesis.sdsugoggles.ui.helper

4.5 DETAIL INFORMATION OF ALL THE PACKAGES

edu.sdsu.master.thesis.sdsugoggles.exception: This package contains the classes which handles the smartphone exceptions. Mainly Java provides an extensive set of in-built exceptions. There are cases in which we may need to define our own exceptions in order to handle the various application specific errors that we might encounter. In this application I have defined a class which handles Null Pointer Exception, Parsing Exception, IO Exception.

edu.sdsu.master.thesis.sdsugoggles.model: This package contains the same classes which defines for server models. The reason behind is that we need to generate request convert it into JSON format and send request to Server. Server do some calculation and replies back with response. The response is in the JSON format and using GSON we parse the response and store values in response classes available in this package. Finally, we use those classes to get data and shows the requested items to user. In this way the classes available in this package play essential role.

edu.sdsu.master.thesis.sdsugoggles.parser: This package contains all the parser classes. Server replies in JSON format. To retrieve values from the JSON string we need to parse it. For that I have used GSON. After parsing data I stored all the retrieved value in response classes.

edu.sdsu.master.thesis.sdsugoggles.services: This package has 2 main classes (a) SDSUServiceFactory (b) ClientHTTPServices. I have followed one object oriented design pattern which is known as Factory Pattern. So I have define interfaces for creating objects. SDSUServiceFactory contains all the Interface definition which give permission to subclasses to decide which class to instantiate. ClientHTTPService contains all the URL required to call server for response. It also contains generic doget() and dopost() methods. These methods takes care of HTTP communication and handle the exception. All the Interface definitions are define in this package only.
edu.sdsu.master.thesis.sdsugoggles.services.impl : This package mainly focuses on implementation of the interface, communicate with server, parse data and error handling. This package implements the interface, generate the URL, communicate with server, get response back, using parsers available in parser package generate response and get back to UI package. It handles almost all kind of exceptions and errors.

edu.sdsu.master.thesis.sdsugoggles.ui : The classes available in this packages are the entry point of each and every screen/intent/activity. The activity starts from here, generates UI for User, call services and show results to user.

.edu.sdsu.master.thesis.sdsugoggles.ui.helper : This is helper class which helps to keep track of data which will be required in more than one activity.

### 4.6 Flow of Application in Client

Flow of application in client: Followings are screenshots of running application.

#### 4.6.1 Main Screen

Main screen: This is landing page of SDSU goggles. You can see 2 option on the same page. I have used tab view to show 2 different option on the same page. (see Figure 4.3).

#### 4.6.2 Search Subject

Search Subject: As shown in Figure 4.4, you can search any subject number and get detail about that subject

#### 4.6.3 Search Result

Search Result : As shown in Figure 4.5, I have searched for a professor whose first name is James, once click on search background thread will create and calls server and get response from it. While in Figure 4.6 shows the result of my query search.

#### 4.6.4 Main Detail

Main Detail : As shown in Figure 4.7, When you click on List item you can see the detail of professor. All the comments and rating of that particular professor. In bottom you can see 2 option Rate and Review
Figure 4.3. SDSU goggles’ landing page.
Figure 4.4. Search subject screen.
4.6.5 Options

Add Review: As shown in the Figure 4.8, you can write your review anonymously, and submit it. This will redirect you professor detail page again and you can see your comments. You can see refresh page in Figure 4.9.

4.6.6 Add Review

Add Rate: Another functionality is add rate. I have searched for the cs580 (see Figure 4.10). It gives me list of subject whose number is cs580 (see Figure 4.11). I selected one of them and app redirects me in detail page same as professor detail page. Then I chose rate my class option. As shown in the Figure 4.12, you can rate your class anonymously, and submit it. This will redirect you subject detail page. and you can see your comments. You can see refresh page in Figure 4.13.
Figure 4.6. List of professor first name as Joseph.
Figure 4.7. Professor detail page.
Figure 4.8. Add review screen.

algorithms
It clear my doubts about basic algorithms. Assignment of this class is the best practice of to
nice teaching style

I audit his CS660 class. Through that class I have learnt a lot about real time algorithms. It cleared my doubts about basic algorithms. Assignment of this class is the best practice of how to implement real time algorithms.

Figure 4.9. Detail page with new comments.
Figure 4.10. Search subject.
Figure 4.11. Subject detail screen.
Figure 4.12. Rate my class screen.
Figure 4.13. Detail of class with new avg rate.
CHAPTER 5

CONCLUSION

Most of the time before selecting the class many students are looking for reviews of the subject, how easy or difficult the subject for them? Can he/she pass it? Does this professor help him/her out for their problems? Is this right professor the to ask for particular problem? All this questions are answered by SDSU star and review system. This application of SDSU professor and subject allows student as well as staff and faculty to rate and review any subject and/or professor. It helps student to select subject of interest. This system is helpful, easy, and transparent to use. Student can rate and review the professor anonymously, so they can keep their privacy. Again, smartphones are very common in day to day life. Today each and every person has smartphone. I took advantage of smartphone and made my application very helpful, handy and user friendly.
CHAPTER 6

FUTURE PLAN

The project is so flexible to add new features. In future one feature can be made like user can rate the subject/class or professor in different category and can see the average rating on the landing page. Another feature can be add like discussion of the assignment, where user can discuss about assignment questions and get answers. Add more information about subject, professor. Another, a very important feature and also highly advanced is to add voice based search, rate and review. So instead of clicking the buttons user just say search professor and result will be list of professors having the same name. Finally, we can make it global so other universities can use it too.
REFERENCES


