VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Sujan Raj Shrestha

A COLLEGE ANDROID MOBILE APPLICATION

All the College's information in one place

Information Technology
2016

ACKNOLEDGEMENTS

Firstly, I would express my sincere thanks to my Supervisor and software teacher Dr. Ghodrat Moghadampour for providing me the valuable guidelines to complete my thesis.

I am thankful for my family who have provided me unceasing encouragement, support, and attention and my special person who always makes me smile.

I want to express gratitude to all the department faculty members of VAMK for their help and support, to my Android teacher for providing me guidance and Finland for providing me the opportunity to study in this beautiful environment.

Sujan Raj Shrestha
e1100612@edu.vamk.fi
+358443474648
Vaasa, Finland

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Degree Program in Information Technology

ABSTRACT

Author	Sujan Raj Shrestha
Title	A College Android Mobile Application
Year	2016
Language	English
Pages	109
Name of Supervisor	Dr. Ghodrat Moghadampour

The world of technology is growing rapidly. Mobile devices and their applications are one of them. Android is one of the top operating systems, being one out of more than billions of devices.

The main objective of the project was to develop a native Android application which provides the user with information about every college that is affiliated with their respective universities. The application allows the user to register and to login using their registered credentials. The user can select a university and a college with their affiliation. The application features call, send email, let view the website, receive push notification message and view the location of the college. Admin can update every piece of information provided for users as well as add, update and delete using same application.

The application was developed on the Android platform using Java in Android Studio IDE. PHP was used for server side, and MySQL database was used to store the data. XML was used to designed the layout which is seen in UI.

CONTENTS

LIST OF FIGURES AND TABLES

LIST OF ABBREVIATIONS

ADT	Android Development Tool
AOT	Ahead of Time
API	Application Programming Interface
APK	Android Application Package
App	Application Program
ART	Android Runtime
AVD	Android Virtual Device
Config	Configuration
DB	Database
DBMS	Database Management System
ER	Entity Relationship
FS	Functional Specification
GCM	Google Cloud Messaging
GNU	GNU’s Not Unix
GPS	Global Positioning System
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
ID	Identity
IDE	Integrated Development Environment

iOS	iPhone Operating System
HTML	Hypertext Markup Language
JAVA EE	Java Platform Enterprise Edition
JDK	Java Development Kit
JRE	Java Runtime Environment
JSON	JavaScript Object Notation
JSP	Java Server Pages
JVM	Java Virtual Machine
MS	Microsoft
OHA	Open Handset Alliance
OOP	Object Oriented Programming
OpenGL	Open Graphics Library
OS	Operating System
PC	Personal Computer
Pdf	Portable Document Format
PHP	Hypertext preprocessor
QFD	Quality Function Deployment
RDBMS	Relational Database Management System
SDK	Software Development Kit
SE	Software Engineering

SGML	Standard Generalized Markup Language
SQL	Structured Query Language
SSL	Secure Socket Layer
UI	User Interface
UML	Unified Modeling Language
URL	Uniform Resource Locator
VAMK	Vaasan Ammattikorkeakoulu
VM	Virtual Machine
W3C	World Wide Web Consortium
WIFI	Wireless Fidelity
WWW	World Wide Web
XML	Extensible Markup Language

1 INTRODUCTION

The main target of the project is to develop an Android mobile application which provides different functions for users to get information of various colleges affiliated with their respective university.

1.1 Background

Development of mobile phones is fast with in the development of technologies. A mobile phone is easy to use, portable, handles big data, multifunctional and user-friendly. So, many operating systems are being developed for those devices like Android operating system, iOS and Windows Phone operating system.

Android is a Linux-based OS developed by Open Handset Alliance and is currently developed by Google. The OS is designed for smartphones and tablet computers. Today it is also being developed for car's dashboards, televisions, smart watches, refrigerators that use their interfaces to run their application.

1.2 Motivations

The application focuses on the context of Nepal where different colleges are affiliated to different universities. It is hard to find the right college with specific faculties with the right affiliation in a very short period. So, the development of this application may provide that information in one place without the need to visit the colleges. I started this application in 2015 when I was in Nepal studying Android Application Development but due to my classes the development of the application could not continue, and now I am developing it on.

1.3 Objectives

The primary objective of the application is to provide users with information, phone numbers, official websites, locations and notifications of the college affiliated to respective university. The administration has the rights to manage and to update all the information and descriptions of the universities and colleges.

All data is saved inside MySQL database which is sent and received by implementation of PHP in server side, using internet android application communicates with server side to fetch data in the interface.

1.4 Description of the Topic

One College Android mobile is an application which is made in Android platform to provide the user with most of the information of a college affiliated to their respective university in one place.

2 RELEVANT TECHNOLOGIES

This section includes a detailed description of the technologies that are used during the development process, the structure of the application and its development cycle.

To develop an application many different technologies are needed. It is hard to build an application without the components that are relevant. The technologies that are used to build up the application used are Android, PHP, MySQL, JSON, and XML.

2.1 Android

Android is an open source Linux-based OS made for mobile devices (smartphones and tablet computers), developed by Open Handset Alliance led by Google. Android provides various open source libraries, framework, SDKs and also plug-ins and documentation. Android has many features like a beautiful interface, connectivity, media support, multi-touch, multi-language, and much more. /1/

Table 1: Android versions. /1/

Version	Code Name	API
2.2	Froyo	8
2.3.3 - 2.3.7	Gingerbread	10
4.0.3 - 4.0.4	Ice Cream Sandwich	15
4.1.x	Jelly Bean	16
4.2.x		17
4.3		18

Versions	Code Name	API
4.4	Kitkat	19
5.0	Lollypop	21
5.1		22
6.0	Marshmallow	23

2.1.1 Android OS Architecture Overview

The OS consists of different layers of software divided into five sections that are divided into four main sections. Below is the architecture diagram of Android OS:

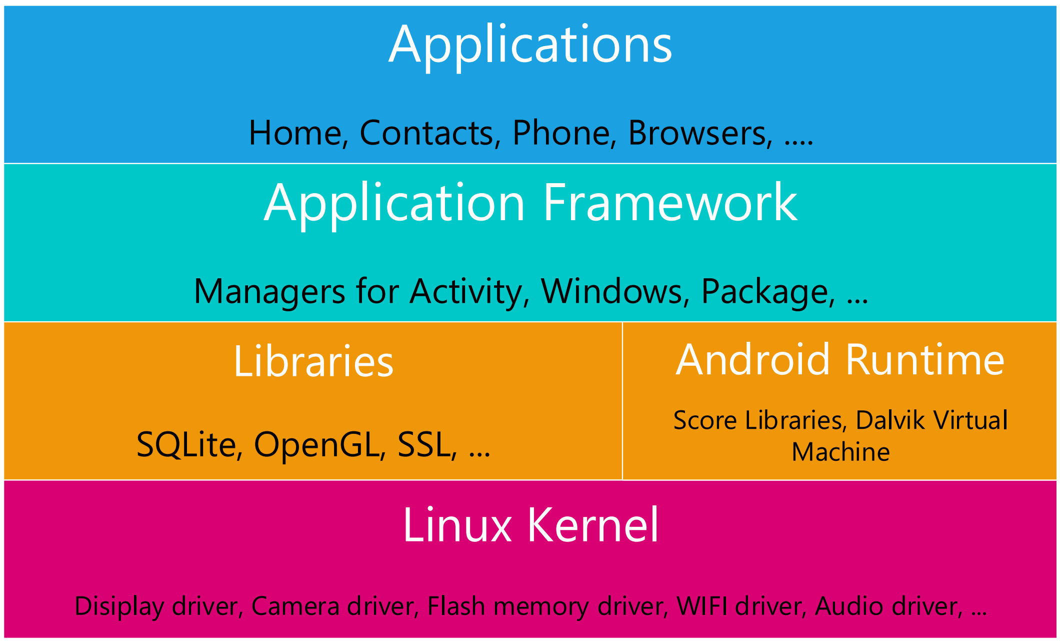

Applications
Home, Contacts, Phone, Browsers, ….
Application Framework
Managers for Activity, Windows, Package, …
Libraries
SQLite, OpenGL, SSL, …
Android Runtime
Score Libraries, Dalvik Virtual Machine
Linux Kernel
Disiplay driver, Camera driver, Flash memory driver, WIFI driver, Audio driver, …

Figure 1: Architecture Diagram of Android Operating System. /2/

- The bottom layer is Linux kernel, which handles device drivers, memory, power, network and security. /2/

- The next layer is Android's native library that handles application data. Different functionalities are defined inside this Android library. Surface Manager manages off-spring buffering, Media framework provides different media codecs and formats, SQLite is the database engine, WebKit is the browser engine that displays HTML content whereas OpenGL renders graphics.
  Dalvik Virtual machine and Core Java libraries are inside Android Runtime. This part of the layer forms the basics of the application framework. To run the multiple instances efficiently in the device Dalvik is needed, which is registered based virtual machine. A new VM called ART was released by Google. It is replacing Dalvik VM. The advantages of ART are AOT over Dalvik VM and garbage collection improvement. This advantage boosts the performance of the application. /2/
- Application Framework layer provides all the classes and Java libraries that used to develop an application. There are several blocks inside this layer. An application manager regulates the activity process of the application. The content providers regulate the information sharing between applications. Telephony Manager handles all voice calls. Location Manager does the management of location, use of GPS or the antenna. Resource Manager operates different resources used in the application. /2/
- Application Layer consists of all the available applications. The layer provides developers to develop an application to the OS. Some common Android pre-installed applications in every device are: Messaging app, Web Browser, Contact Book and Calling Application. /2/

2.1.2 Android Application Components

Android application components are the blocks of different elements which are merged to build an application. The components are already defined in SDKs as objects and the application can run provided by different methods. To use the application, the developer only need to enhance these classes. Some of the main components are given below.

2.1.2.1 Activity

Activity is a visual screen of a device, called as individual user interface screen. A user can interact with the activity in UI so the whole screen makes an activity. Activity contains different views or widgets. Java codes are used to make an activity and XML to create UI.

An application consists of more than one activity which can be allied to each other. A manifest file consists of each activity defined. The Figure 2 below demonestrate the activity. /3/

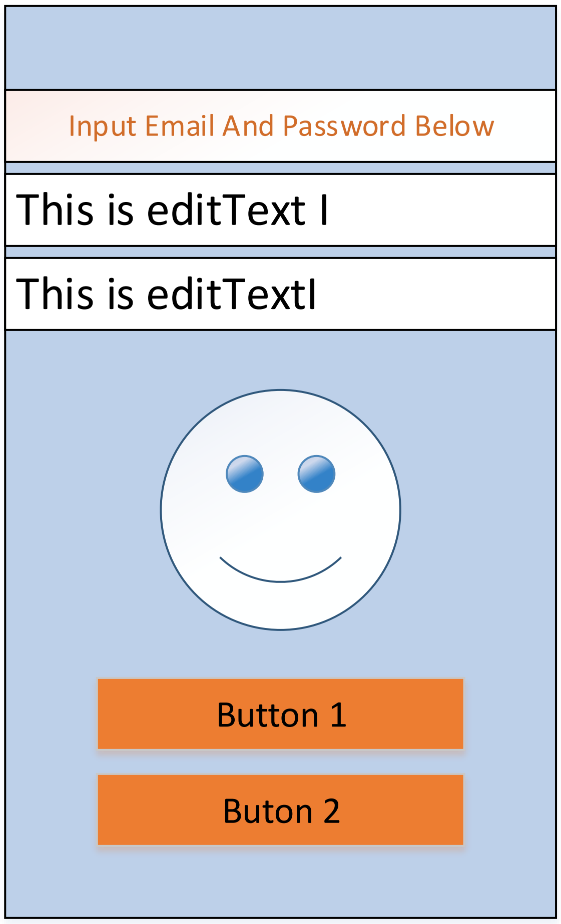


Figure 2: Activity User Interface. /3/

In this Figure 2 there is a text view where there is some written text, edit text can let user write some text, a picture where the user can see it and a button where the user can tap it to work as its functions like a sign in or opens some other activity.

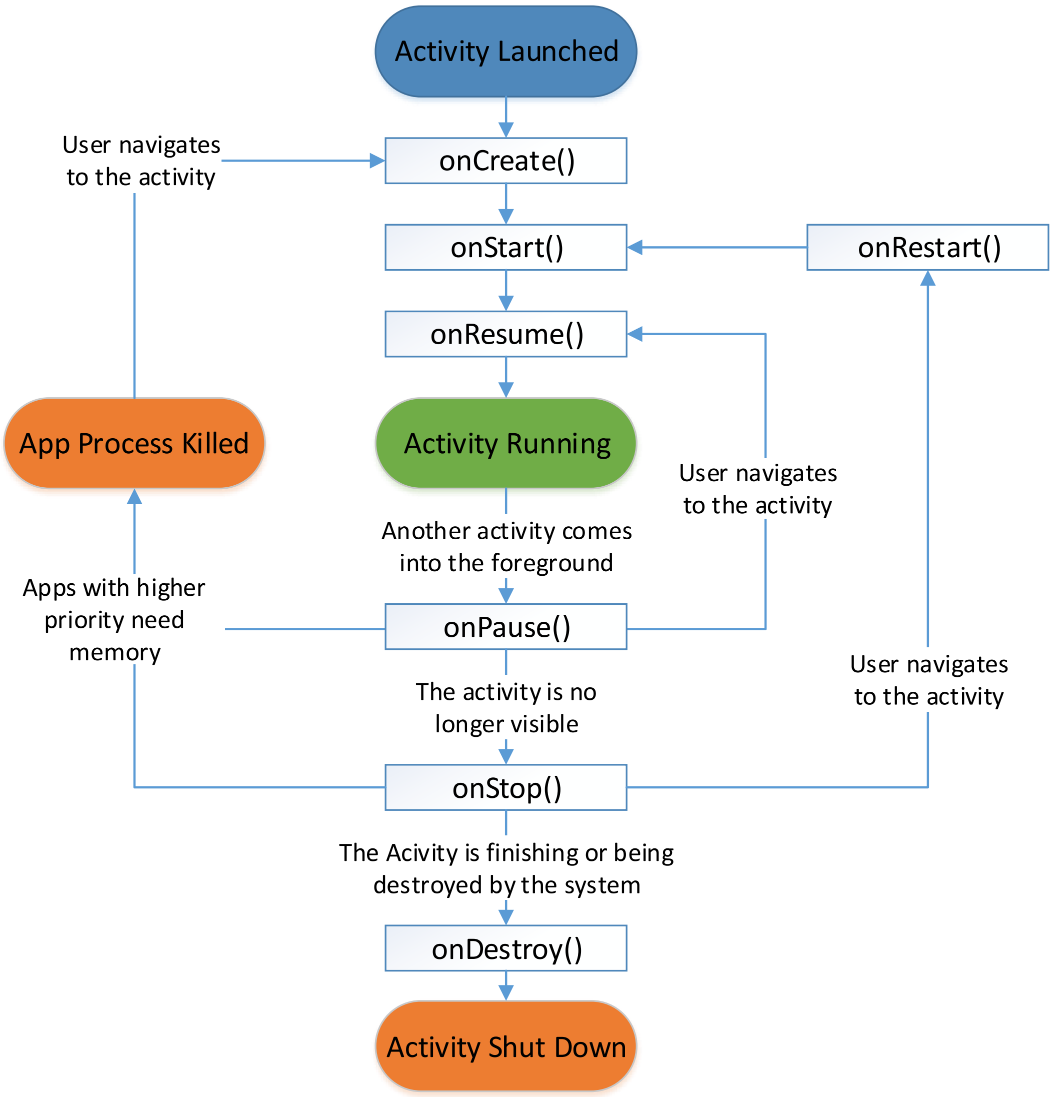

Activity Launched
User navigates to the activity
onCreate()
onStart()
onRestart()
onResume()
App Process Killed
Activity Running
User navigates to the activity
Another activity comes into the foreground
Apps with higher priority need memory
onPause()
User navigates to the activity
The activity is no longer visible
onStop()
The Acivity is finishing or being destroyed by the system
onDestroy()
Activity Shut Down

Figure 3: Activity Flow Diagram. /4/

Below are the descriptions of an Activity diagram: /4/

- An application in Android OS starts from an Activity whereas other programs use main() while starting.
- When first activity created, onCreate() is a first callback.
- When activity is visible in the UI, onStart() callback is called.
- When the user is interacting with an application, onResume() call back is called.

- When the current activity paused, and previous activity is resumed, on-Pause() callback is called. This activity is not in use that does not respond anything.
- When an activity is no longer visible, onStop() callback is called.
- Before an activity is destroyed by a system, onDestroy() callback is called.
- When the activity restarts, onRestart() callback is called.

2.1.2.2 Fragments

A fragment is a part of activity. An activity consists of different fragments. Multi-pane UI and multiple fragments can build a single activity. Fragments can also be reused in multiple activities.

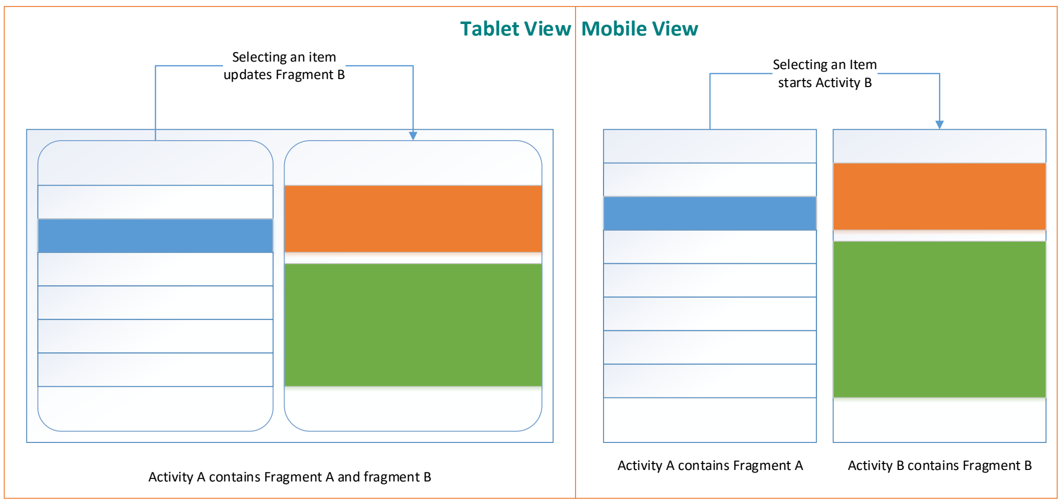

Tablet View
Mobile View
Selecting an item updates Fragment B
Selecting an Item starts Activity B
Activity A contains Fragment A and fragment B
Activity A contains Fragment A
Activity B contains Fragment B

Figure 4: Fragments. /5/

2.1.2.3 Services

Services is a component that runs behind the UI of an application. While the user is working on UI in the foreground, services perform processing in the background. Even after user switches to another application the service of a minimized application will continue in the background. For example, when playing music the user can open another application. Even if the first activity gets destroyed, the music app continuous to run in the background. That shows that the services control the activity.

Downloading a file from the internet downloads in the background even if a user is using any other application at the same time. /3/

There are two types of services.

- Unbound services are not bounds to any components that still continuously run in the background even if it kills or it will stop itself when its task is complete.
- Bound services are bounded to the components and runs till the respective component runs. /3/

2.1.2.4 Content Providers

Content providers is a standard interface that manages and accesses the structure of data. Content provider is an inbuilt component in Android OS that defines inside Android SDK. The main function of a content provider is to process data all over the application inside the OS as long as the content provider allows it. The user can access the data that is created in one application to another application. One can query, access or even update the data generated if the content provider allows that. For example, the database of contacts can allow any other application to view, read and update its contents. /3/

2.1.2.5 Broadcast Receivers

Receiving any messages or notification is seen in the UI are broadcasted by Android OS or any kind of application inside the device. Android broadcast receivers do these functions. Some examples of broadcast receivers are turning off the display, changing time zone, low battery notification, charging battery logo, etc. /3/

2.1.2.6 Intents

Intents are an asynchronous message which allow users to interact with different activities in the same application as well with other applications. For example, an application needs to take a picture. That application uses intents to use the activity of a camera to take a picture being in the same application. There are two types of intents given below: /3/

- **Explicit Intents:** Explicit intents connects the activity in between application. For example, by clicking a button, you can have another UI that is another activity inside an application. /3/
- **Implicit Intents:** Implicit Intents connects the activity in between an application to another application. For example, if a user wants to view a contact information, implicit intents request another application (suppose: contact application) to view the contact information. /3/

2.1.2.7 Process and Threads

The process defined as a unit execution of a program that runs on its allocated memory and resources. Running a program usually contains many processes (main and child), and they are independent of each other. The program hosted in an OS does communication via Inter Process Communication. For example, Google Chrome runs different tabs in a browser, where every tab is a process and independent of others. One tab can run even if any other tab crashes. /6/

The thread runs parallel with main application execution process that is a part of its program functionality. The process contains many threads that run upon the same memory and resources. Creating any new thread inside a program is lighter than creating a new process. Threads have same allocated resources. They are not independent like a processes. For example, while downloading multiple files in a program, even if one of the downloading files stop or crashes, other files are not affected. /6/

When an Android application starts the main thread or UI Thread is executed by a new process. That process holds Activity, Service or Broadcast receiver. This thread is responsible for widgets in UI. This process runs all the components and the same application unless some other function provides to that application. /7/

For example in a Single Thread Mode, a service is started by startService() method can check the main application process and thread from an activity. However, this can cause poor performance and can create an unresponsive application. /7/

Another thread is Worker Thread(s) that runs in the background parallel with the main thread. This thread allows the main thread to work for UI to make a responsive Android application. The worker thread can be done in different ways, by extending thread class, using Intent Service instead of applying Service class, using Async-Task, applying view.post method, etc. /7/

2.1.2.8 UI Layout Designs (android official)

Android OS has a responsive UI that runs on various small devices like a wrist watch or a smartphone to the big screens like a tablet computer or a television. Multiple devices run on such UI. UI also can be created for both landscape and portrait layouts.

An activity component is a single UI of an application. There can be more than one activity in an application so many different UI can do various activities. An activity contains different Views and View Groups. Activity has a View that is arranged by View Groups. /8/

Figure 5 illustrates View and View Groups:

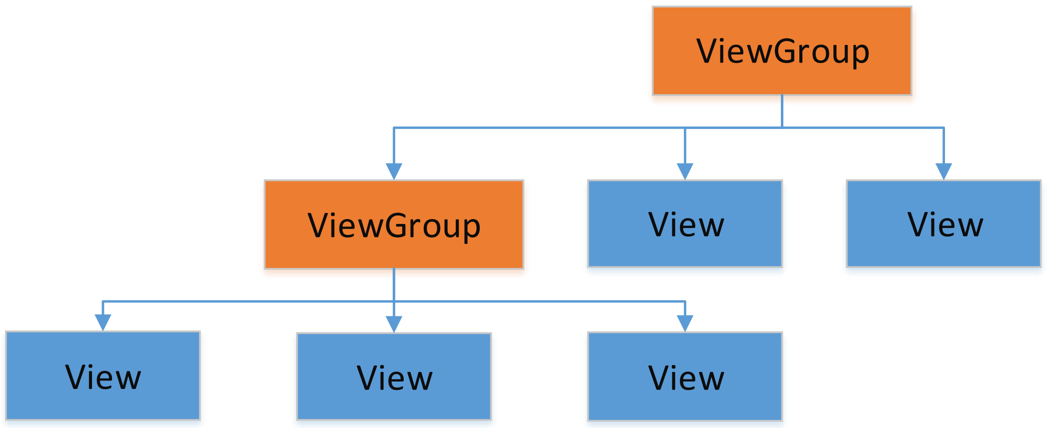


Figure 5: View and View Groups. /8/

View Groups are divided into different natures: /9/

- Linear layout: Arranges View vertically or horizontally
- Relative layout: Arranges View relative to each other
- Table layout: Arranges Views in grids

- Frame layout: Arranges Views in single view
- Absolute layout: Arranges Views according to X and Y coordinates

2.2 Java

Android uses Java to build native apps. Java utilizes Android SDK libraries to develop an application. James Gosling designed the first Java in 1991. Sun Microsystems developed it in 1995 and now it has been acquired by Oracle Corporation. Java is an object-oriented high-level programming language. Below are some of the features of Java: /10/

- It is statically-typed, platform independent.
- The Concept of OOP makes it easy to learn and has high performance.
- Because of authentication techniques based on public-key encryption, it is a secure programming language.
- Java is multithreaded, portable, distributed, interpreted and dynamic.

Java is divided into many different data types, some of the basic data types are given below:

- **Classes and Objects:** Objects are known as the component that has some kind of behaviors or states. The class is the main component or the blueprint that describes behaviors which support the object type. /11/ For example, an object can be a color, name or behavior like silent, loud of a car. The class can have properties of a car like common things. It has gears, tires, doors, etc.
- **Methods:** Method describes the behavior. It is a component type where logics are written, and execution is done by manipulating the data. There can be many methods in a class. Methods are reusable. /12/
- **Packages:** Package is a group of different classes. It is a collection of classes that are related. /11/

- **Access Modifiers:** The keywords that control any class, variable or method are access modifiers that are public, private and a protected. That can define how classes, methods and member variables are accessed. Below Table 2 can shows how access modifiers are used and accessed. /11/

Table 2: Java Access Modifiers. /11/

Access Modifiers	**Same Class**	**Same Package**	**Sub Class**	**Other Packages**
Public	Yes	Yes	Yes	Yes
Protected	Yes	Yes	Yes	No
Private	Yes	No	No	No
No Accessed Modifiers	Yes	Yes	No	No

- **Conditionals:** The process of checking a condition of any data that meets its requirement or functions or not is conditionals in programming. If condition can check the values using Boolean values true or false. /11/
- **Arrays:** Array is a collection of the same type of variables. Variables can be declared using numbers like [0], [1], [2] and so on. Data can be accessed in such a way as they are structured. /12/
- **Loops:** If there are more than hundreds of data to look at the application, is not possible to write every code that can analyze every data unit by unit. So, a loop can work in such a way that its limited codes can check hundreds of data to be analyzed. Different kinds of loops can be used to check data. For-Loop, at first initialize a statement, which can be terminated by another syntax and a second step statement can define how that works. For example, number zero can initialize a statement that can be defined by action element as ++ to go from 1,2 and so on and if the terminate statement is 5 it analyzes

data till it reaches 5 from 0. Other loops types are, the do-while loop and for-each loop. /11/

- **Inheritance:** In Java inheritance means the process where a class can acquire the properties of another class. It is also known as sub or child class, and the class which properties are inherited are known as super, base or parent class. /14/
- **Overriding:** The functionality of an existing method can be overridden in OOP that is overriding. A particular implementation of the method that already in the parent class can be allowed to a child class by the use of override feature. /15/
- **Polymorphism:** Ability of an object to take different forms is a polymorphism. A parent class refers a child class by the use of polymorphism. (tutorials point) For example, a parent class shape can be enabled to be used in different methods like in a circle, rectangle, triangle, etc. /16/
- **Abstraction:** The process of providing the main functionality to the user by hiding of the other data that implemented is abstraction. For example, in the real world a user can only see the filling boxes when sending email but not see how that email works. /17/
- **Encapsulation**: Encapsulation is the hiding data of the variables of a class from other classes. Those data can only be accessed through the methods of their current classes. The advantages of encapsulation are: the class field can make a read or a write only, the class and the user can control the stored data will not get to know if the class stores the data or not. /18/

Java is used to develop many other projects too. The built-in libraries can develop JSP, applications with GUI, web servlets (Servlet API) and other community libraries. It uses JVM a universal platform to run every Java based applications.

2.3 PHP

PHP is a scripting language that is used widely and it is executed on the server. It is open, free to use and download from the official PHP resource: (http://www.php.net).

It has extension file named ".php". On various computer types and operating systems PHP can run, like Mac OS X, Windows PCs, and Linux. /19/ Some of the functions of the PHP are given below: /20/

- Dynamic page content can be created using PHP.
- PHP supports a broad range of databases. For example, Oracle, MySQL, and PostgreSQL.
- To create, to open, to read, to write, to delete, and to close files on the server.
- Form data can be collected using PHP.
- Sending and receiving cookies.
- The functions such as to add, to delete, to update data in the database can be easily done using PHP.
- PHP can control user access.
- The encryption of information in the database.

2.4 MySQL Database

The collection of data or any information organized inside a system is known as a database. Organization of data can be accessed easily, managed properly and can be modified or updated as per the system's or application's needs. /21/. Many systems need to store tons of information in its system which is managed or stored by the database system called as DBMS.

The database can be accessed and manipulated using the standard language called SQL. The functions of SQL are to execute a query against the database, retrieve data, insert, update and delete records. The query also can create a new database and tables. The permission also can be set in a table. RDMS is one of SQL versions.

MS Access, SQL Server, and MySQL are types of RDBMS. It uses a server-side scripting language like PHP or ASP. /22/

MySQL is developed, distributed and supported by Oracle Corporation. It is the relational database that stores data in separate tables. It is open source, fast, reliable, scalable, and convenience to use. /23/

2.5 JSON

JSON is open text-based light-weight scripting software whose data is interchangeable for human reading. Its file extension is .json. JSON uses in various programming languages like PHP, PERL, Python, Ruby, Java, etc.

Some of the JSON elements and syntax rules are: data is in the key/value pairs that make JSON Object. Commas separate it, and key values are in different forms like integer, double or string, etc. Curly braces hold the object {} and the square brackets holds the arrays [ ]. The example syntax of JSON is: /24/

```
"students":[
        {"firstName":"Sujan","lastName":"Shrestha"},
        {"firstName":"John","lastName":"Cena"},
]
```

JSON data can be manipulated using four different classes: JSONArray, JSONObject, JSONStinger and JSONTokenizer. /25/

- **JSON Parsing:** JSON consists of different objects whose key/values are parsed. To parse it, JSON has a separate function for each component. Below are some of the methods described: /26/

Table 3: JSON Methods. /26/

Methods	**Descriptions**
names()	Returns an array containing the string names in this object
length()	Returns the number of name/values mappings in the object
toString()	Returns the encoded JSON string
get(String name)	Returns the value just in the form of object type
getBoolean(String name)	Returns the double value specified by the key

2.6 XML

XML is derived from SGML, which is text-based markup language developed by W3C available as open standard. XML is used in various software development processes which can be configured to implement the views of web pages, in transferring data and also in Android. Creating a tag can be self-descriptive which can suit the application. It is also used to store the data that can be irrespective of its presentation. /27/

There are two different objects in XML: markups and contents. Markups have tag which is start tag, < and end tag, /> (<start tag> and </end tag>) or empty tags (<empty tag/>). Start and end tag have content and also have attributes and an empty tag having out content are elements.

In Android, XML is used to create layouts for user interface and in the manifest. In the manifest of application, XML codes state the permission that is allowed or not, the version and all of the activities is made during the project or that are used in the application. Android OS interpret the components of XML views from the activities and make it viewable to the UI. Other static variables are also inserted in the form of XML for GUI such as strings, width, height, colors.

The main advantage of the use of XML in UI that it controls its presentation separately that controls its behaviors. UI layouts are external, so it's easy to work precisely and properly and to modify without any troubles. Every layout contains one primary or root element and child elements are defined inside it. Below is an XML layout sample: /28/

```
<Root element>
<Child element android:id="@+id/"
                    android:text="this is example"/>
<Another Child element/>
</Root element>
```

2.7 Push Notification

Push notification is a kind of messaging service that is sent to the user who has an application in the device. Push notification can be sent or can be received by the user in application UI even the application is not in use. Application icon and a message can be seen in the status bar in the user's application and device when it is received. It can be sent to every user who has application. Broadcasting Push notification has many reasons like giving some information when making some changes in the application, for the marketing campaign, etc. By using GCM, the Parse library provides push notifications services. /29/

2.8 Google Maps API

The customization and information of maps can be done using Google Maps API based on Google Maps. Downloading data, displaying maps and gesture map response are automatically handled and accessed to Google Maps server by the API. It can be used to add markers, changing user's view, overlays, and polygon in maps and allows users to different graphics like anchoring marker to the specified location, polylines, segment polygons, ground overlays and tile overlays. /30/

Implementation, several classes inside Android application project, can use Google Maps API. Google provide some key to access the maps from Google server. This application is implemented with Google Maps API version 2.

2.9 Application Structure

The application is divided into two main different parts, the client side which is user's interface where data is visualized or seen and the server side where data is written and is fetched to the client side.

MySQL database is used to store data. The Android project is not able to communicate with MySQL database directly, so PHP scripts are used to communicate, to edit and to execute and interactive interpreters to an Android device.

Calling PHP scripts from Android application that connects MySQL database to execute an operation. This way MySQL database store data from an application. The diagram and the steps below can show how it works: /31/

- Client requests HTTP POST to serve
- MySQL server is connected using PHP scripts
- SQL sends data to PHP
- PHP scripts write the data by assigning the keys for the values in JSON array
- Finally, JSON data is parsed by an application.

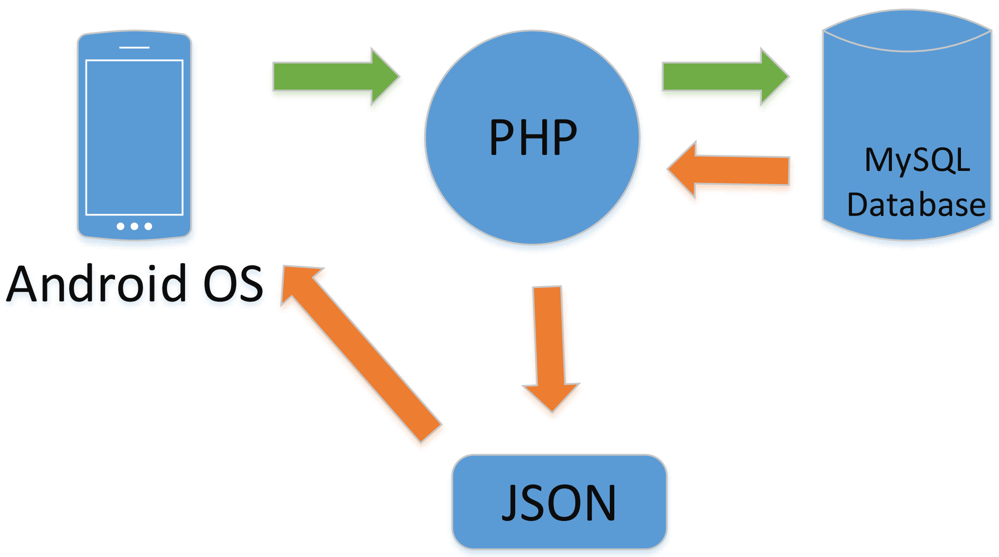


Figure 6: Application Structure. /31/

2.10 Application Development Environments

To get every work done, the right environment is needed. So to get the objective of the project, right development environments are needed. Android application development needs minimum Java JDK 5, and JRE 6 is required. To make this successful project installation of the following tools are needed:

2.10.1 Hardware

- A computer with OS Windows 7 or above version is needed or
- Mac OS X 10.8.5 or later version with Intel chip or
- Linux that includes GNU C Library 2.7 or above

2.10.2 Android Studio

The official IDE for Android application development is Android Studio. Android Studio can be downloaded from its official website (https://developer.android.com/studio/index.html) according to the OS of the computer, and can be installed according to the installation wizard. Android SDK also comes with the download package with Android Studio.

2.10.3 Xampp

Xampp is a cross-platform webserver solution stack package developed by Apache Friends. It is a free and open source. It is used to create the local web server for deployment and testing of programs that developers made. It works in Windows OS, Linux, and Mac OS. It supports application server Apache, database MariaDB, phpMyAdmin, etc. So, Xampp can be used as the local web server for the android application while developing an application. Further server and database can be changed to the online server. Xampp can be downloaded from its official website (https://www.apachefriends.org/download.html) according to the OS on the computer and can be installed according to the installation wizard.

2.11 Application Development Process

In Android Studio, there are different development process to develop an application. It is divided into four main development phases like setup, development, debugging and testing, and publishing.

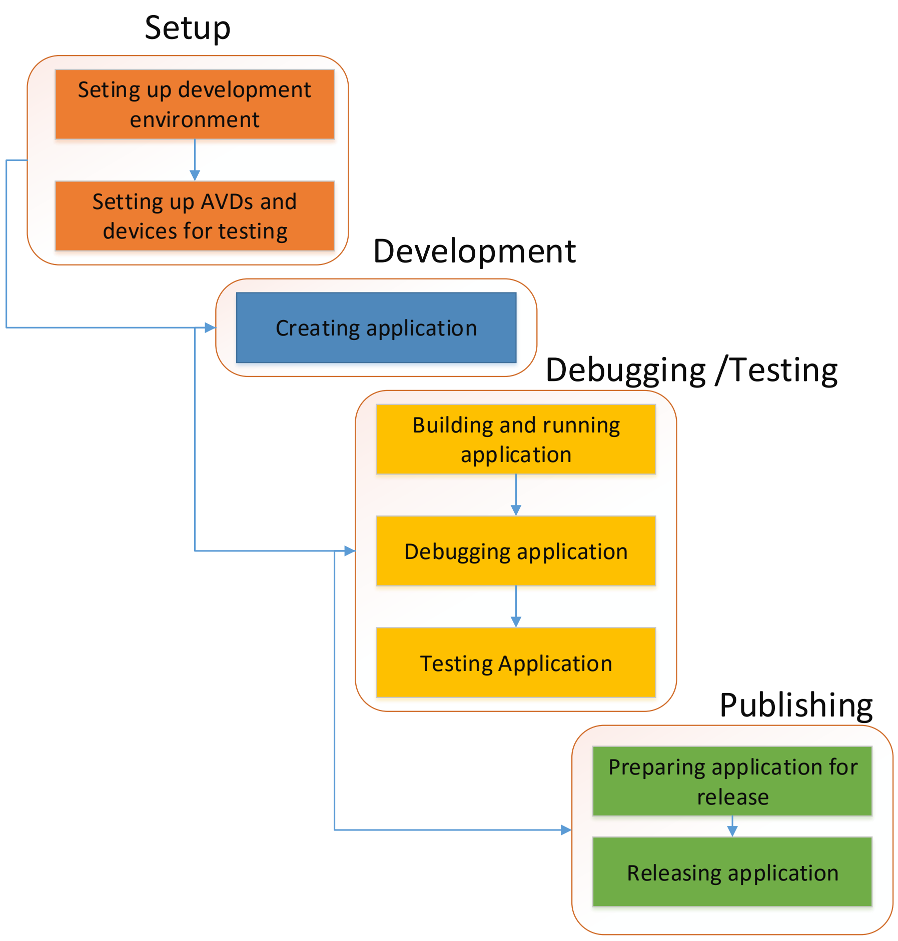

Setup
Seting up development environment
Setting up AVDs and devices for testing
Development
Creating application
Debugging /Testing
Building and running application
Debugging application
Testing Application
Publishing
Preparing application for release
Releasing application

Figure 7: Application Development Process. /32/

- **Setup Phase:** In this phase required android SDK, ADT and Android platforms are made. ADV are created and are connected to hardware devices so that it can be used to test application during the development phase.
- **Development Phase:** Android project is created using the source code, resources, and Android manifest file.
- **Debugging and Testing Phase:** In this phase, at first application is built and run in debugging mode. After that, the application is debugged using Android debugging and logging tools. Then the application is tested using Android testing and instrumentation framework.
- **Publishing Phase:** In this phase the application is configured, build and tested in release mode. Finally, the application is publicized, sold and distributed to the users.

3 APPLICATION DESCRIPTION

In this section, the general descriptions and overview of the whole systems are defined. An application will be explained to show how it works and interacts with another system, user, and its functionality. It also provides the description that who will use the application and how the feature works according to the user.

The main concept of the project is to provide every college's information with their affiliated university. The Internet is needed to run the application in the device. On the context of Nepal, the application based where there are many colleges and that are affiliated with other many different universities.

The application allows the user to register and login via their email and password. The user can view the list of universities and the colleges that affiliated to respective university. The user can view college list and inside college, user can view, college's phone number, website, and college's descriptions. The user can visit a website using a button from an application via any web browser that is inside the device, call to the college's phone number using call button, email to the college using email button and view college location using view map button. The user is also able to see push notification sent by an administrator. The user can log out using logout button.

The application allows the administrator to login via their login credentials. The administrator can view, to add, to updated and to delete university. The administrator can also view, to add, to update and to delete college and college descriptions (college's name, college's address, college's phone number, college's email, college's website URL and college's location using latitude and longitude). The administrator also can be able to send the push notification to the user who have installed the application. The administrator can log out using logout button.

Requirement analysis means determining the required needs and components that are necessary to complete a project. All the requirements including analyzing, documenting, validating and managing the project are inside this section. The section consists of various other analysis given below:

3.1.1 Quality Function Deployment (QFD)

Quality Function Deployment (QFD) helps to organize the activities and development process. QFD describes the understanding of requirement which is valuable for the project and then processed to the deployment. Use of QFD helps to save lots of time and to arrange infrastructure according to the project's need. Relevant requirements and topics are in more priority.

Table 4 describes three primary identifiers of QFD:

Table 4: Quality Function Deployment.

Normal Requirements (Must have priority 1)
• User must be able to register using their email id and password • User must be able to login using their registered email id and password • User must be able to view and select list of universities • Application should allow user to view and select list of colleges • User must be able to view college information (college's phone number, website, and college's descriptions) • User must be able to visit a website using a button from an application via any web browser that is inside the device • User must be able to call the college's phone number using call button dialer • User must be able to send email to the college using email button • User must be able to view college location using view map button • User must be able to view push notification received • Administrator must be able to login using their credentials provided • Administrator must be able to view, to add, to view, to updated and to delete University • Administrator must be able to view, to add, to update and to delete college and college descriptions (college's name, college's address, college's phone number, college's email, college's website URL and college's location using latitude and longitude)

• Administrator must be able to send push notification to the user who have installed the application • Administrator must be able to call, visit website, view location and to send email to the College email.
Expected Requirements (Should have priority 2)
• The application will be installed and run on all the device having Android OS minimum version 4.0 (Ice Cream Sandwich) to max version 6.0 (Marshmallow) • The application should be easy to use and user friendly • User will be able to use the functionalities easily
Exciting Requirements (Nice to have priority 3)
• The application may have search functions using search tab • The application may allow user to validate their email • The application may allow user to fill in their other details like name, address and phone number • The application may have other different setting features like forgot password, delete account • The application may allow administrator to upload pdf format prospectus of college to the application • The application may allow user to view and download college's prospectus in pdf format

3.1.2 Functional Specification (FS)

The specification table in which the application's functions defined is Functional Specification (FS). It serves as the outline of the role of the complete application.

FS lists user task description, comparison, an external interface and compatible software, hardware, versions and OS. This application needs an internet connection to do every activity or functions provided.

From requirement analysis, the following FS table is drawn:

Table 5: Functional Specification.

#	Case	Precondition	Input or Action	Description	Expected result	Exception
1	Login administrator Activity	Application should started Internet connection	Administrator's credentials	Check administrator's credentials	Administrator should be able to login	Incorrect credentials Internet disconnected
2	Add University Activity	Administrator should be logged in Internet connection	Administrator should input University name	University name will be written to database	University name will be saved to database	Incorrect input Internet disconnected
3	Update university Activity	university name should be listed in an Activity Internet connection	Administrator should tap and hold in the name of the university and select update option New name should be given	University name will be updated or renamed in the database	Old university name will be replaced by new university name	Incorrect input Internet disconnected

#	Case	Precondition	Input or Action	Description	Expected result	Exception
4	Delete University Activity	University name should be listed in an Activity Internet connection	Administrator should tap and hold in the university name and select delete option	University will be deleted in the database	University will be deleted	Internet disconnected
5	Add College Activity	University list should be shown in an Activity Internet connection	Administrator should tap add college button and fill in the required fields (existing university name, college name, college address, college phone number, college email, college URL, college latitude and longitude, college's description)	College name and all the details will be written to its respective University name inside database	College will be added to its respective university	Internet disconnected Invalid input
6	Update College Activity	There should be college added previously Internet connection	Administrator should tap and hold in the college name and select update New name and all the details should be given	College name will be updated or renamed in the database	Old college name will be replaced by new college name	Invalid input Internet disconnected

#	Case	Precondition	Input or Action	Description	Expected result	Exception
7	Delete College Activity	There should be college added previously	Administrator should tap and hold in the college's name and select delete option	College will be deleted in the database	College will be deleted	Invalid input Internet disconnected
8	Send Push Notification Activity	Internet collection	Administrator should tap send notification button and write message to be sent and tap send	Notification message will be sent online	Notification will be sent to the user who have application installed	Server busy Internet disconnected
9	Register User Activity	Application should started Internet connection	User credentials (email and password)	Get user's email and password to write in database	User should be able to register	Email address is already in use. Internet disconnected

#	Case	Precondition	Input or Action	Description	Expected result	Exception
10	Login User Activity	User's credentials should be registered Internet connection	User's registered credentials	Fetch user's registered email and password from database	User should be able to login	User account not found. User account does not exist. Incorrect email or password Internet disconnected
11	University list Activity	User should log in and administrator should have Added Universities Internet connection	User should be logged in	Fetch University list form database	University list Activity list is viewed in UI	University list may not load if internet get disconnected
12	College list Activity	College should be added inside its affiliated university name Internet connection	User should tap in one of the university list given	Fetch College list from database	College list Activity is viewed in UI	College list may not load if internet get disconnected

#	Case	Precondition	Input or Action	Description	Expected result	Exception
13	Visit website	Application should fetch College's website URL Device should have any kind of web browser Internet connection	User should tap to the visit website button and select any kind of browser installed in the device	Fetch website URL from database	Website URL will load	Website will not load if internet is disconnected
14	Call to College's phone number	Application should fetch College's phone number from database Device should have cellular function Internet connection	User should tap to the call button and select dialer application installed in the device	Fetch phone number from database and dialing application is viewed	Call will ring to the phone number provided	No cellular network may end calling Internet disconnected

#	Case	Precondition	Input or Action	Description	Expected result	Exception
15	Send email	Application should fetch email address Internet connection User should have any kind of mailing application installed	User should tap send email button and select one of the mailing application installed in the device	Fetch email address from database and mailing address application is viewed	Email will be sent to the respective College's email address after user	Email won't be send if internet is disconnected
16	View Location	Application should fetch location (latitude and longitude) address Device should have GPS Internet connection	User should tap view location button	Fetch location (latitude and longitude) from the database	Location can be viewed in the map User can zoom in and out in the map	Maps can't be viewed if interned is disconnected.

#	Case	Precondition	Input or Action	Description	Expected result	Exception
17	View Push notification	Administrator should send push notification Application should be installed in the device Internet connection	Push notification can be received automatically only if the application is installed on the device	Receive notification message sent by administrator	User can view notification message	Notification cannot be received if internet is disconnected

3.1.3 Use Case Diagram

The simplest representation of user’s interaction with the system is Use Case Diagram. It shows different cases in which the user is involved and the relationship between the user. In this diagram the sequence of actions that implements the value to an actor is described.

A person, a user, an organization or any external system that has one or more roles is an actor. Actor interacts with one or more use case inside the system, it is represented as the symbol of a person or a stick person. Associations are solid lines that are indicated between the actor and the use cases.

The project consists of a user and an administrator as an actor who can interact with the system. The administrator has all the rights to read, to write, to update and to delete data. The user only has the rights to read and use the data from the application activities. MySQL database stores all the data that are seen in UI.

In Figure 8 the descriptions and the diagrams are explained:

3.1.3.1 User Use Case Diagram

The following Figure 8 describes the Use Case diagram of t user.

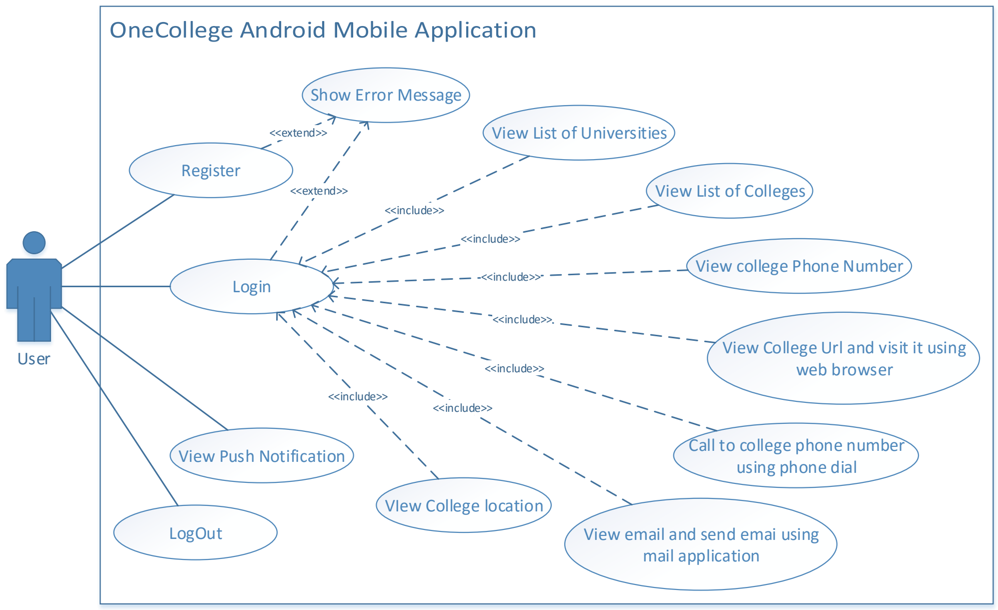

OneCollege Android Mobile Application
Show Error Message
View List of Universities
Register
View List of Colleges
User
Login
View college Phone Number
View College Url and visit it using web browser
Call to college phone number using phone dial
View Push Notification
VIew College location
LogOut
View email and send emai using mail application
<<extend>>
<<extend>>
<<include>>
<<include>>
<<include>>
<<include>>
<<include>>
<<include>>
<<include>>

Figure 8: User Use Case Diagram.

Figure 8: Use Case Diagram of the user shows the simplest interaction of users with the system. A user needs to register to login inside the system. A user can use his/her email and password to register. After registration, a user can log into the application system. A user does not have any rights to modify data inside this application. If the email is already in use and if the login credentials are incorrect while login, the app shows the error message. After login, a user can view the list of universities and colleges. A user can view all the college information and to call, to check the website, to view location and to send an email. Any user who has an application installed can receive and view notification message if connected to the internet. The user does not need to register or log in. A User can log out using the log out button.

3.1.3.2 Administrator Use Case Diagram

The following Figure 9 describes the Use Case diagram of the administrator.

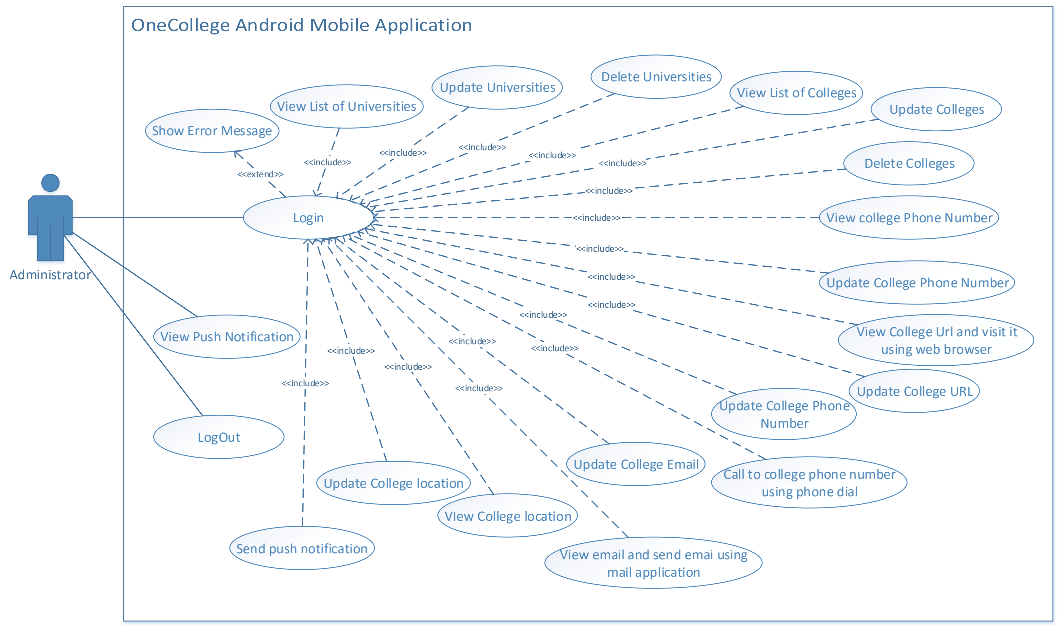

OneCollege Android Mobile Application
Administrator
Login
Show Error Message
View List of Universities
Update Universities
Delete Universities
View List of Colleges
Update Colleges
Delete Colleges
View college Phone Number
Update College Phone Number
View College Url and visit it using web browser
Update College URL
Update College Phone Number
Call to college phone number using phone dial
Update College Email
View email and send emai using mail application
VIew College location
Update College location
Send push notification
View Push Notification
LogOut
<<extend>>
<<include>>

Figure 9: Administrator User Case Diagram.

Figure 9 Use Case Diagram of Administrator shows the simplest interaction of administrator with the application system. Admin has its credentials included inside the system, so the administrator does not need to register. An administrator can directly login using his/her credentials. If the credentials did not match, the application shows an error message.

An administrator has all the rights to access the data in the application. An administrator can view, update, and delete all the data of the university and colleges. An administrator also can send the push notification to all the users who have the application installed on their devices. An administrator can log out using the log out button.

3.1.4 Class Diagram

Class diagram is a UML static structure diagram that shows the system structure with classes, attributes, operations or methods and their relationship. Class diagram can be mapped directly with OOP languages, so it is used widely in the modeling of OOP. A class diagram describes the system responsibilities, design and analysis of the static view of an application, initial base for component and deployment. It is a forward and reverse technique in Software Engineering.

Figure 10 shows the Class diagram of the application:

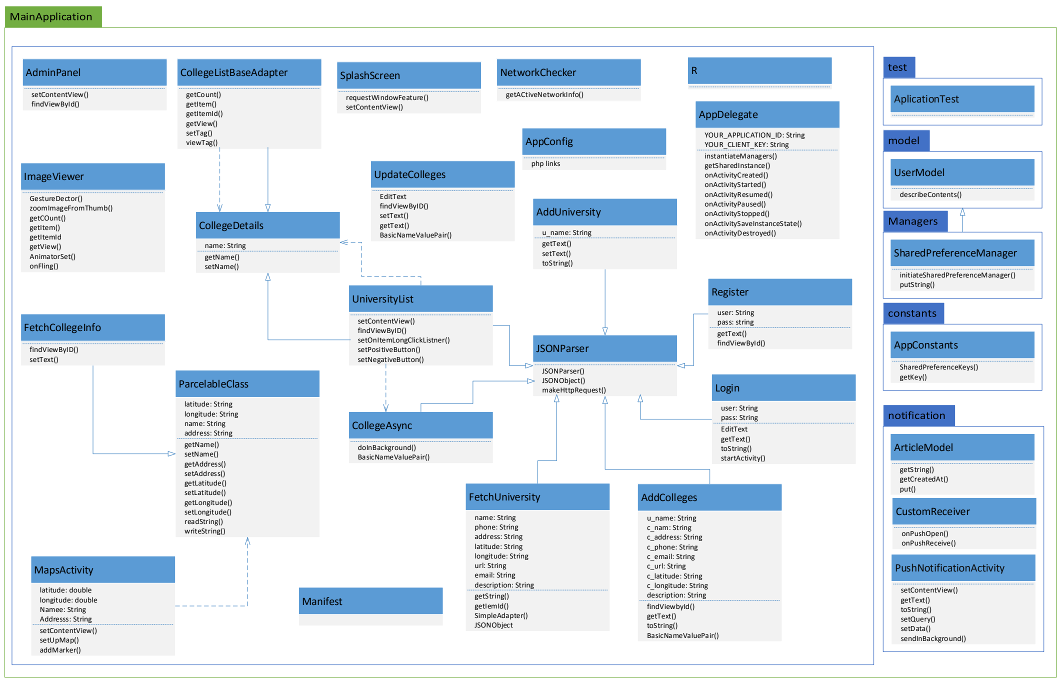


Figure 10: Class Diagram.

Figure 10 defines the various classes used in the application and its links to each other. Those classes are also defined inside own packages.

3.1.5 Sequence Diagram

The sequence diagram is a visual interactive behavior represented in UML diagrams. This diagram emphasizes the message on a time sequence. The sequence diagram captures dynamic behavior, the flow of the message, organization of the object structure and objects interactions of the system.

The application needs an internet connection to run every Activity, so the given sequence diagram runs only when the internet connection is available in the device.

Figure 11-27 shows the Sequence diagrams of this application.

3.1.5.1 Administrator Login Sequence Diagram

Figure 11 describes the sequence how the administrator log in to the application. All the administrator credentials are inserted into the application, so the admin should enter the given correct credentials to the application. After the input of credentials and the Login button is taped, the query is sent to the DB Handler and it checks the credentials in the database. If the data did not match, the application sends an error toast message otherwise admin will be logged in to the admin panel activity.

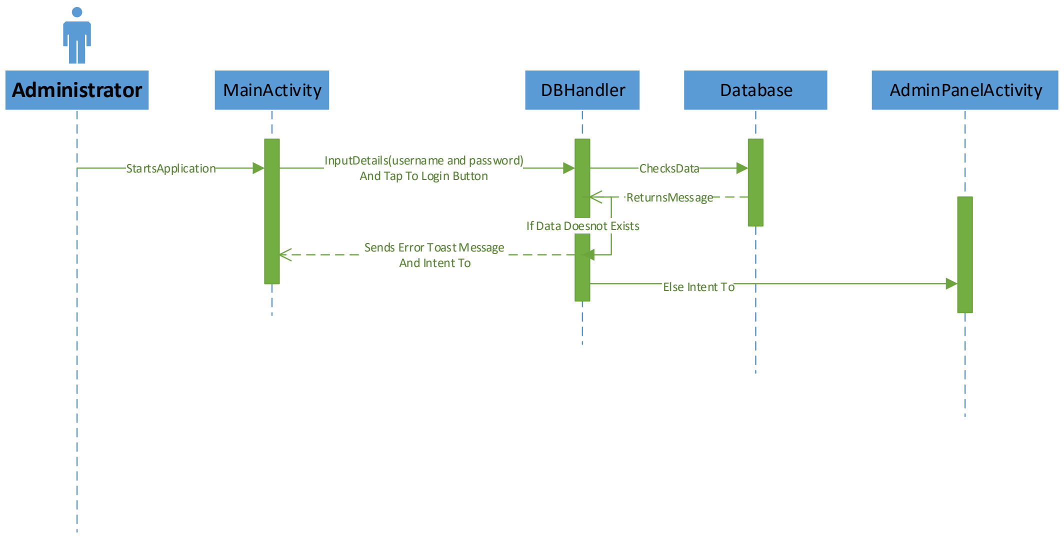


Figure 11: Administrator Login Sequence Diagram.

3.1.5.2 Admin’s Add University Sequence Diagram

Figure 12 describes the sequence how the admin adds university. After the login, the admin will be intended to the admin panel activity. The admin should tap the Add University button to add a university. When tapping Add University button, the application shows add/update university activity. In this activity, the admin can input a university name where the university should be written in small letters and then tap Add/Update Button. Tapping Add/Update button, the application sends the query to the DB Handler. If the input is incorrect, the application toast error message otherwise adds the university is added to the database.

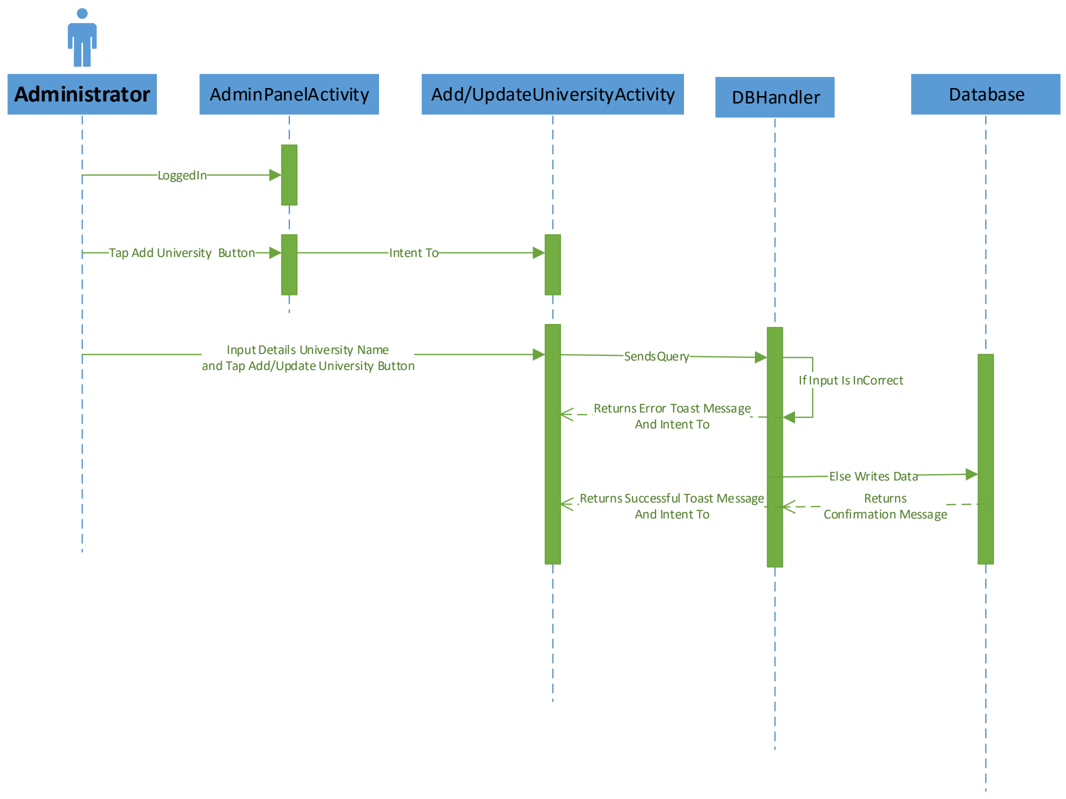
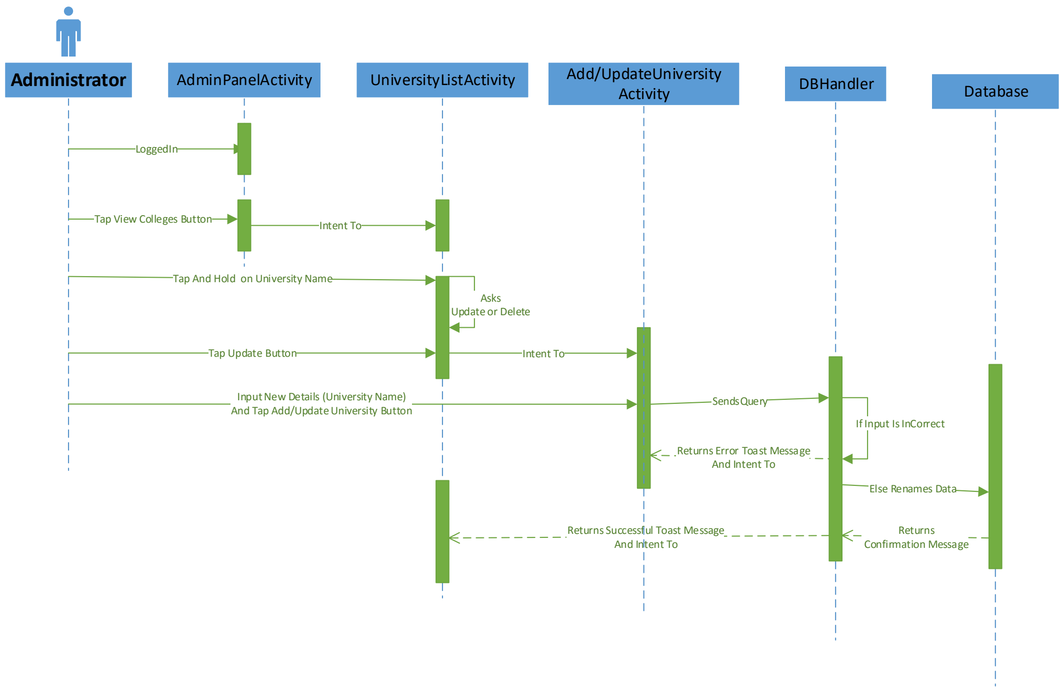

Administrator
AdminPanelActivity
Add/UpdateUniversityActivity
DBHandler
Database
LoggedIn
Tap Add University Button
Intent To
Input Details University Name and Tap Add/Update University Button
Sends Query
If Input Is InCorrect
Returns Error Toast Message And Intent To
Else Writes Data
Returns Successful Toast Message And Intent To
Returns Confirmation Message

Figure 12: Admin’s Add University Sequence Diagram.

3.1.5.3 Admin’s Update University Sequence Diagram

Figure 13 describes the sequence how the admin will update the university. After login, the admin is leads to Admin Panel Activity. Tapping the View Colleges button intents to the university list activity. In this activity, when the admin taps and holds in the university which needs to update shows update or delete options. Tapping the Update button intents to Add/Update University Activity. The admin can input a new name of the university and again tap the Add/Update button. This tap sends a query to DB Handler. If the input is incorrect, a toast error message is shown otherwise University is renamed in the database. Finally, the application intents to university list activity.



Administrator
AdminPanelActivity
UniversityListActivity
Add/UpdateUniversity Activity
DBHandler
Database
LoggedIn
Tap View Colleges Button
Intent To
Tap And Hold on University Name
Asks Update or Delete
Tap Update Button
Intent To
Input New Details (University Name) And Tap Add/Update University Button
Sends Query
If Input Is InCorrect
Returns Error Toast Message And Intent To
Else Renames Data
Returns Successful Toast Message And Intent To
Returns Confirmation Message

Figure 13: Admin’s Update University Activity Sequence Diagram.

3.1.5.4 Admin’s Delete University Sequence Diagram

Figure 14 describes the sequence how the admin will delete a university. After the login, the admin is intent to Admin Panel Activity. Tapping the View Colleges button intents to the university list activity. In this activity, when the admin taps and holds in the university which needs to be deleted shows Update or Delete options. Tapping Delete button sends a query to the DB Handler. The DB Handler sends a delete query to delete the university from the database. When the delete is successful, a successful message is toasted in the UI and the application is intended to the university list activity.

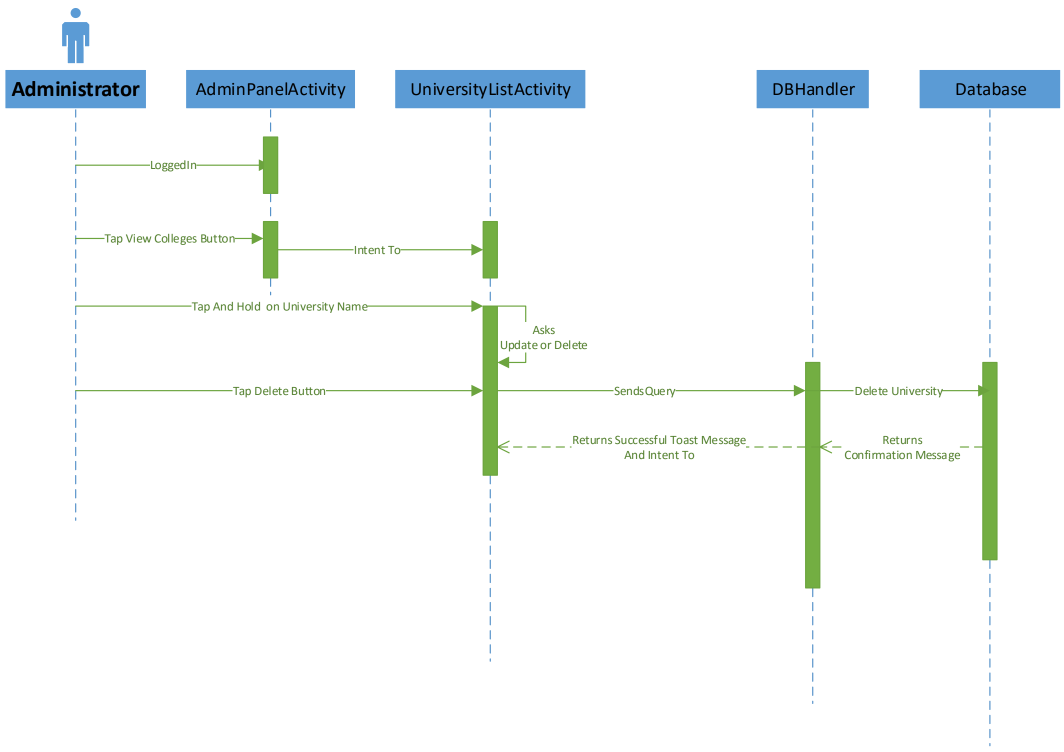


Figure 14: Admin’s Delete University Sequence Diagram.

3.1.5.5 Admin’s Add College Sequence Diagram

The Figure 15 describes the sequence how the admin adds College. After the login, admin is intended to the admin panel activity. The admin should tap the Add College button to add a college. After tapping the Add College Button, the application is intended to the add college activity. In this activity, admin can input College details (University Name, College Name, Address, Phone Number, E-mail, URL, Latitude, Longitude, Description) and the Add button should be tapped where the university should be in small letters and the phone number should be in the international format. Tapping Add button, the application sends the query to the DB Handler. If the input is incorrect, application toast error message otherwise adds college to the database. Finally, the application intents to the admin panel activity.

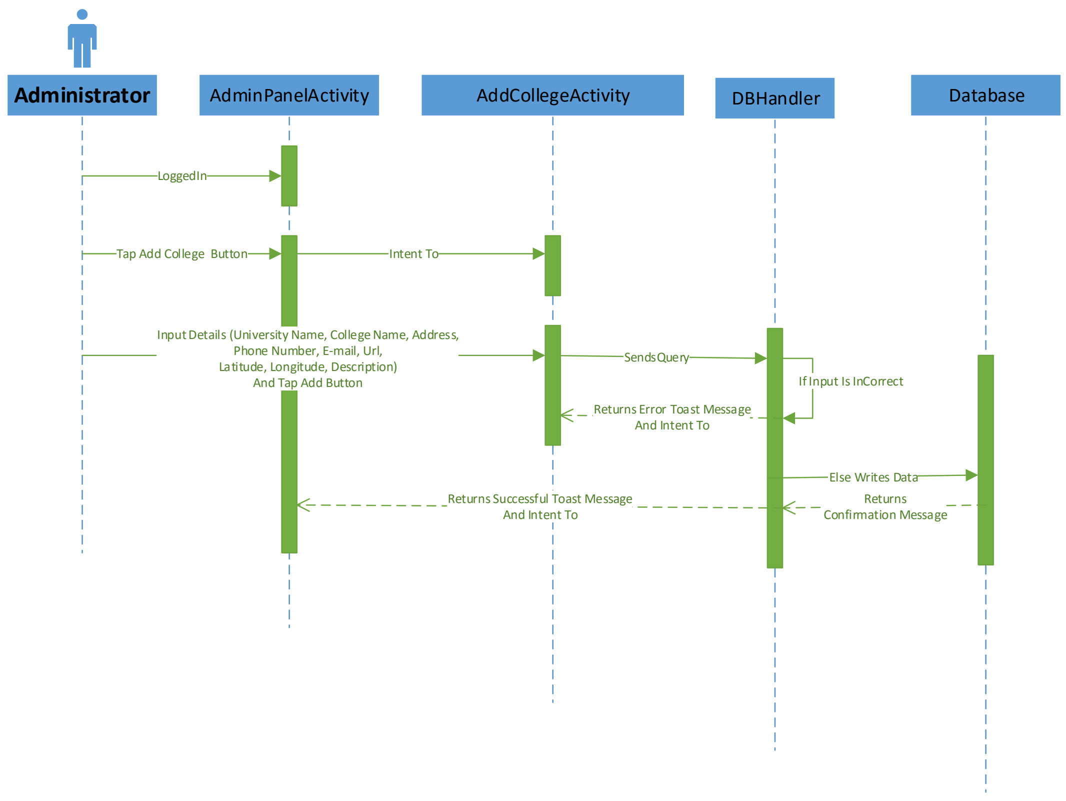

Administrator
AdminPanelActivity
AddCollegeActivity
DBHandler
Database
LoggedIn
Tap Add College Button
Intent To
Input Details (University Name, College Name, Address, Phone Number, E-mail, Url, Latitude, Longitude, Description) And Tap Add Button
Sends Query
If Input Is InCorrect
Returns Error Toast Message And Intent To
Else Writes Data
Returns Successful Toast Message And Intent To
Returns Confirmation Message

Figure 15: Admin’s Add College Sequence Diagram.

3.1.5.6 Admin’s Update College Sequence Diagram

Figure 16 describes the sequence how the admin will update college. After the login, the admin is intended to the admin panel activity. Tapping the View Colleges button intents to the university list activity. In this activity, the admin can tap on a university whose affiliated college needs to be updated. When the admin taps and holds in the college name, which needs to be updated shows Update or Delete options. Tapping the Update button intents to the update college activity. The admin can input the new details of the college (University Name, College Name, Address, Phone Number, E-mail, URL, Latitude, Longitude, Description) and to tap Update button. The university name should be in small letters, and the phone number in the international format. The tap sends a query to the DB Handler. If the input is incorrect, toast error message is shown otherwise renames the college information to the database. Finally, the application intents to the update college activity.

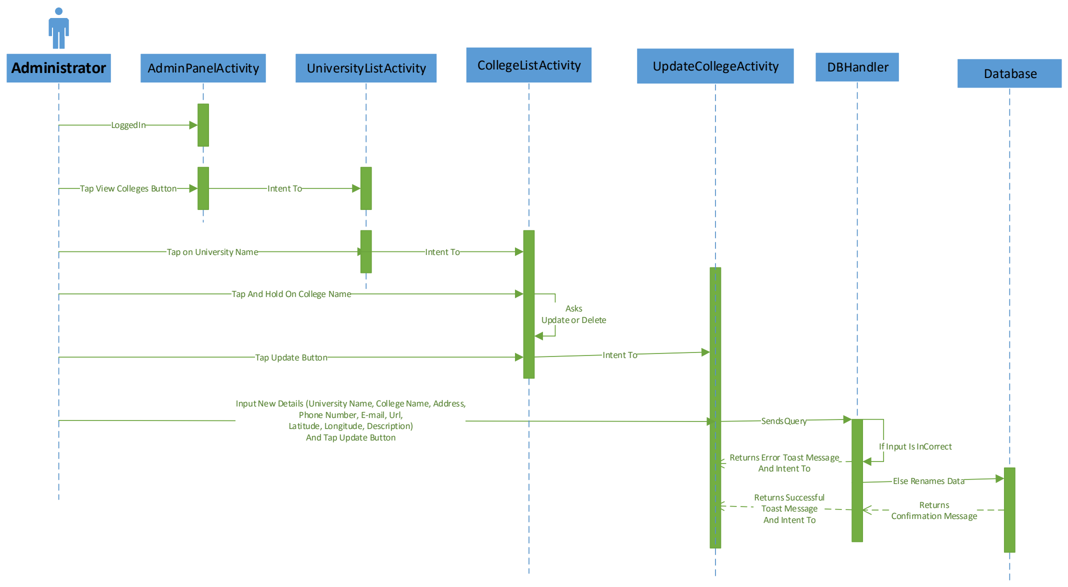

Administrator
AdminPanelActivity
UniversityListActivity
CollegeListActivity
UpdateCollegeActivity
DBHandler
Database
LoggedIn
Tap View Colleges Button
Intent To
Tap on University Name
Intent To
Tap And Hold On College Name
Asks Update or Delete
Tap Update Button
Intent To
Input New Details (University Name, College Name, Address, Phone Number, E-mail, Url, Latitude, Longitude, Description) And Tap Update Button
Sends Query
If Input Is InCorrect
Returns Error Toast Message And Intent To
Else Renames Data
Returns Successful Toast Message And Intent To
Returns Confirmation Message

Figure 16: Admin’s Update College Sequence Diagram.

3.1.5.7 Admin’s Delete College Sequence Diagram

Figure 17 describes the sequence how the admin deletes the college. After the login, the admin is intended to the admin panel activity. Tapping the View Colleges button intents to the university list activity. In this activity, the admin can tap on a university whose affiliated college needs to be deleted. When the admin taps and holds the college name, which needs to be deleted shows update or delete options. Tapping the Delete button sends a query to the DB Handler. The DB Handler sends delete query to delete the college and all its information from the database. When the delete is successful, successful toast message is seen in the UI and is intended to the college list activity.

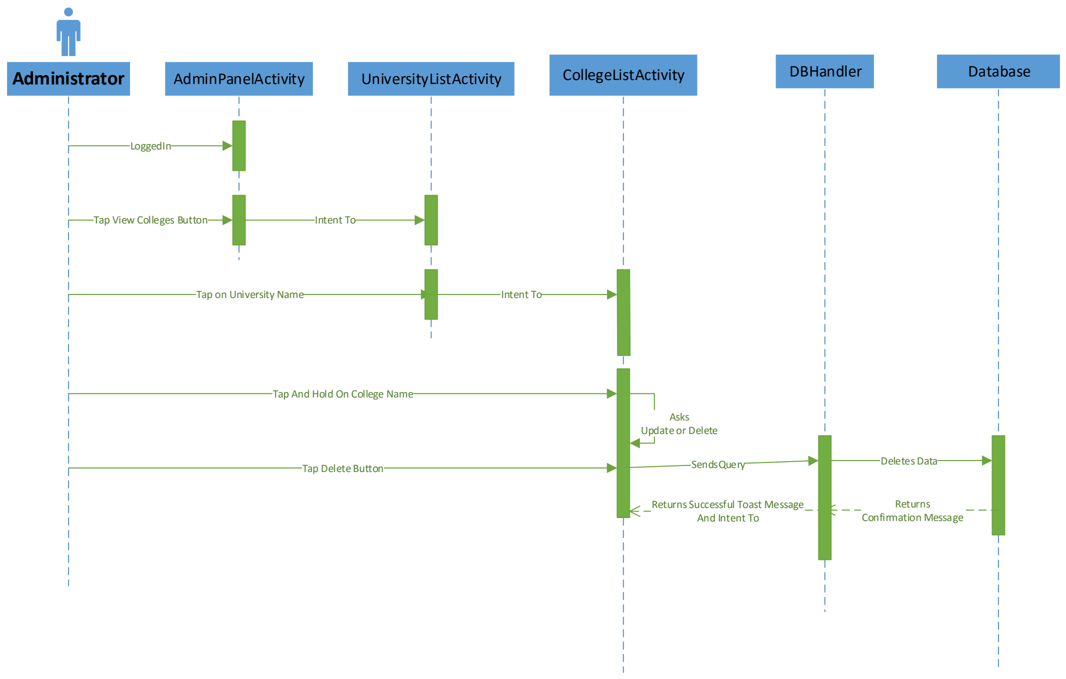


Figure 17: Admin’s Delete College Sequence Diagram.

3.1.5.8 Admin's Push Notification Sequence Diagram

Figure 18 describes the sequence how the admin sends a push notification. After the login, the admin will be intended to the admin panel activity. Admin should tap the Send Notification button. Tapping the Send Notification button, the application intents to the Send Notification Activity. In this activity, the admin can input the notification message and have to tap the Send Push button. Tapping the Send Push button, the application sends the query to Parse. Successful toast message is seen, and that sends notification message to all the application installed on the device.

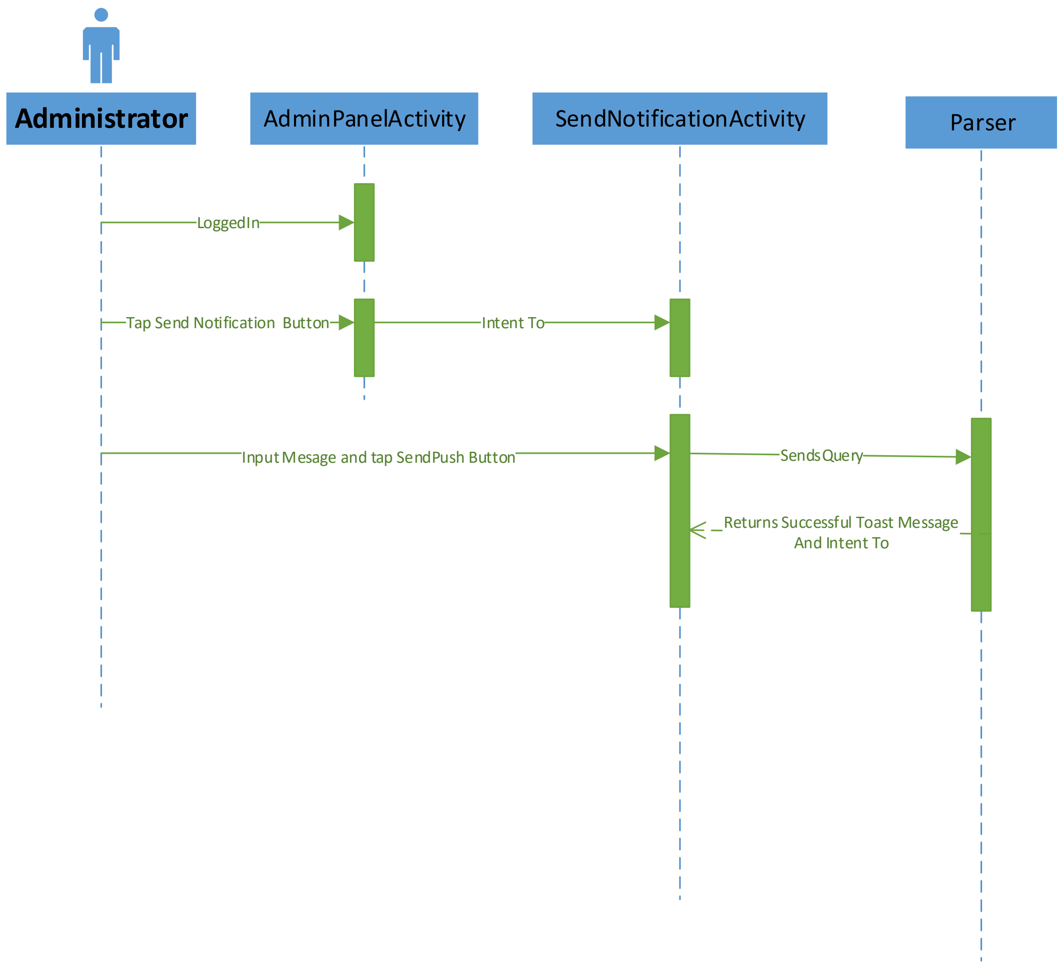

Administrator
AdminPanelActivity
SendNotificationActivity
Parser
LoggedIn
Tap Send Notification Button
Intent To
Input Mesage and tap SendPush Button
Sends Query
Returns Successful Toast Message And Intent To

Figure 18: Admin's Push Notification Sequence Diagram.

3.1.5.9 User’s Registration Sequence Diagram

Figure 19 describes how the user get registration in the application. Starting the application shows the main activity. When the user taps the Register button, it intent to the registration activity. The user can input a new email, a password, and tap Sign Up button. This action sends the query to DB Handler, and it checks data from the database. If the data already exists, the DB Handler sends the toast error message otherwise sends toast successful message and intent to the university list activity.

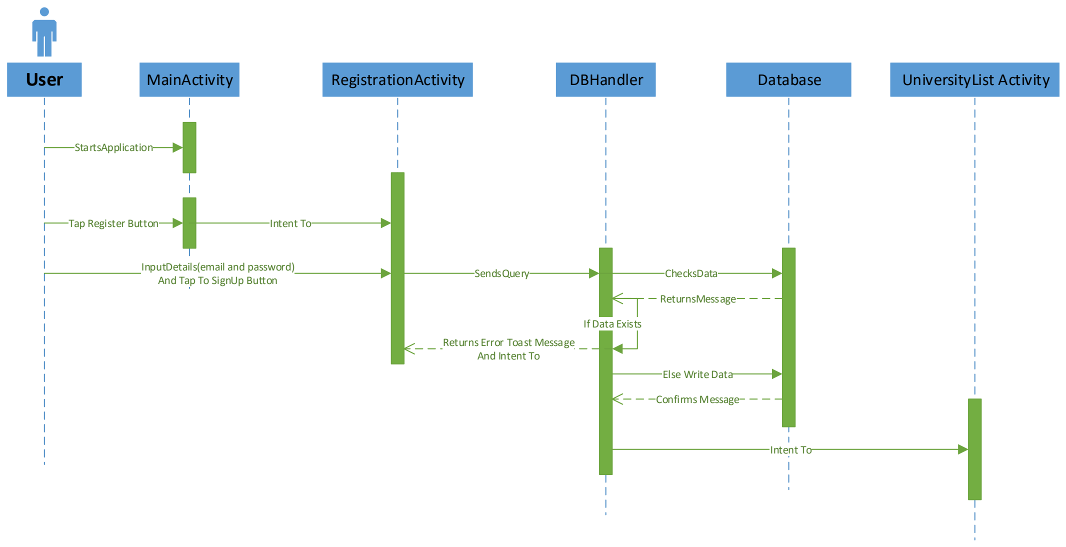


Figure 19: User’s Registration Sequence Diagram.

3.1.5.10 User's Login Sequence Diagram

Figure 20 describes how the user log in to the application. Starting the application shows the main activity. The user can input registered email, password, and tap Login button. This action sends the query to DB Handler, and it checks the data from the database. If the data does not exist, the DB Handler sends the toast error message otherwise sends successful toast message. Finally, it intent to the university list activity.

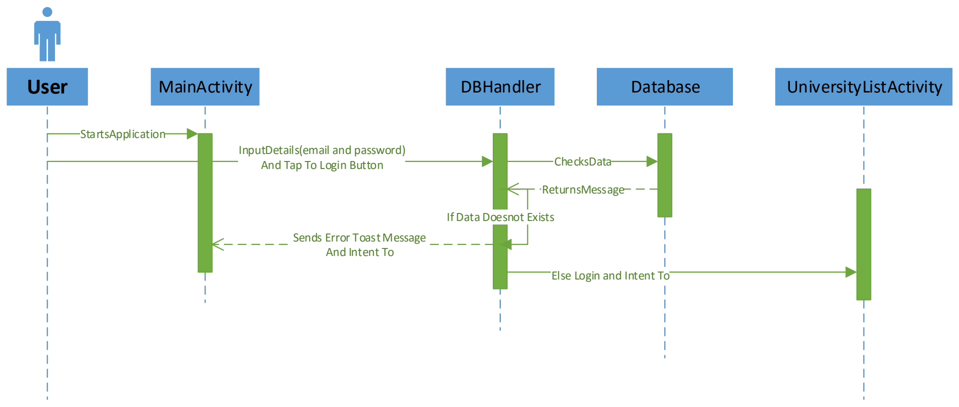


Figure 20: User's Login Sequence Diagram.

3.1.5.11 User’s College List Sequence Diagram

Figure 21 describes how the user view college list in the application. After the user logged in, the application intent to the university list activity. The user can tap the university whose affiliated college to be viewed. When the user taps the university name, the DB Handler sends the query to the database and return message is sent to the DB Handler. Finally, the user is intended to the college list activity.

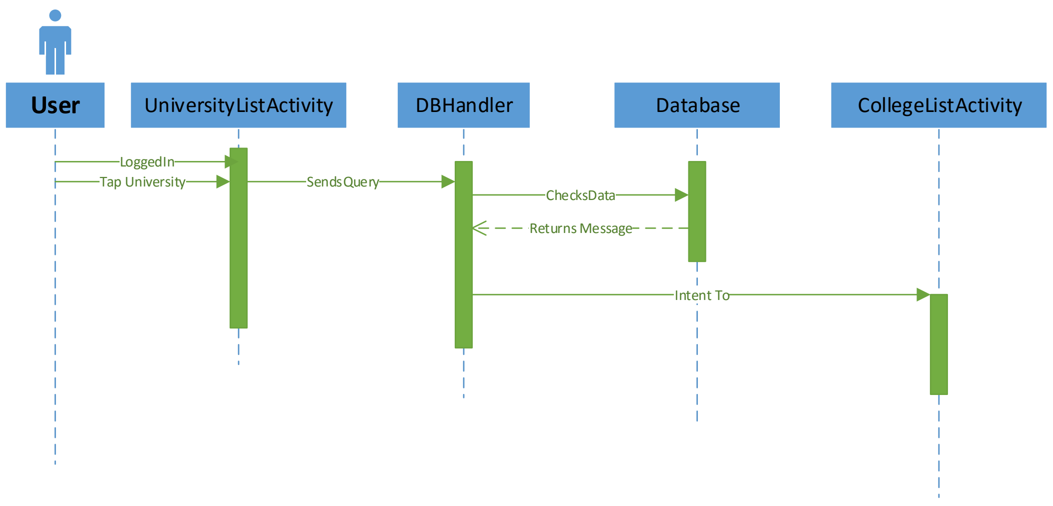


Figure 21: User’s College List Sequence Diagram.

3.1.5.12 User's College Info Sequence Diagram

Figure 22 describes how the user view the college Information on the application. After the user logged in, the application intents to the University List Activity. The user can tap the University whose affiliated College to be viewed. When the user taps the university name, DB Handler sends the query to the database and message is returned to the DB Handler and the application is intended to the College List Activity. Tapping on the any college that needs to be viewed, will intent to the College Info Activity.

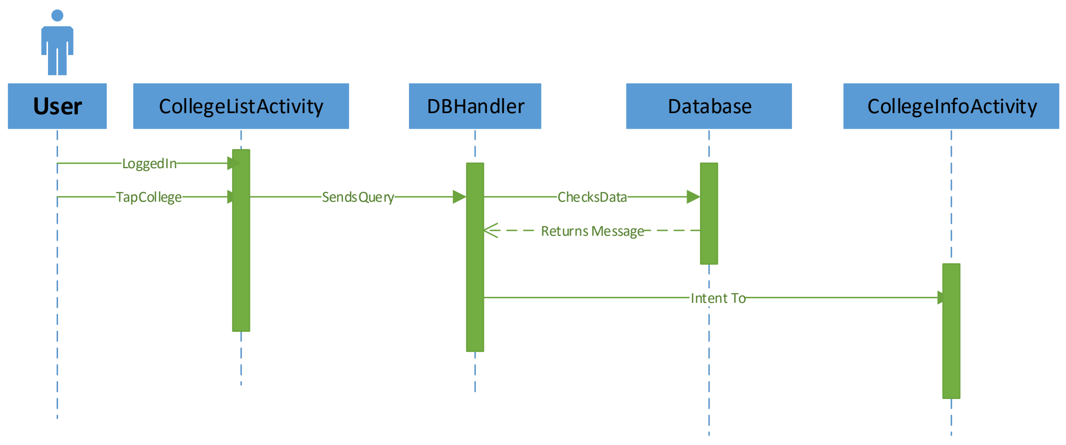


Figure 22: User's College Info Sequence Diagram.

3.1.5.13 User’s Visit College Website Sequence Diagram

Figure 23 describes how the user visit college’s website in the application. After the user logged in and intents to the College Info Activity, the user can tap Visit Website button. When the user taps Visit Website Button, DB Handler sends the query to the database and the message returned to the DB Handler. If the data does not exist, error message toasted otherwise the application ask permission to use the web browser installed on the device. The user can select the web browser and that intents application to the web browser that loads the website.

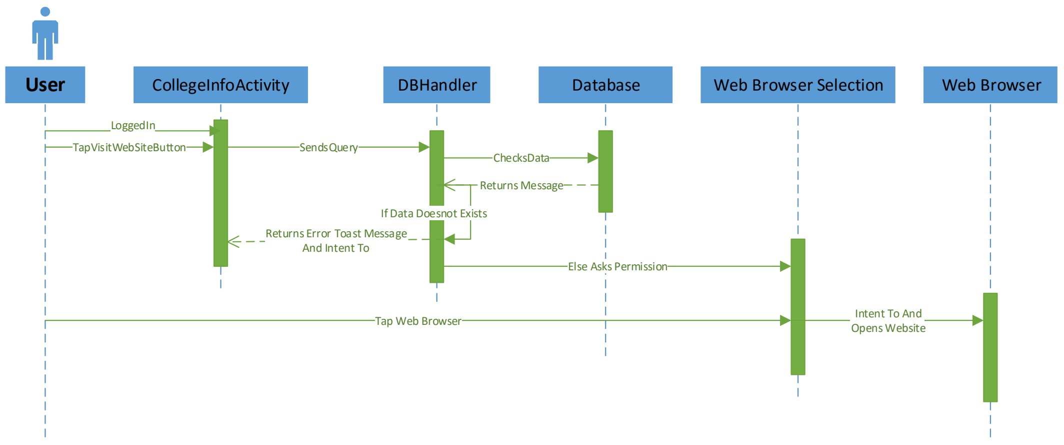


Figure 23: User’s Visit Website Sequence Diagram.

3.1.5.14 User’s Call College Phone Number Sequence Diagram

Figure 24 describes how the user call college’s phone number from the application. After the user logged in and intents to the College Info Activity, the user can tap the Call button. When the user taps Call button, DB Handler sends the query to the database and message is returned to the DB Handler. If the data does not exist, error message is toasted otherwise the application asks permission to use the dialer application installed on the device. The user can select the dialer application, that will make the call to college’s phone number.

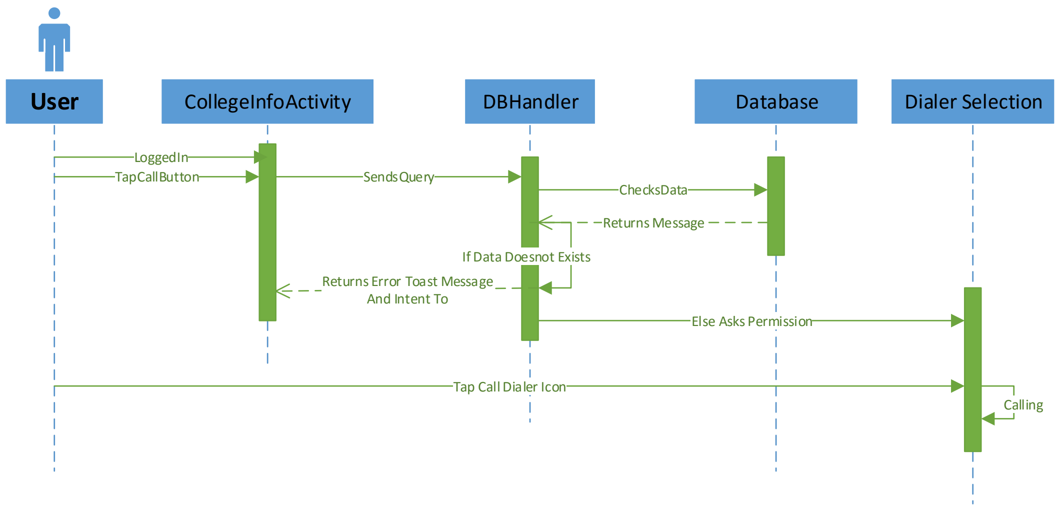


Figure 24: User’s Call College Phone Number Sequence Diagram.

3.1.5.15 User’s Send Email to College Sequence Diagram

Figure 25 describes how the user sends an email to the college’s email address from the application. After the user logged in and intents to the College Info activity, the user can tap an Email button. When the user taps Email button, DB Handler sends the query to the database and message is returned to the DB Handler. If the data does not exist, error message is toasted otherwise the application will ask permission to use the mailing application installed on the device. The user can select mailing application, that will intents application to the selected application. Now, the user can type the subject and the message and can tap Send button to send the email.

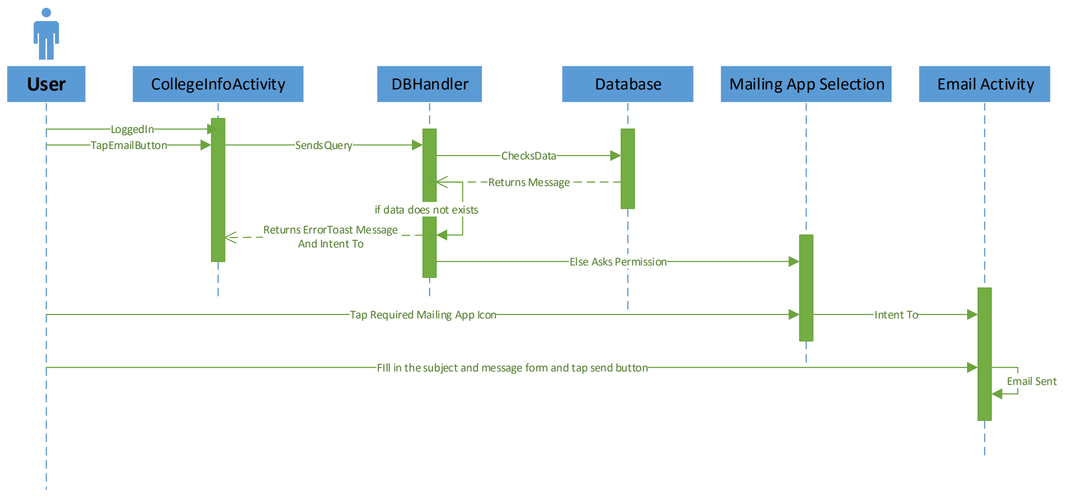


Figure 25: User’s Send Email to College Sequence Diagram.

3.1.5.16 User’s View of College Map Sequence Diagram

Figure 26 describes how the user view college’s map from the application. After the user logged in and intents to the college info activity, the user can tap the View Map button. When the user taps View Map button, DB Handler sends the query to the database and the message is returned to the DB Handler. If the data does not exist, error message is toasted otherwise the application will be intended to the map activity.

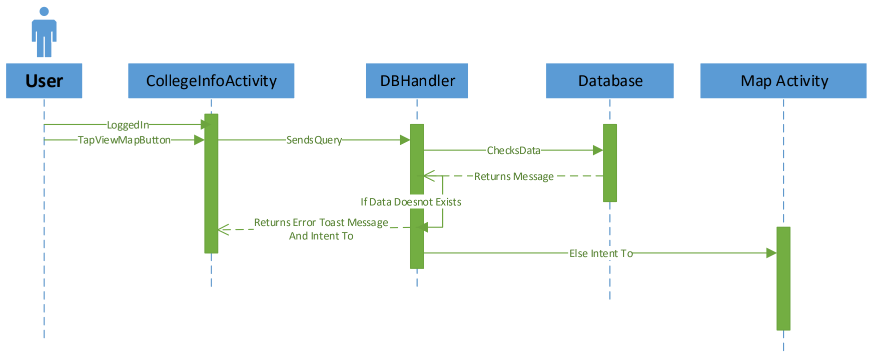


Figure 26: User’s View of College Map Sequence Diagram.

3.1.5.17 User's Receive Notification Sequence Diagram

Figure 27 describes how the user receives the push notification message on the application. User does not need to be logged in to the application, only the installation of the application is required to receive the push notification message. When the user receives the message, the tapping of the notification can let user see the message sent by admin.

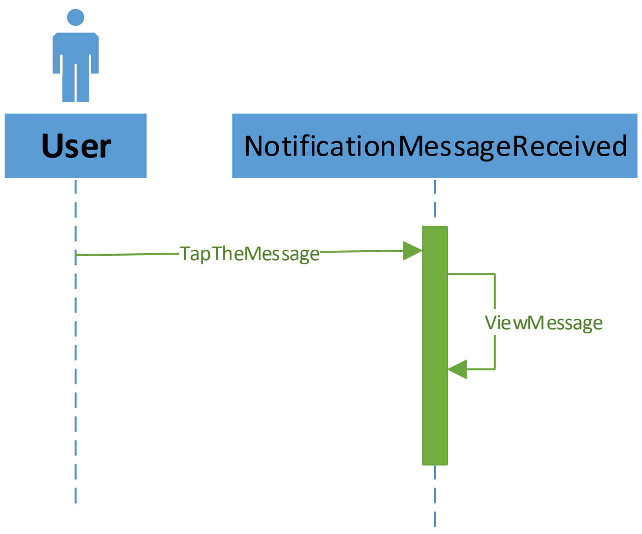


Figure 27: User's Receiver Notification Sequence Diagram.

3.1.6 Component Diagram

Figure 28 elaborates how each component works on the device and to the application. The services and device need an internet connection to make communication. The UI is divided into two major views. Admin view can be accessed only by Admin and User view by the user. Server side has database connection using PHP, and other services are Google Maps API and Push Notification which also communicates via an internet connection.

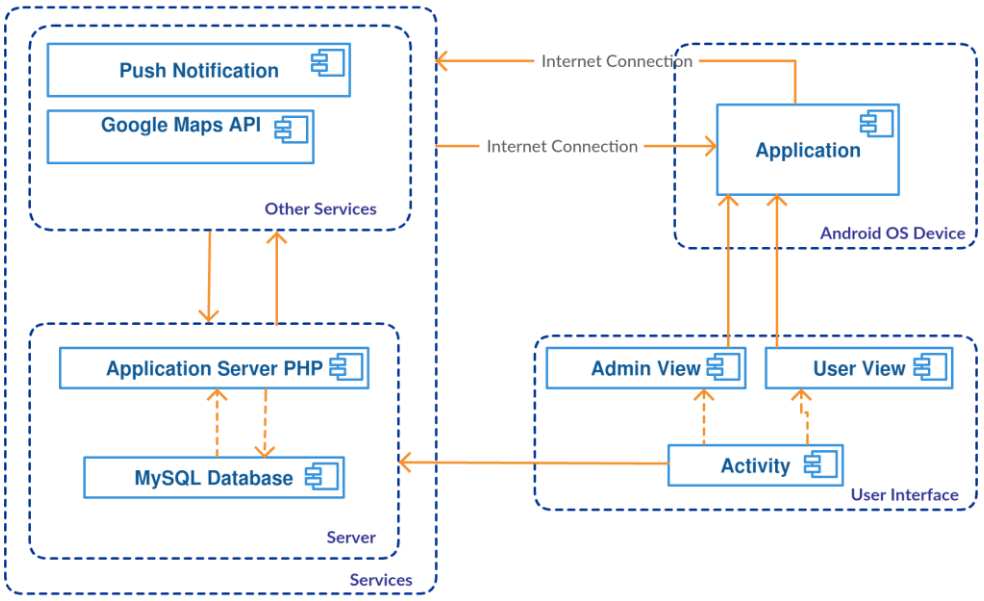


Figure 28: Component Diagram.

5 DATABASE

5.1 Design of the Database

A database is created using MySQL Database Management System. All the data are stored inside VAMK's School server (www.mysql.cc.puv.fi). PHP is used in the server side script. PHP parse data to JSON format because an Android application cannot be connected directly to the PHP. A URL handler is created to make communication between the application and database that links Java to PHP.

ER Diagram is given below:

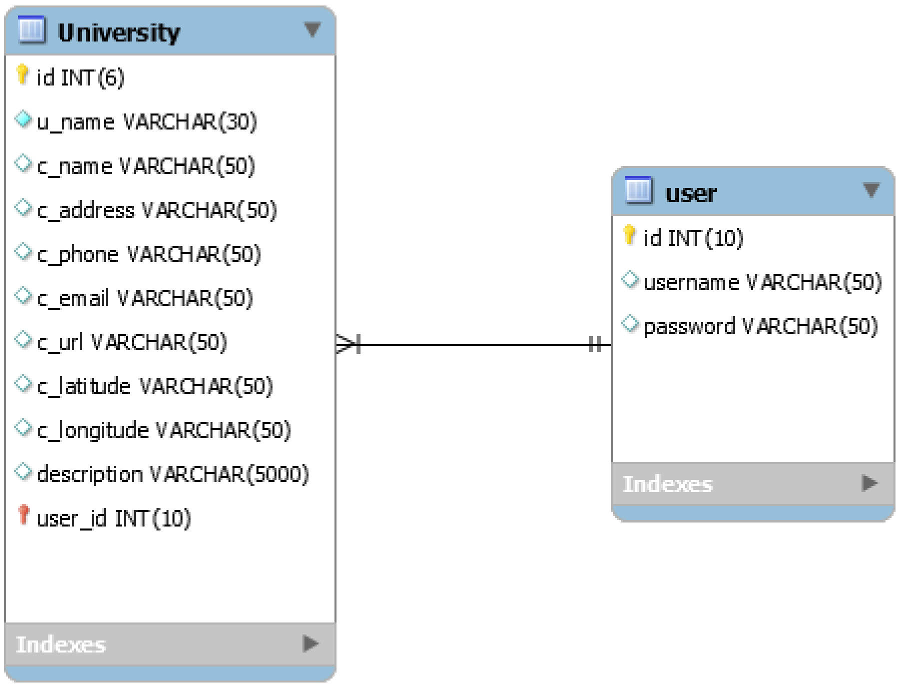


Figure 29: ER Diagram.

As shown in the Figure 29. The database consists of one table that is the user and many other university tables. The user table contains of two columns that are username and password. User name stores the user's email and also the admin's user name that is created during the project.

The password column holds every registered user's password and the admin's password. The university table is created every time administrator adds university. Also, university can be updated or renamed and deleted. Adding of college to its affiliated University are stored as rows in the respective University. Admin can delete University table which also can delete college and college data.

Figure 30 shows some examples of a university table that has been created:

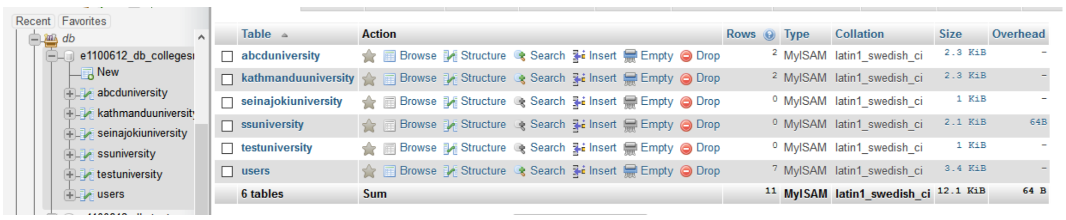


Table	Action	Rows	Type	Collation	Size	Overhead
abcduniversity	Browse Structure Search Insert Empty Drop	2	MyISAM	latin1_swedish_ci	2.3 KiB	-
kathmanduuniversity	Browse Structure Search Insert Empty Drop	2	MyISAM	latin1_swedish_ci	2.3 KiB	-
seinajokiuniversity	Browse Structure Search Insert Empty Drop	0	MyISAM	latin1_swedish_ci	1 KiB	-
ssuniversity	Browse Structure Search Insert Empty Drop	0	MyISAM	latin1_swedish_ci	2.1 KiB	64B
testuniversity	Browse Structure Search Insert Empty Drop	0	MyISAM	latin1_swedish_ci	1 KiB	-
users	Browse Structure Search Insert Empty Drop	7	MyISAM	latin1_swedish_ci	3.4 KiB	-
6 tables	Sum	11	MyISAM	latin1_swedish_ci	12.1 KiB	64 B

Figure 30: Screen Shot of Database Tables.

The inputs like registration can be called using the server's URL (http://www.cc.puv.fi/~e1100612/Collegesnepal/register.php). For every task, PHP is scripted in the server's URL like Login, add University, update University, Add College, Delete University and others. These PHP files are kept inside public_html directory. Those PHP files are executed to write, to update and to fetch data to and from the application. It is done using POST method. The output value of the data is sent as JSON format to the application. This way application could communicate with the database.

7 GRAPHICAL USER INTERFACE DESIGN

This section explains about the overview of User Interface of the application. The language that is used to design an Android application is XML. XML files are created inside a layout package of the application project. UI are also knowns as layout while developing an application. UI are the activity that is seen on the screen where the user can do actions or place input to the application.

The screen shots of the application's UI made are given below.

7.1 Splash Screen

Figure 31 shows the start of the application called Splash Screen in Android. When the application icon is tapped, the application starts this Splash Screen, which has loading animation. After some seconds, this screen of the application will be intent to Main Activity.

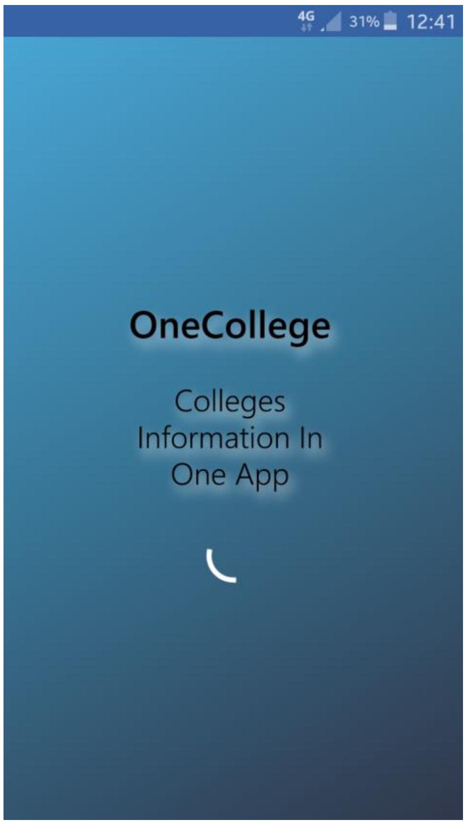


Figure 31: Splash Screen Shot.

7.2 Main Activity

Figure 32 shows the main activity of the application. This activity consists of two input text fields, email and password and two buttons Login and Register. A new user can tap Register button to make a registration. A registered user can input his/her registered credentials and tap the Login button to log in to the application. The admin’s credentials are created while developing the application, so the admin can directly input his/her credentials and log in.

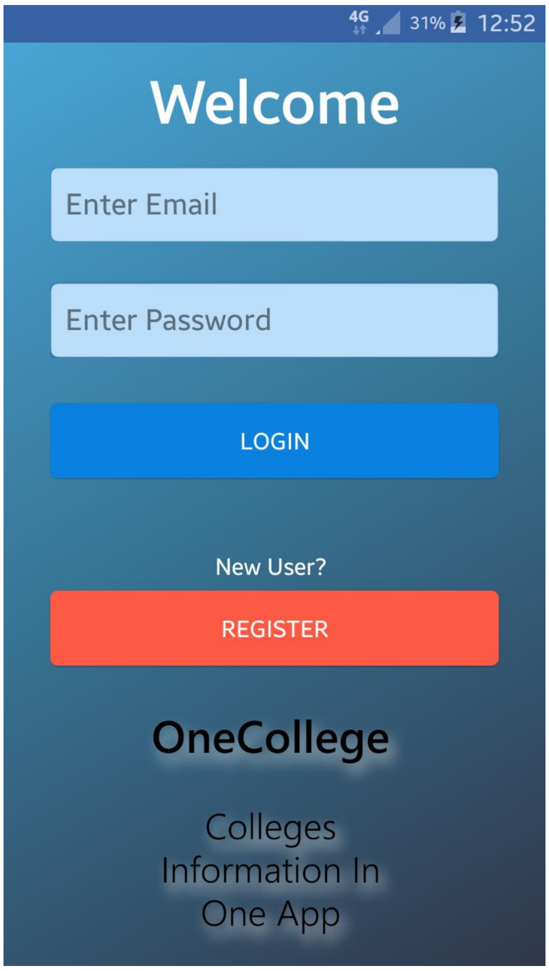


Figure 32: Main Activity Screen Shot.

7.3 Admin’s Panel Activity

Figure 33 shows the admin panel activity of the application. After the admin logs in, this activity starts. The activity consists of five buttons. Add College Button intents to the activity where college is added according to the university. Add University button intents to the activity where university is added. View Colleges Button intents to the university list activity. Send Notification button intents to the push notification activity. Log out button will log out the admin out of the application. The option button consists of a Logout button which is viewed on every Activity.

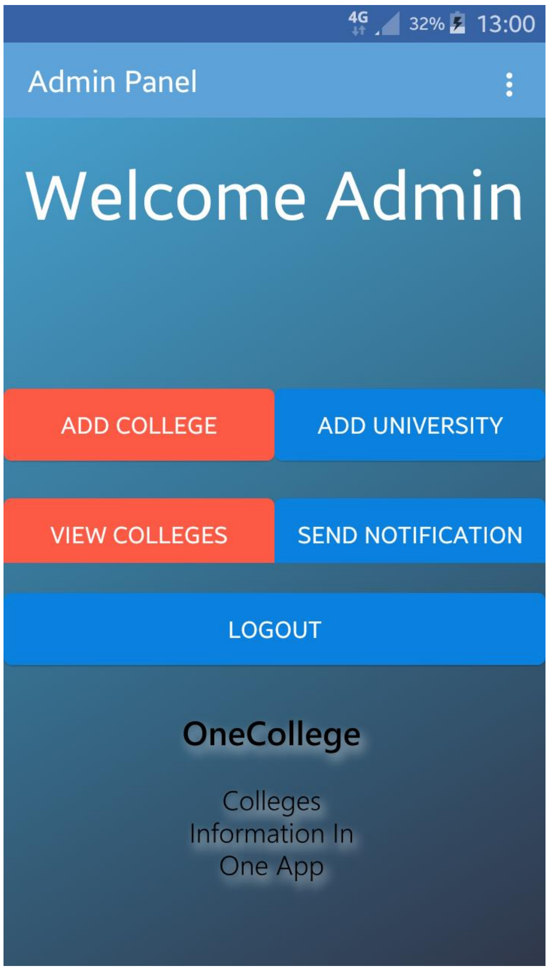


Figure 33: Admin’s Panel Activity Screen Shot.

7.4 Admin’s Add University Activity

Figure 34 shows add university activity of the application. The activity consists of input text box where the university name is given. University name should be written in small letters. Add/Update button adds the university to the database which also makes the table inside the database.

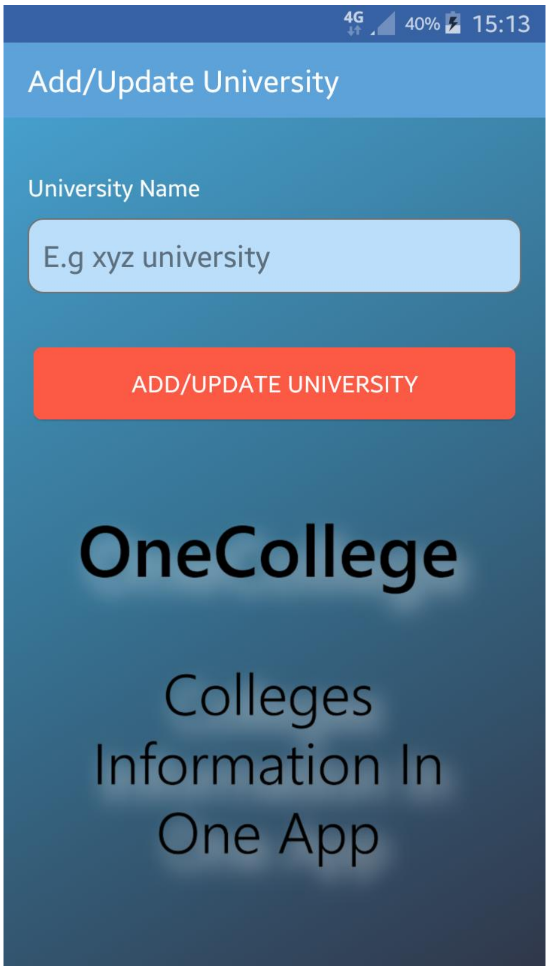


Figure 34: Admin’s Add University Activity Screen Shot.

7.5 Admin’s Add College Activity

Figure 35 shows add college activity of the application. The activity consists of nine input text fields. By filling those fields and with the tap of Add button will adds college according to the university mentioned. The phone number should be international format, and the university name should exist to add to the affiliated the university. The option button has log out options that logs out from the application.

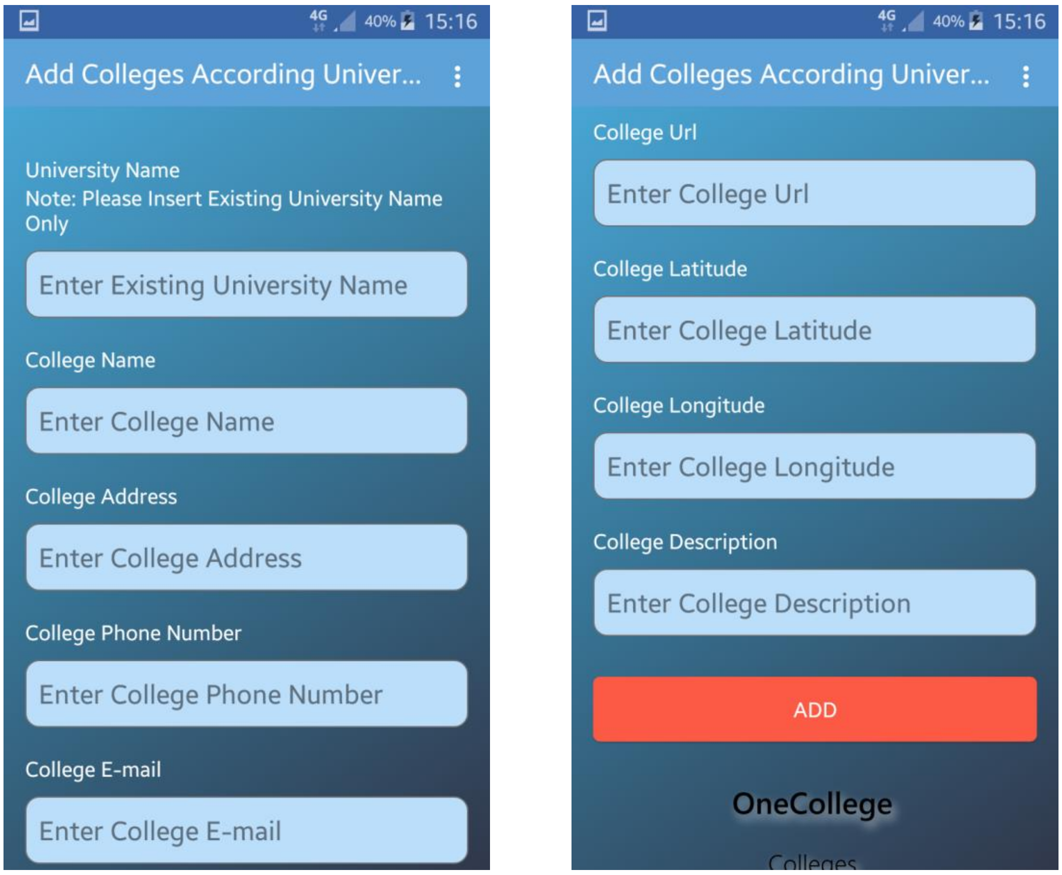
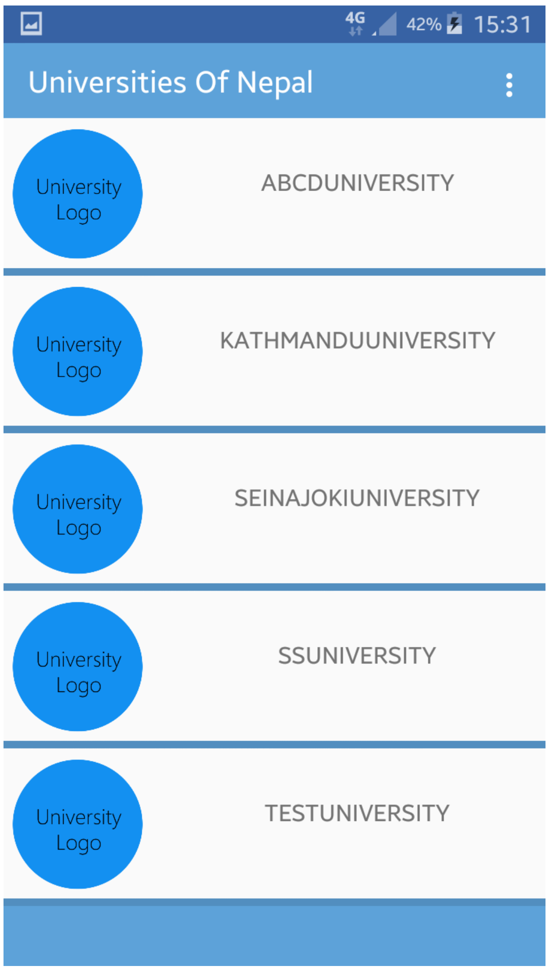


Figure 35: Admin’s Add College Activity Screen Shot.

7.6 University List Activity

Figure 36 shows university list activity. The activity is same for user and also for admin. The activity consists of list of universities that are added by admin. MySQL database is fetched to get university list. The options button consists of log out option, that logs out of the application.



ABCDUNIVERSITY

KATHMANDUUNIVERSITY

SEINAJOKIUNIVERSITY

SSUNIVERSITY

TESTUNIVERSITY

Figure 36: University List Activity Screen Shot.

7.7 College List Activity

Figure 37 shows the college list activity. The Activity is same for user and admin. The activity consists of the list of colleges that are added by admin. MySQL database fetched the list to get college names. If an admin is logged in, the admin can update and delete college from this Activity. The options button consists of log out option, that logs out of the application.

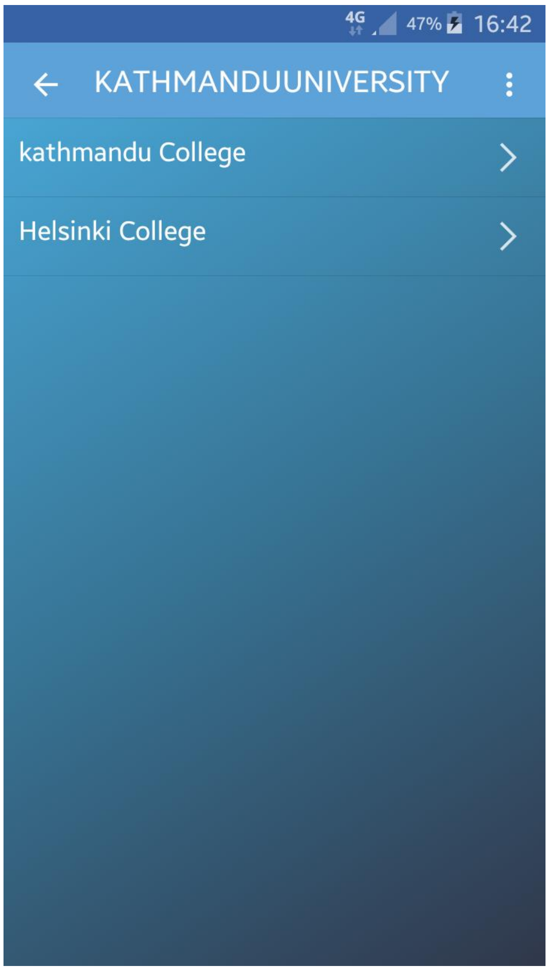
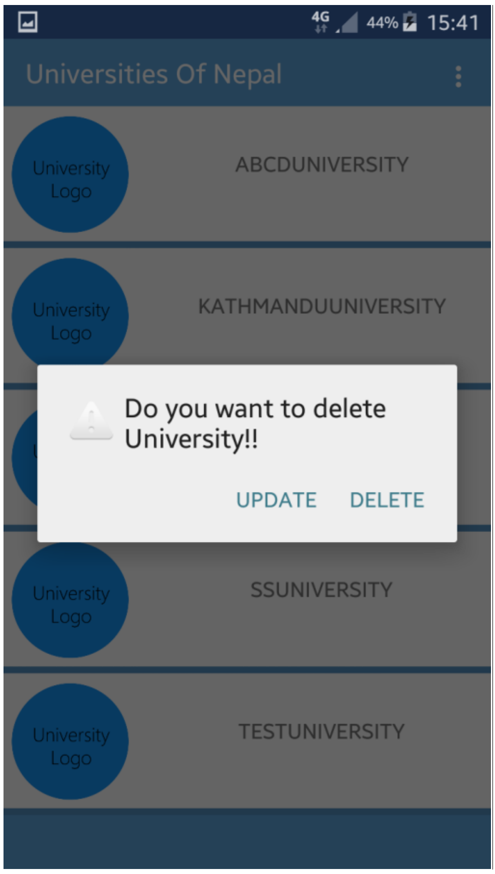


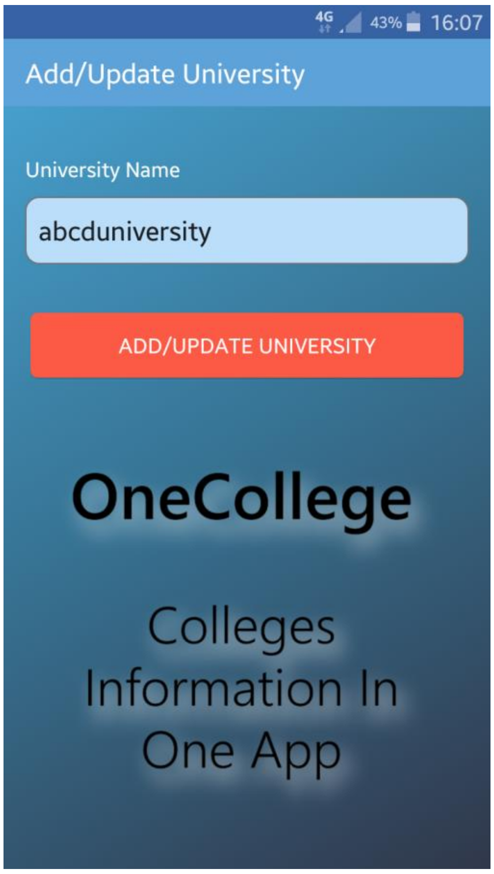
Figure 37: College List Activity Screen Shot.

7.8 Admin’s Update/Delete University Activity

Figure 38 shows update or delete options for the university. When the admin taps and holds the university name which has to be updated or deleted, this option pops out. Tapping Update button intents, the application to update activity where new name of the university can be inputted. Tapping Add/Update Button, updates university name. Tapping Delete button deletes the university.



Do you want to delete University!!

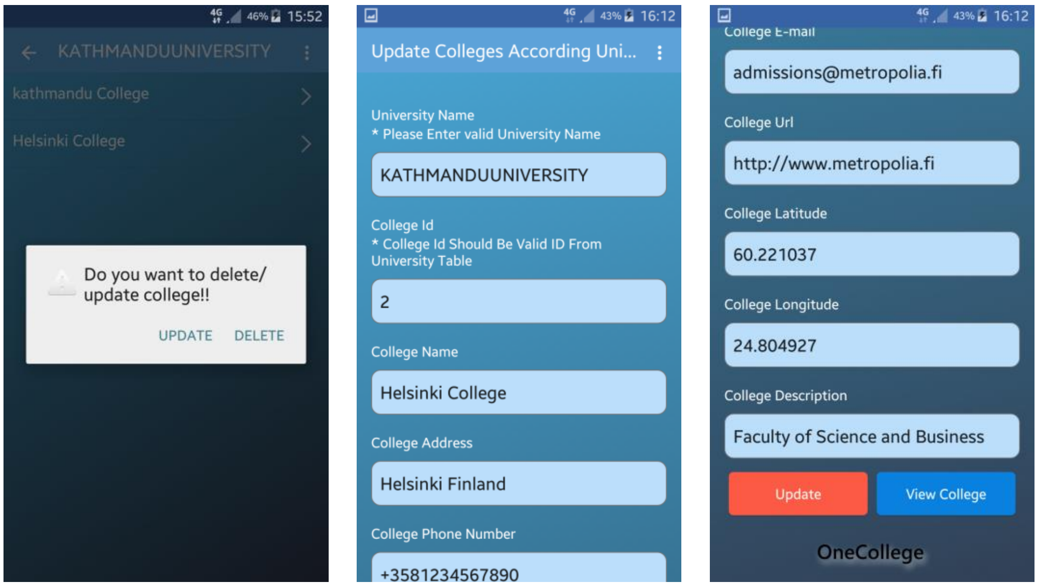


abcduniversity

Figure 38: Admin’s Update/Delete University Activity Screen Shot.

7.9 Admin’s Update/Delete College Activity

Figure 39 shows the option to update and to delete college. When admin taps and holds on college name which is to be updated or deleted, Update and Delete option pop's out. Delete button deletes the selected college. Update button intents the application to the update college activity. Inside update college activity, the admin can rename every detail except college's ID to update college and can tap the Update button. The View button intents the application to the college info activity.



kathmandu College

Helsinki College

Do you want to delete/ update college!!

UPDATE DELETE

University Name
* Please Enter valid University Name

KATHMANDUUNIVERSITY

College Id
* College Id Should Be Valid ID From University Table

2

College Name

Helsinki College

College Address

Helsinki Finland

College Phone Number

+3581234567890

College E-mail

admissions@metropolia.fi

College Url

http://www.metropolia.fi

College Latitude

60.221037

College Longitude

24.804927

College Description

Faculty of Science and Business

Figure 39: Admin's Update/Delete College Activity Screen Shot.

7.10 Admin's Push Notification Activity

Figure 40 shows the Push Notification Activity. This Activity consists of a large edit text box, where the admin can input the message and tapping Send Push button sends push notification message.

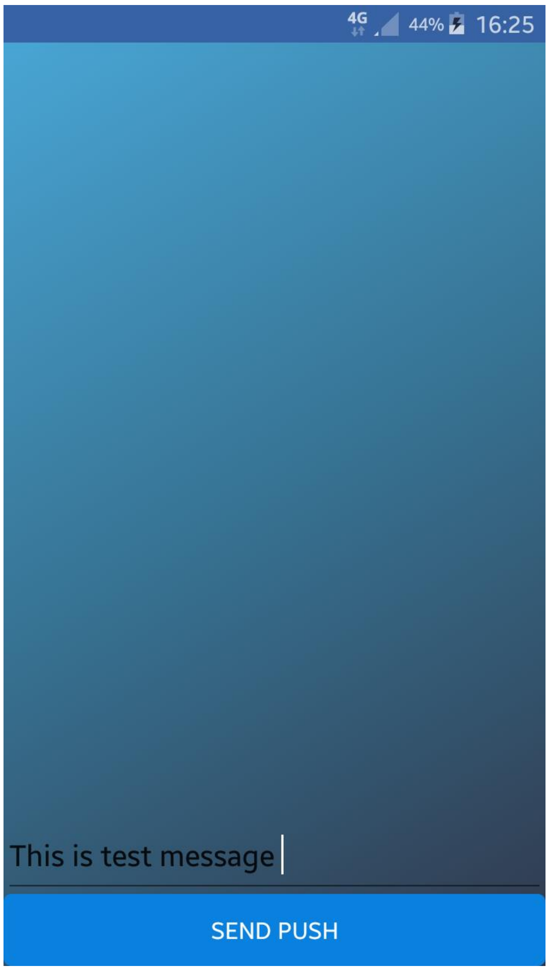


Figure 40: Admin’s Push Notification Activity Screen Shot.

7.11 College Info Activity

Figure 41 shows the college info activity. This Activity consists information about the college. The text field consists of college name, a phone number, a web URL and the descriptions. There are five buttons in this Activity. Visit Website button intents the application to select the different web browsers installed on the device. Call button let the user call to the given college’s phone number, email button intents application to choose any of the mailing apps installed on the device to send an email and View Map button intents the application to show the location of the college using Google Maps API.

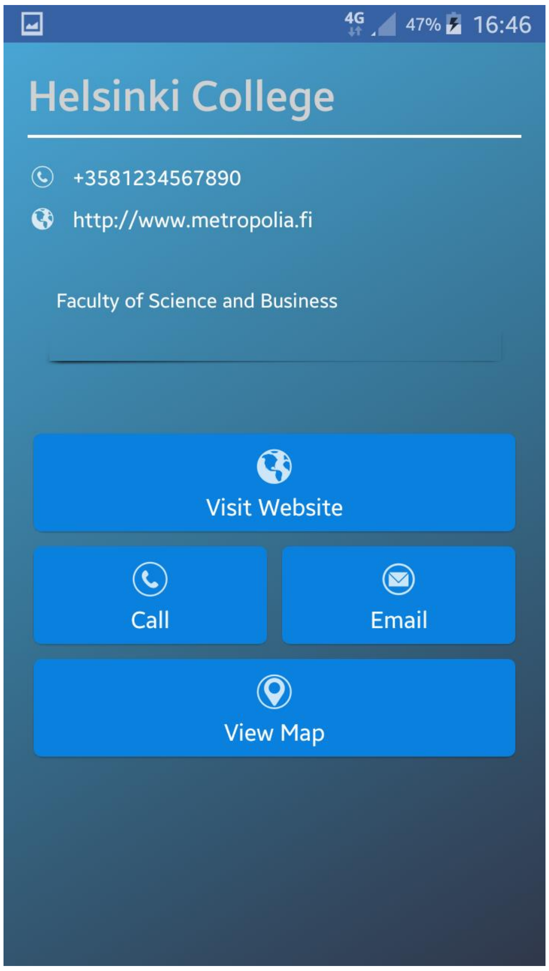
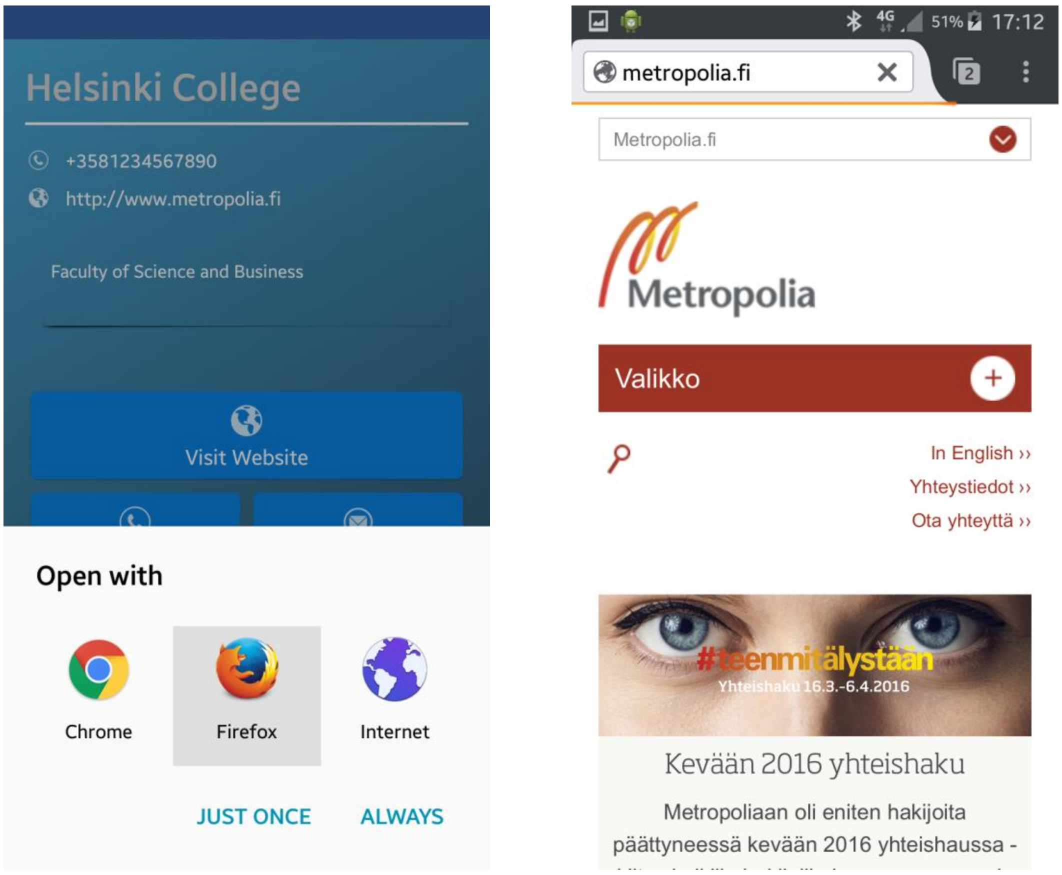


Figure 41: College Info Activity Screen Shot.

7.12 Visit Website Activity

Figure 42 shows how the website can be visited using the application. After Visit Website button is tapped. The application asks to select any of the web browser installed on the device. Selection of a web browser let application browse the fetched website URL.



Kevään 2016 yhteishaku

Metropoliaan oli eniten hakijoita päättyneessä kevään 2016 yhteishaussa -

Figure 42: Visit Website Activity Screen Shot.

7.13 Call Activity

Figure 43 shows the call activity of the application. After tapping Call button, the application shows the phone number and intents the application to the dialer application installed on the device and calls to the respective phone number given.

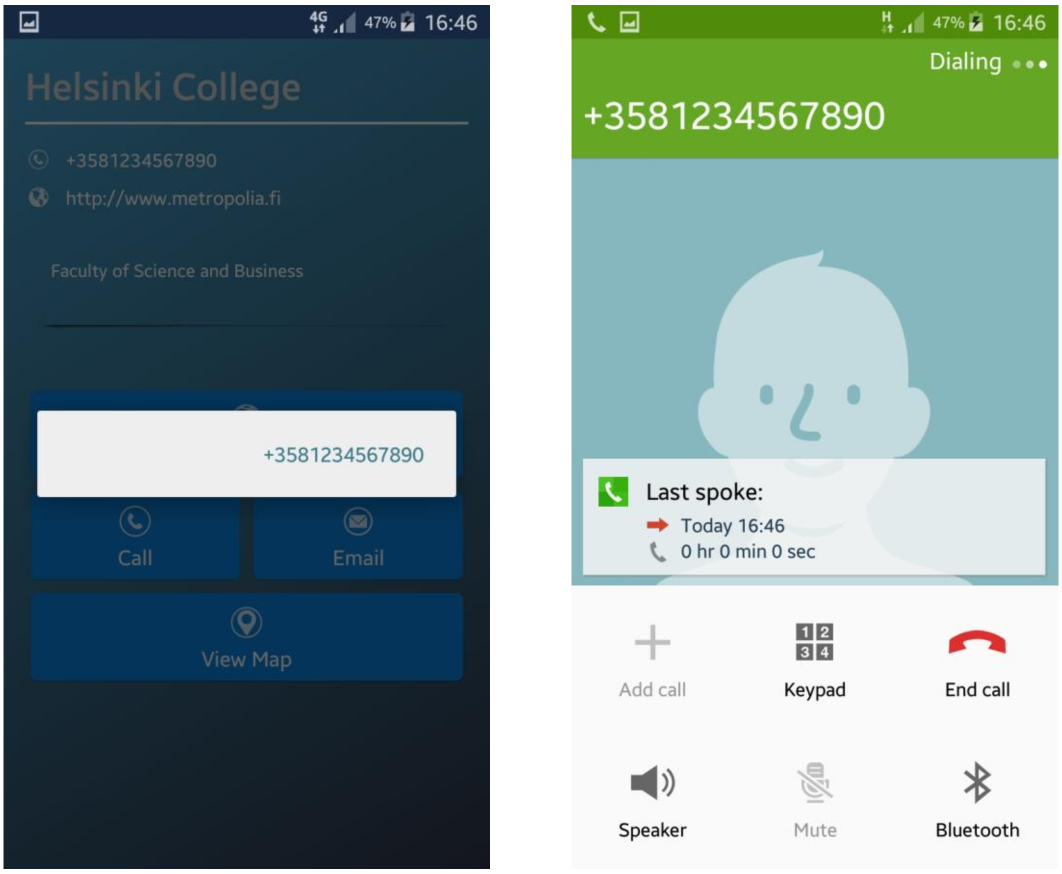
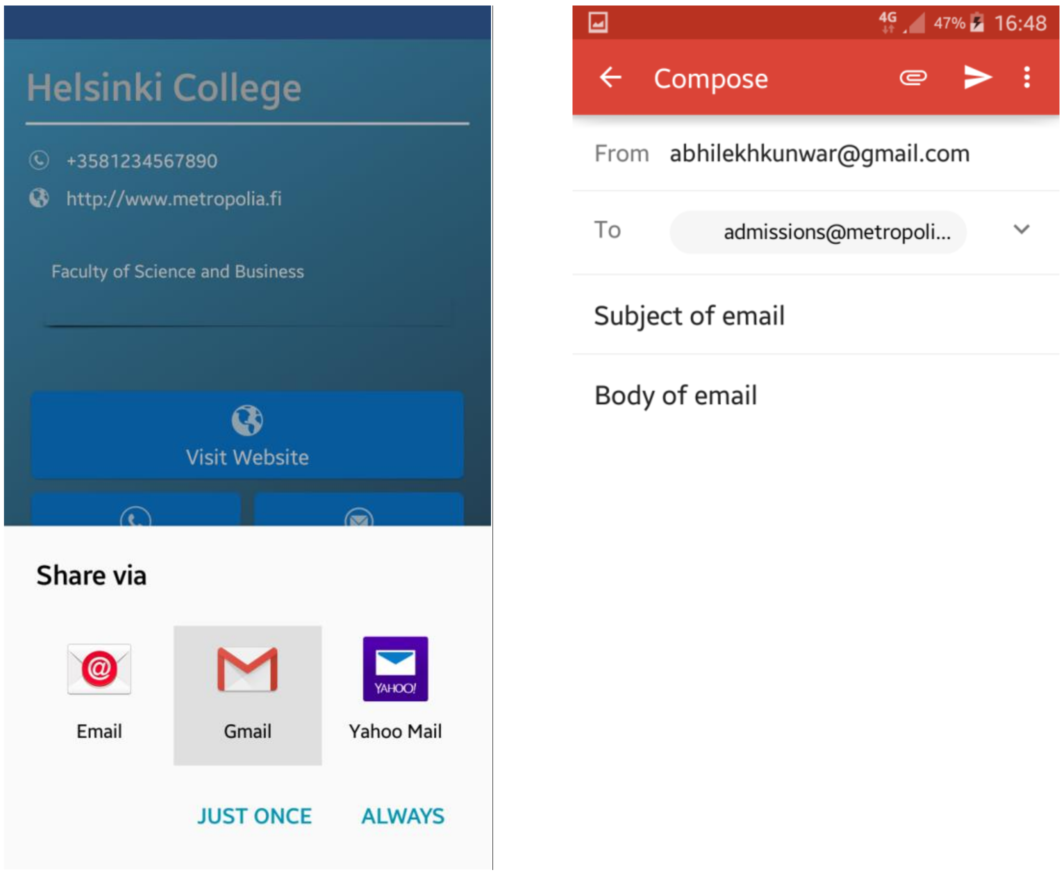


Figure 43: Call Activity Screen Shot.

7.14 Send Email Activity

Figure 44 shows how the email can be sent using the application. After Email button is tapped, the application asks to select any of the mailing application installed on the device. Selection of a mailing application, intents the application to the mailing application where the user can fill the respective fields and sends an email.


Helsinki College
+3581234567890
http://www.metropolia.fi
Faculty of Science and Business

From abhilekhkunwar@gmail.com
To admissions@metropoli...
Subject of email
Body of email

Figure 44: Send Email Activity Screen Shot.

7.15 View Map Activity

Figure 45 shows the view map activity of the application. After tapping View Map button, the application intents to the Google Maps activity. Google Maps API fetch the given latitude and longitude to mark the location to this Map Activity. The user can zoom in and out using the pinch method. The second figure shows the zoom in of the map.

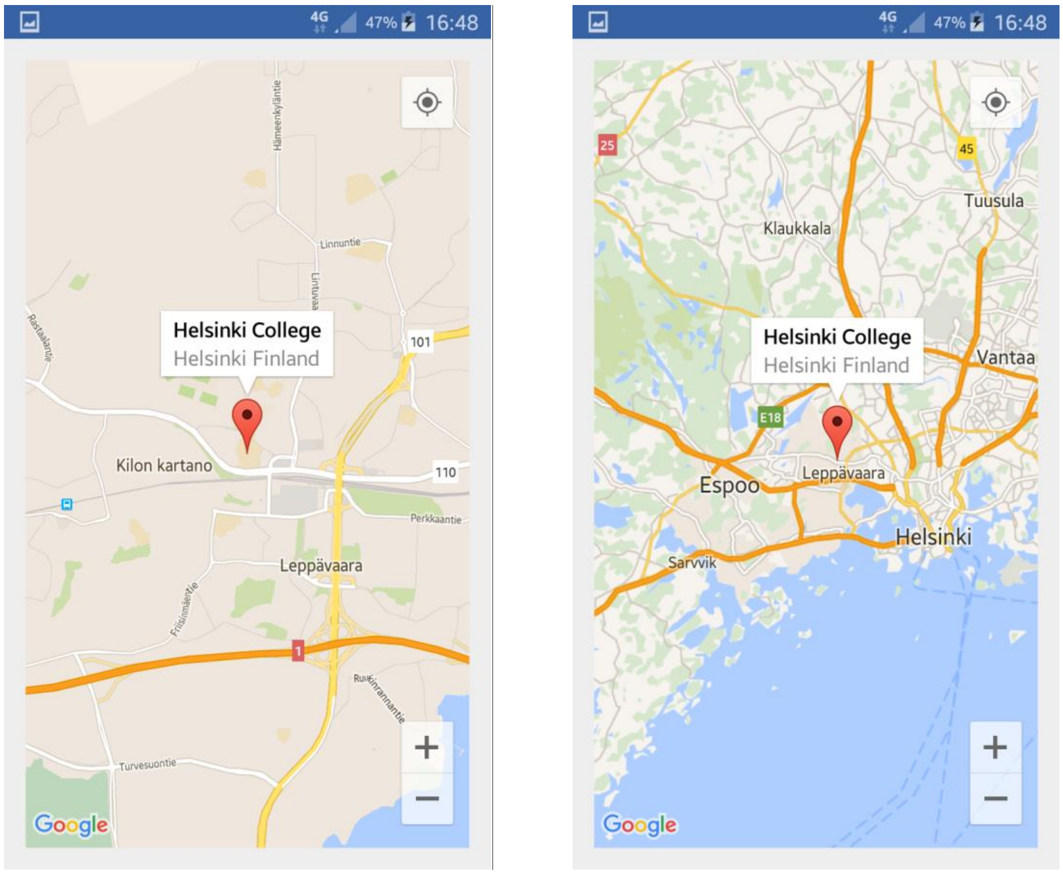
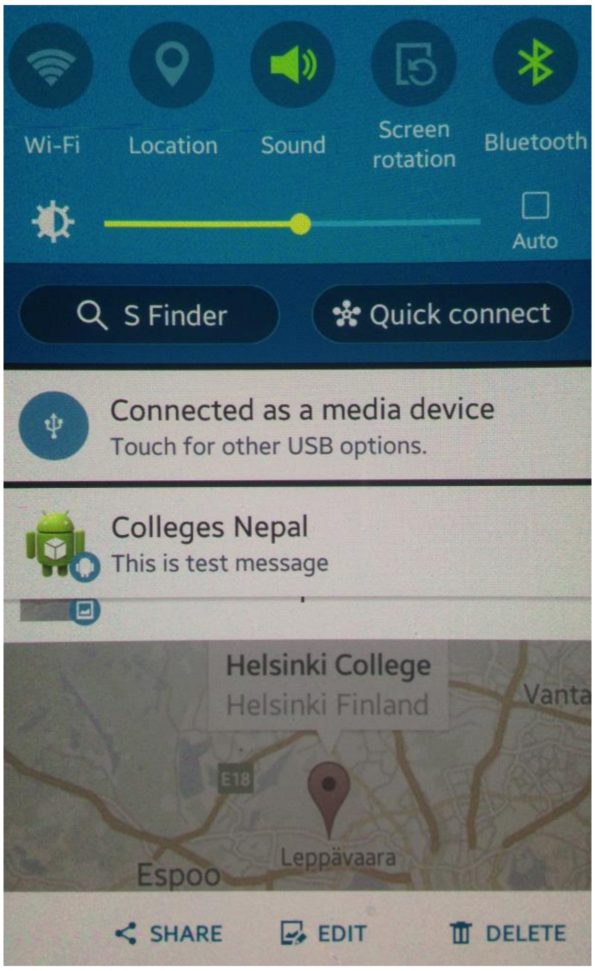


Figure 45: View Map Activity Screen Shot.

7.16 View Notification Activity

Figure 46 shows the push notification received by the application which is installed on the Android device.



Connected as a media device
Touch for other USB options.

Colleges Nepal
This is test message

Figure 46: View Notification Activity Screen Shot.

8 IMPLEMENTATION

The development of this application is divided into two parts, Java and PHP, for the server. Implementation is done using Java Programming language for the functions of the application. Use of MySQL database is used to store all the data that is included the application.

The application consists of three main parts, the Splash Screen, which starts the application, the Main Activity and Other Activities which are intent from one to another. Integration of Google Maps API Version 2 and Push Notification is in the application.

Following are the main code snippets of the application. Comments are re-moved:

8.1 Splash Screen

The start of the application shows Splash Screen, which takes 3000 Millisecond to load and will intent to the main activity which is login class. onCreate is a callback that creates the Activity at first. R.layout.splashscreen links the layout, that is displayed in the UI from the R package.

```
public class SplashScreen extends Activity {

  ProgressBar bar;
  long Delay = 3000;

  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    requestWindowFeature(Window.FEATURE_NO_TITLE);

    setContentView(R.layout.splashscreen);

    Timer RunSplash = new Timer();

    TimerTask ShowSplash = new TimerTask() {
      @Override
      public void run() {

        finish();
        Intent myIntent = new Intent(SplashScreen.this,
            Login.class);
        startActivity(myIntent);
```

```
            }
        };

        RunSplash.schedule(ShowSplash, Delay);
    }
}
```

Code Snippet 1: Splash Screen.

8.2 Login

The login class is joined with the another class AppConfig with the URL, which fetch, read and write data using PHP to and from database. The code is integrated with the admin in email or username and its password. So, if admin inputs admin’s credentials which is made for admin, the application intent to AdminPanel class otherwise to the UniversityList class.

```
JSONParser jsonParser = new JSONParser();

private static final String LOGIN_URL =AppConfig.LOGIN_URL;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        UserModel userModel = SharedPreferenceManager.getSharedInstance().getUser-
    ModelFromPreferences();
        if(userModel!=null){

            if(userModel.userName.equalsIgnoreCase("admin")){
                Intent intent = new Intent(Login.this, AdminPanel.class);
                intent.putExtra(UserModel.class.getSimpleName(), userModel);
                startActivity(intent);
                finishAffinity();
            }else {

                Intent intent = new Intent(Login.this, UniversityList.class);
                intent.putExtra(UserModel.class.getSimpleName(), userModel);
                startActivity(intent);
                finishAffinity();
            }
        }
        setContentView(R.layout.login);

        user = (EditText) findViewById(R.id.username);
        pass = (EditText) findViewById(R.id.password);
        progressBar = (ProgressBar) findViewById(R.id.loading);
        progressBar.setVisibility(ProgressBar.INVISIBLE);
        mSubmit = (Button) findViewById(R.id.login);
        mRegister = (Button) findViewById(R.id.register);

        mSubmit.setOnClickListener(this);
        mRegister.setOnClickListener(this);

    }
```

Code Snippet 2: Login.

```
mSubmit = (Button) findViewById(R.id.login);

mRegister = (Button) findViewById(R.id.register);


mSubmit.setOnClickListener(this);

mRegister.setOnClickListener(this);
```

Code Snippet 3: Login buttons.

The Login and the Register buttons are assigned, where the Login button intents to the admin panel if Admin’s credentials are inputted and the Registration button intents to the registration activity for new user. findViewById links the layout using layout id assigned in the R package.

```
Log.d("request!", "starting");
                JSONObject json = jsonParser.makeHttpRequest(LOGIN_URL, "POST",params);
```

Code Snippet 4: Http request and POST method.

The application makes http request and using the Get method from the login url using JSON Parser that connects the application using PHP to the MySQL database.

8.3 Registration

The registration uses HTTP request and post method to write data to database that links by JSONParser. After the user inputs, the registration credential toast message is seen in UI as user created and intents to University List activity else return toast error message.

```
params.add(new BasicNameValuePair("username", username));
                params.add(new BasicNameValuePair("password", password));

                Log.d("request!", "starting");

                JSONObject json = jsonParser.makeHttpRequest(REGITER_URL,"POST",
                params);

                        Log.d("Registering attempt", json.toString());

                        success = json.getInt(TAG_SUCCESS);
                        if (success == 1) {
                        Log.d("User Created!", json.toString());

                        UserModel userModel = getUserModelPreferences(username);
                        if(userModel!=null){
    SharedPreferenceManager.getSharedInstance().saveUserModel(userModel);

                                }
                        startActivity(newIntent(Register.this,UniversityList.class));
                                return json.getString(TAG_MESSAGE);
                        } else {
                        Log.d("Registering Failure!", json.getString(TAG_MESSAGE));
                                return json.getString(TAG_MESSAGE);
                        }
                } catch (JSONException e) {
                        e.printStackTrace();
                }
            return null;
            }
```

Code Snippet 5: Registration.

8.4 Admin Panel

Admin Panel consists of five buttons; each button is assigned with setOnClick-Listener method. These method intents every buttons to its specific Activity or their own class as mentioned and task. The layout view of every button is linked using setContentView.

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_admin_panel);
    mToolbar = (Toolbar) findViewById(R.id.admin_toolbar);
```

```
setSupportActionBar(mToolbar);
getSupportActionBar().setDisplayShowHomeEnabled(true);
getSupportActionBar().setTitle("Admin Panel");
add_Colleges = (Button) findViewById(R.id.add_butt);

btn_logout = (Button) findViewById(R.id.btn_logout);

add_Colleges.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
      startActivity(new Intent(AdminPanel.this,AddColleges.class));
   }
});

btn_logout.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
      SharedPreferenceManager.getSharedInstance().clearAllPreferences();
      startLoginActivity();
   }
});
push_button = (Button) findViewById(R.id.btnSendNotification);

push_button.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
      startActivity(new Intent(AdminPanel.this, PushNotificationActivity.class));
   }
});

m_view_but = (Button) findViewById(R.id.view_but);
m_view_but.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
```

Code Snippet 6: Admin Panel.

8.5 Add University

When admin taps Add University button, the application intents to add aniversity activity. JSONObject pass the data what is inputted using ArrayList. HTTP request and POST method is used to write data to the database using JSONParser. The aniversity name must be inputted in small letters. By clicking btn_AddUniversity will POST the university name to the database.

```
uniName =getIntent().getExtras().getString("Heading").toLowerCase();

    etUni_Name = (EditText) findViewById(R.id.add_Universityname1);
    btn_AddUniversity = (Button) findViewById(R.id.btn_add_newUniversity);
```

```
        if(uniName.equals("test")){
            etUni_Name.setText("");
        }else {
            etUni_Name.setText(uniName);
        }
        btn_AddUniversity.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if(etUni_Name.getText().toString().contains("University")){
                    new AddUniversityDetails().execute();
                }else {
                    Toast.makeText(AddUniversity.this, "Please make sure to put space before Uni-
versity string and in small letter", Toast.LENGTH_SHORT).show();
                }

            }
        });

    }

class AddUniversityDetails extends AsyncTask<String, String, String> {


        @Override
        protected void onPreExecute() {
            super.onPreExecute();

            pDialog = new ProgressDialog(AddUniversity.this);
            pDialog.setMessage("Creating/Renaming University..");
            pDialog.setIndeterminate(false);
            pDialog.setCancelable(true);
            pDialog.show();
        }

        @Override
        protected String doInBackground(String... args) {
            // TODO Auto-generated method stub


            int success;
            String UniversityName = etUni_Name.getText().toString();
            if (UniversityName.contains("University")) {
                try {
                    JSONObject json= null;
                    // Building Parameters
                    List<NameValuePair> params = new ArrayList<NameValuePair>();

                    Log.e("testing",UniversityName);
                    params.add(new BasicNameValuePair("u_name", UniversityName));
                    Log.d("request!", "starting");

                    if(getIntent().getExtras().getInt("check")==1){
                        List<NameValuePair> paramss = new ArrayList<NameValuePair>();
                        List<NameValuePair> paramsss = new ArrayList<NameValuePair>();
                        //String id = "1";
                        Log.e("testing",UniversityName);
                        paramss.add(new BasicNameValuePair("uni_name", uniName));
                        paramss.add(new BasicNameValuePair("u_name",UniversityName));
     json=jsonParser.makeHttpRequest(AppConfig.UDATE_UNIVESITY,"POST",paramss);
                        paramsss.add(new BasicNameValuePair("u_name",UniversityName));
                        jsonParser.makeHttpRequest(AppConfig.UPDATE_NEW_COLLEG-
ESS,"POST",paramsss);
                    }else {
                        json = jsonParser.makeHttpRequest(AppConfig.ADD_UNIVERSITY, "POST",
params);
                    }

                    if(json.getInt(TAG_SUCCESS)==1){
                        return json.getString(TAG_MESSAGE);
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                }
            } else {
                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        Toast.makeText(getApplicationContext(), "Field must have University string
to add an University", Toast.LENGTH_SHORT).show();
                    }
                });
            }
```

Code Snippet 7: Add University.

8.6 Add College

Add College button intents the application to the add college activity where admin can add College according to the University. The phone number should match the validation given in the code. getText is an editable box where admin can input the college details and the button is clicked. Add Button is assigned which will post the data using http request.

```
m_add_but = (Button) findViewById(R.id.add_but);
    m_add_but.setOnClickListener(this);

  }
  public void onClick(View v) {
    String phonepattern = "^[+]?[0-9]{10,15}$";
    String post_c_phone = c_phone.getText().toString();
    if (!post_c_phone.matches(phonepattern)) {


        if (!post_c_phone.matches(phonepattern)) {
            c_phone.setText("");
            Toast.makeText(getApplicationContext(),
                    "please enter valid phone num eg: min 10 is required",
Toast.LENGTH_SHORT).show();

        }
    }else  if(u_name.getText().toString().equals("  ")||c_name.getText().toString().equals("
")||c_address.getText().toString().equals("          ")||c_phone.getText().toString().equals("
")||c_email.getText().toString().equals("")||c_url.getText().toString().equals("")||u_name.g
etText().toString().equals("")||c_longitude.getText().equals("")||c_lati-
tude.getText().equals("")||description.getText().equals("")){
        Toast.makeText(AddColleges.this,    "Please    fill    all    the    field",
Toast.LENGTH_SHORT).show();

    } else {

        new PostCollege().execute();
    }
  }
```

Code Snippet 8: Add College.

8.7 University List

After the user's login or admin's View College button tap the application intents to the university list activity. JSONArry call backs the JSONObject to fetch university list from the database. In this code snippet, when the admin, tap one of the University name it will intents application to the college list activity.

```
public void jsonCallback(String url, JSONObject json, AjaxStatus status) {
  if (json != null) {
    try {
      Log.e("testing", "testing");
      contacts = json.getJSONArray(TAG_CONTACTS);
      int len = contacts.length();
      for (int i = 0; i < len; i++) {
        JSONObject c = contacts.getJSONObject(i);

        name = c.getString(TAG_NAME);

        if(name.contains("University")){

          CollegeDetails cc =new CollegeDetails();
          String str =name;
          String finalStr= str.toUpperCase();

          cc.setName(finalStr);
          contactlist.add(cc);
        }
        Log.e("testing", "testing" + name);
        Lv.setAdapter(new CollegeListBaseAdapter(this,contactlist));

        Lv.setOnItemClickListener(new AdapterView.OnItemClickListener() {
          @Override
          public void onItemClick(AdapterView<?> parent, View view, final int position, long
id) {
            Object o = Lv.getItemAtPosition(position);
            final CollegeDetails obj_itemDetails = (CollegeDetails)o;
            Toast.makeText(UniversityList.this, "You have chosen : " + " " +
obj_itemDetails.getName(), Toast.LENGTH_LONG).show();

            CollegeAsync aa = new CollegeAsync(UniversityList.this);
            aa.execute(AppConfig.COLLEGE_INFO, obj_itemDetails.getName());


          }
        });
```

Code Snippet 9: University List.

8.8 College List

When user tap on the university name, the application intent to the college list activity. ArrayList implements any data that is fetched to the order list.

```
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);

   setContentView(R.layout.University_kathmandu);

   final String heading = getIntent().getStringExtra("heading");

   mToolbar = (Toolbar) findViewById(R.id.toolUniversityName);
   setSupportActionBar(mToolbar);
   getSupportActionBar().setDisplayHomeAsUpEnabled(true);
   getSupportActionBar().setTitle(heading);

   kuCollegeList = new ArrayList<HashMap<String, String>>();
   lv = (ListView) findViewById(R.id.listt);
```

Code Snippet 10: College List.

8.9 College Information

For fetching college information JSONObject is made which posts the data that is fetched form data base to the application’s UI.

```
JSONObject jsonObject = new JSONObject(name1);
        JSONArray jsonArray =jsonObject.getJSONArray("posts");

        for(int i = 0; i<s;i++) {
           JSONObject jsonObject1 = jsonArray.getJSONObject(i);
           Description = jsonObject1.getString(TAG_DESCRIPTION);
           Web = jsonObject1.getString(TAG_WEB);
           Phone = jsonObject1.getString(TAG_PHONE);
           Address = jsonObject1.getString(TAG_ADDRES);
           NameC = jsonObject1.getString(TAG_NAME);
           latitude = jsonObject1.getString(TAG_LATITUDE);
           longitude = jsonObject1.getString(TAG_LONGITUDE);
           Email = jsonObject1.getString(TAG_EMAIL);
        }
```

Code Snippet 11: College Information.

8.10 Update or Delete University

In the university list activity, setOnItemLongCLickListener method works only when there is a long tap and hold on the screen. When the admin taps and holds on the university which should be updated or deleted. The positive button setPositive-Button function to delete the university name linked with JsonParser by sending HTTP request POST method. setNegativeButton function to update the university name.

```
    Lv.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {
            @Override
            public boolean onItemLongClick(AdapterView<?> parent, View view, int posi-
tion, long idd) {
              Object o = Lv.getItemAtPosition(position);
              final CollegeDetails obj_itemDetails = (CollegeDetails)o;
              UserModel userModel = SharedPreferenceManager.getSharedIn-
stance().getUserModelFromPreferences();
              if (userModel != null) {
                if (userModel.userName.equalsIgnoreCase("admin")) {
android.app.AlertDialog.Builder adb = new android.app.AlertDialog.Builder(Univer-
sityList.this);

                  adb.setTitle("Do you want to delete University!!");
```

```
                    adb.setIcon(android.R.drawable.ic_dialog_alert);
                    adb.setCancelable(false);

                    adb.setPositiveButton("Delete", new DialogInterface.OnClickListener() {
                      public void onClick(DialogInterface dialog, int which) {

                    AsyncTask<String,String, String> aa = new AsyncTask<String, String,
String>() {
                           @Override
                           protected String doInBackground(String... params) {
                             try {
                               List<NameValuePair> list = new ArrayList<NameValuePair>();
                        list.add(new BasicNameValuePair("u_name",obj_itemDetails.get-
Name()));
                               Log.e("string",name);
JSONObject jsontest = jsonParser.makeHttpRequest(AppConfig.DELETE_UNIVERSITY,
"POST", list);
                               Log.e("String", "" + jsontest);

                               int success = jsontest.getInt("success");
                               if (success == 1) {
                                 return jsontest.getString("message");
                               }

                               return jsontest.getString("message");
                             } catch (Exception e) {
                               e.printStackTrace();

                             }
                             return null;
                           }

                           @Override
                           protected void onPostExecute(String s) {
                             super.onPostExecute(s);
                             Toast.makeText(UniversityList.this, s,
Toast.LENGTH_SHORT).show();
                             startActivity(new Intent(UniversityList.this,AdminPanel.class));
                           }
                         };
                         aa.execute();
                       }
                    });
                    adb.setNegativeButton("Update", new DialogInterface.OnClickListener() {
                      @Override
                      public void onClick(DialogInterface dialog, int which) {
                        Intent i =new Intent(UniversityList.this, AddUniversity.class);
                        Bundle bn = new Bundle();
                        bn.putInt("check",1);
                        bn.putString("Heading",obj_itemDetails.getName());
                        i.putExtras(bn);
                        startActivity(i);
                    }
                  });
                  adb.setCancelable(true);
                  adb.show();
                }

              }
              return true;
            }
          });
        }
      } catch (JSONException e) {

      }
    }
  }
```

Code Snippet 12: Update or Delete University.

8.11 Update or Delete College

Like in update and delete university. On college list activity, when admin taps and holds on the college name which should be updated or deleted, setOnLongClick-Listener method will shows two buttons Update and Delete. Taping Update button intents the application to update college activity whereas the admin can replace name as like before with the new name, which works as HTTP request and POST method. Tapping Delete option will send HTTP request and POST method to delete college name and all its details from the database.

```
lv.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {
          @Override
          public boolean onItemLongClick(AdapterView<?> parent, View view, int position,
long idd) {

            UserModel userModel = SharedPreferenceManager.getSharedIn-
stance().getUserModelFromPreferences();
            if (userModel != null) {
              if (userModel.userName.equalsIgnoreCase("admin")) {

              android.app.AlertDialog.Builder adb = new android.app.AlertDia-
log.Builder(FetchUniversity.this);

                adb.setTitle("Do you want to delete/update College!!");


                adb.setIcon(android.R.drawable.ic_dialog_alert);
```

```
        adb.setCancelable(false);

        adb.setPositiveButton("Delete", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) {


                AsyncTask<String,String, String> aa = new AsyncTask<String, String,
String>() {
                    @Override
                    protected String doInBackground(String... params) {
                        try {
                            List<NameValuePair> list = new ArrayList<NameValuePair>();
                            list.add(new BasicNameValuePair("u_name",heading));
                            list.add(new BasicNameValuePair("c_id", id));
                            Log.e("string",nameOnly+id);

                            JSONObject jsontest = jsonParser.makeHttpRe-
quest("http://www.cc.puv.fi/~e1100612/Collegesnepal/delete_College.php", "POST", list);
                            Log.e("String", "" + jsontest);

                            int success = jsontest.getInt("success");
                            if (success == 1) {
                                return jsontest.getString("message");
                            }

                            return jsontest.getString("message");
                        } catch (Exception e) {
                            e.printStackTrace();

                        }
                        return null;
                    }

                    @Override
                    protected void onPostExecute(String s) {
                        super.onPostExecute(s);
                        Toast.makeText(FetchUniversity.this, s,
Toast.LENGTH_SHORT).show();
                        new CollegeAsync(FetchUniversity.this).execute(AppConfig.COL-
LEGE_INFO, heading);
                    }
                };
                aa.execute();
            }
        });
        adb.setNegativeButton("Update", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
            Intent i =new Intent(FetchUniversity.this, UpdateColleges.class);
                Bundle bn = new Bundle();
                bn.putString(TAG_NAME,c_name);
                bn.putString(TAG_PHONE,phone);
                bn.putString(TAG_ADDRES,address);
                bn.putString(TAG_EMAIL,email);
```

```
                                    bn.putString(TAG_DESCRIPTION,description);
                                    bn.putString(TAG_LOGITUDE,logitude);
                                    bn.putString(TAG_LATITUDE,latitude);
                                    bn.putString(TAG_ID,id);
                                    bn.putString(TAG_URL,url);
                                    bn.putString("Heading",heading);
                                    i.putExtras(bn);
                                    startActivity(i);

                                }
                            });
                            adb.setCancelable(true);
                            adb.show();
                        }

                    }
                    return true;
                }
            });
        }
    } catch (JSONException e) {
        e.printStackTrace();
    }
}
```

Code Snippet 13: Update or Delete College.

8.12 Visit Website

When the user taps the View Website button, the application intents it to select any web browser to visit the website. The ACTION_SENDTO method is an SDK content type that intents application to website browser. Then the fetched URL address is inputted in the web browser tab.

```
            String url =  Web;
             Intent i = new Intent(Intent.ACTION_VIEW);
             i.setData(Uri.parse(url));
             startActivity(i);
break;
```

Code Snippet 14: Visit Website.

8.13 Call

On this implementation, onClick Method is used to tap the Call Button and that makes an alertDialog call. It shows alert box that is with the phone number. Permission is made to use dialer application and this intent the application to dialer to call the given phone number.

```
public void onClick(View arg0) {
    // TODO Auto-generated method stub
    switch (arg0.getId()) {
      case R.id.makeCall:
        AlertDialog.Builder alertDialog = new AlertDialog.Builder(FetchCollegeInfo.this);
        alertDialog.setPositiveButton(Phone, new DialogInterface.OnClickListener() {

          @Override
          public void onClick(DialogInterface arg0, int arg1) {
            // TODO Auto-generated method stub

          }
        });
        alertDialog.show();
        Intent phoneCallIntent = new Intent(Intent.ACTION_CALL);

        phoneCallIntent.setData(Uri.parse("tel:" + Phone));
        startActivity(phoneCallIntent);
```

Code Snippet 15: Call.

```
private class PhoneCallListener extends PhoneStateListener {

    String TAG = "LOGGING PHONE CALL";
   private boolean phoneCalling = false;

   @Override
    public void onCallStateChanged(int state, String incomingNumber) {

   if (TelephonyManager.CALL_STATE_RINGING == state
   Log.i(TAG, "RINGING, number: " + incomingNumber);
      }
      if (TelephonyManager.CALL_STATE_OFFHOOK == state) {
        Log.i(TAG, "OFFHOOK");
            phoneCalling = true;

      }
```

Code Snippet 16: Call intent and dialing.

8.14 Send Mail

When the user taps Email Button, the application asks to choose a mailing app that was already installed in the device and intent it to mailing application. ACTION_SENDTO is an SDK content type that intents application to mailing app. The fetched phone number is inputted to the (To Receiver: text box).

```
Intent intent = new Intent(Intent.ACTION_SENDTO);
intent.setType("text/plain");
intent.putExtra(Intent.EXTRA_SUBJECT, "Subject of email");
intent.putExtra(Intent.EXTRA_TEXT, "Body of email");
intent.setData(Uri.parse("mailto:"+Email));
intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
startActivity(intent);
break;
```

Code Snippet 17: Send email.

8.15 View Maps

For viewing maps, maps activity is extended to fragment activity. Above codes are provided by Google Console. LatLng classes is used to define latitude and longitude of this activity. The marker is made using addMarker method which fetches college name and address. The marker names are shown when the user tap to the marker. CameraUpdateFactory is used to make zoom in and out in the Maps.

```
public class MapsActivity extends FragmentActivity
{

   private GoogleMap mMap;
   double latitude,longitude;
   LatLng College ;
   String Namee,Addresss;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_maps);
      ParcelableClass mdata = (ParcelableClass) getIntent()
            .getParcelableExtra(FetchCollegeInfo.PAR_KEY);
      Toast.makeText(getApplicationContext(), mdata.getLatitude() + "+" + mdata.getLongi-
tude()+"+"+mdata.getName()+"+"+mdata.getAddress(), Toast.LENGTH_LONG).show();
      Namee=mdata.getName();
      Addresss=mdata.getAddress();
      latitude = Double.parseDouble(String.valueOf(mdata.getLatitude()));
            longitude = Double.parseDouble(String.valueOf(mdata.getLongitude()));
```

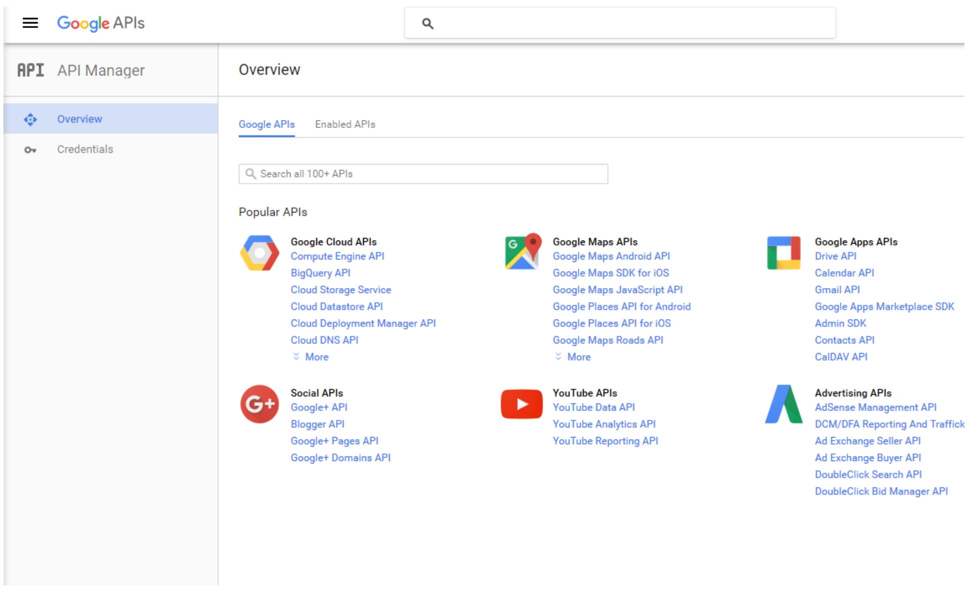
```
    College = new LatLng(latitude, longitude);
     setUpMapIfNeeded();

     if (mMap != null) {
        mMap.setMyLocationEnabled(true);
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(College, 13));
     }

  }

  @Override
  protected void onResume() {
     super.onResume();
     setUpMapIfNeeded();
  }

  private void setUpMapIfNeeded() {
     // Do a null check to confirm that we have not already instantiated the map.
     if (mMap == null) {
        // Try to obtain the map from the SupportMapFragment.
        mMap = ((SupportMapFragment) getSupportFragmentManager().findFrag-
mentById(R.id.map))
              .getMap();

        // Check if we were successful in obtaining the map.
        if (mMap != null) {
           setUpMap();
        }
     }
  }
  private void setUpMap() {
     MarkerColleges=mMap.addMarker(newMarkerOptions()
           .position(College)
           .title(Namee)
           .snippet(Addresss));
  }
}
```

Code Snippet 18: View Maps.

8.15.1 Google Maps API Keys

Here are some steps to get Google Maps API Keys from Google’s Console website (https://console.developers.google.com/apis/library)

- Register and login via Google account
- Go to overview and Google Maps Android API is clicked



Overview

Popular APIs

Google Cloud APIs
Compute Engine API
BigQuery API
Cloud Storage Service
Cloud Datastore API
Cloud Deployment Manager API
Cloud DNS API
More

Google Maps APIs
Google Maps Android API
Google Maps SDK for iOS
Google Maps JavaScript API
Google Places API for Android
Google Places API for iOS
Google Maps Roads API
More

Google Apps APIs
Drive API
Calendar API
Gmail API
Google Apps Marketplace SDK
Admin SDK
Contacts API
CalDAV API

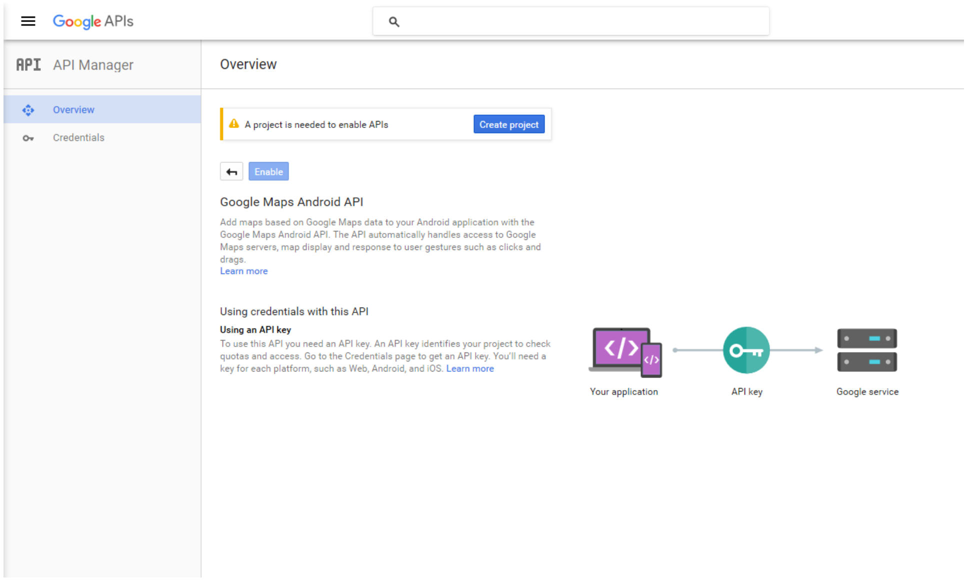
Social APIs
Google+ API
Blogger API
Google+ Pages API
Google+ Domains API

YouTube APIs
YouTube Data API
YouTube Analytics API
YouTube Reporting API

Advertising APIs
AdSense Management API
DCM/DFA Reporting And Traffick
Ad Exchange Seller API
Ad Exchange Buyer API
DoubleClick Search API
DoubleClick Bid Manager API

Figure 47: Google Maps API Key Guide Screen Shot-1.

- Create a new project



Overview

A project is needed to enable APIs

Google Maps Android API

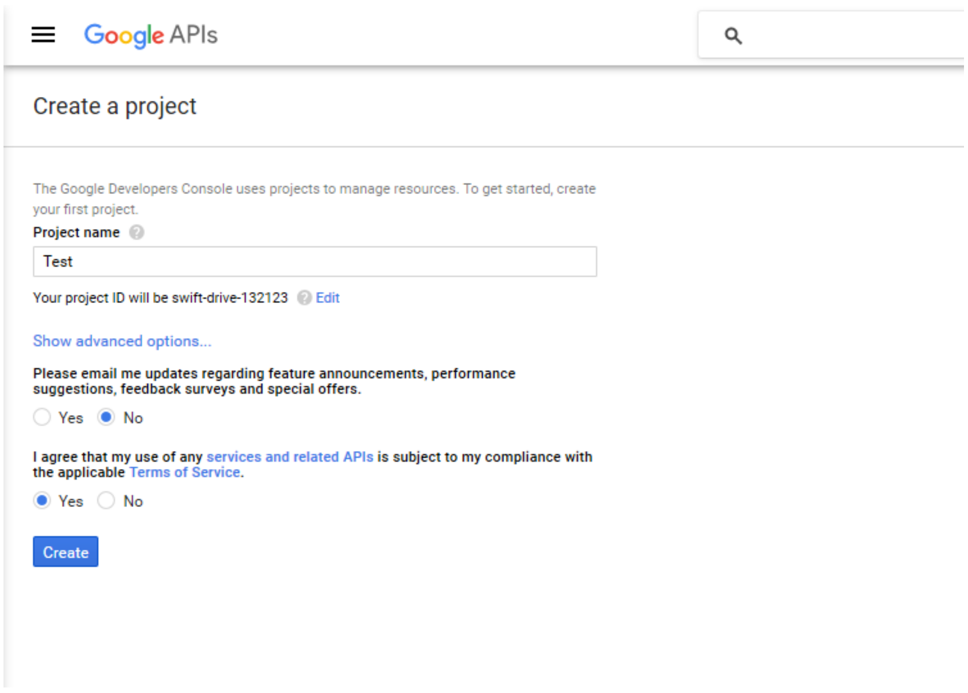
Add maps based on Google Maps data to your Android application with the Google Maps Android API. The API automatically handles access to Google Maps servers, map display and response to user gestures such as clicks and drags.
Learn more

Using credentials with this API

Using an API key

To use this API you need an API key. An API key identifies your project to check quotas and access. Go to the Credentials page to get an API key. You'll need a key for each platform, such as Web, Android, and iOS. Learn more

Figure 48: Google Maps API Key Guide Screen Shot-2.



Create a project

The Google Developers Console uses projects to manage resources. To get started, create your first project.

Project name

Test

Your project ID will be swift-drive-132123 Edit

Show advanced options...

Please email me updates regarding feature announcements, performance suggestions, feedback surveys and special offers.

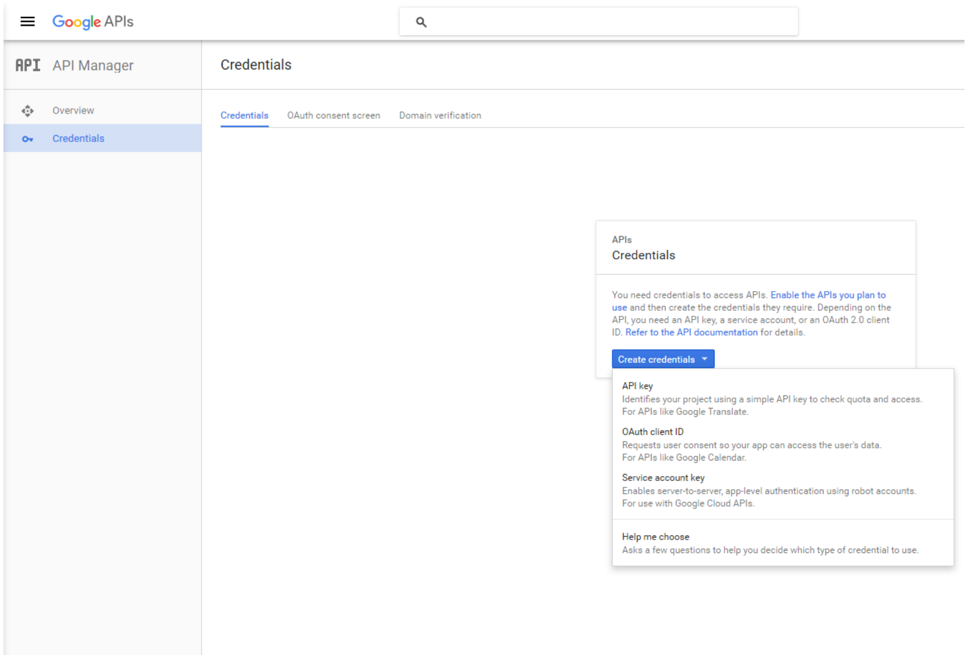
Yes [✓ No]

I agree that my use of any services and related APIs is subject to my compliance with the applicable Terms of Service.

[✓ Yes] No

Figure 49: Google Maps API Key Guide Screen Shot-3.

- Click on credential and make an API key for Android, name it and click OK



APIs

Credentials

You need credentials to access APIs. Enable the APIs you plan to use and then create the credentials they require. Depending on the API, you need an API key, a service account, or an OAuth 2.0 client ID. Refer to the API documentation for details.

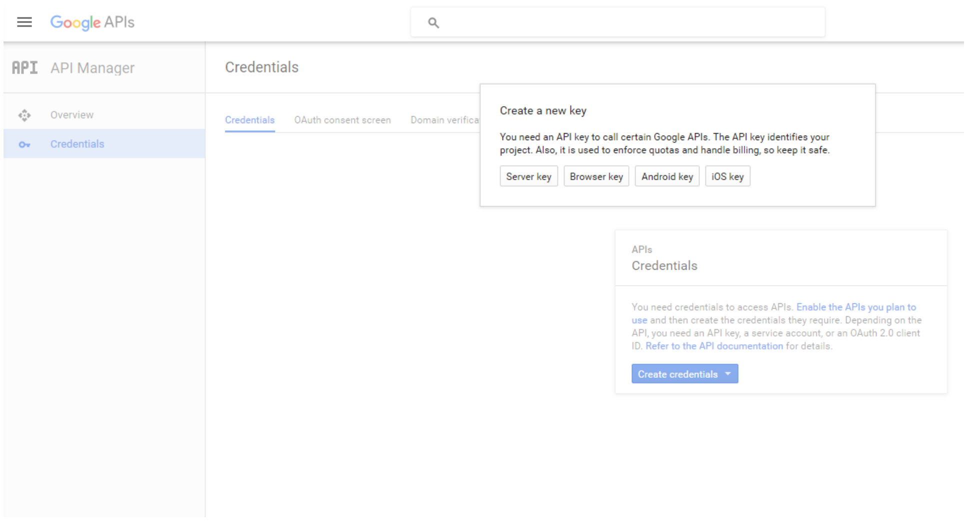
API key
Identifies your project using a simple API key to check quota and access.
For APIs like Google Translate.

OAuth client ID
Requests user consent so your app can access the user's data.
For APIs like Google Calendar.

Service account key
Enables server-to-server, app-level authentication using robot accounts.
For use with Google Cloud APIs.

Help me choose
Asks a few questions to help you decide which type of credential to use.

Figure 50: Google Maps API Key Guide Screen Shot-4.



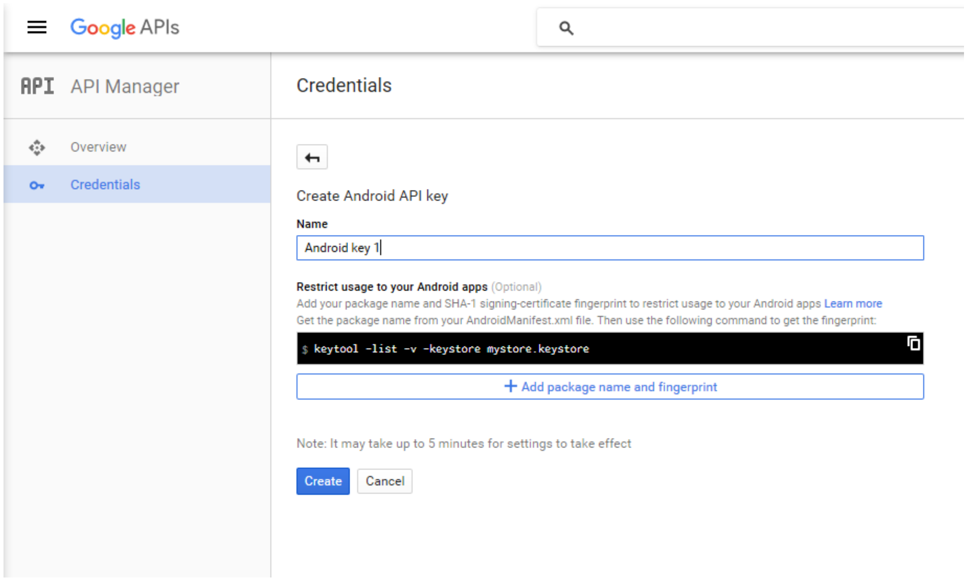
Create a new key

You need an API key to call certain Google APIs. The API key identifies your project. Also, it is used to enforce quotas and handle billing, so keep it safe.

APIs
Credentials

You need credentials to access APIs. Enable the APIs you plan to use and then create the credentials they require. Depending on the API, you need an API key, a service account, or an OAuth 2.0 client ID. Refer to the API documentation for details.

Figure 51: Google Maps API Key Guide Screen Shot-5.



Create Android API key

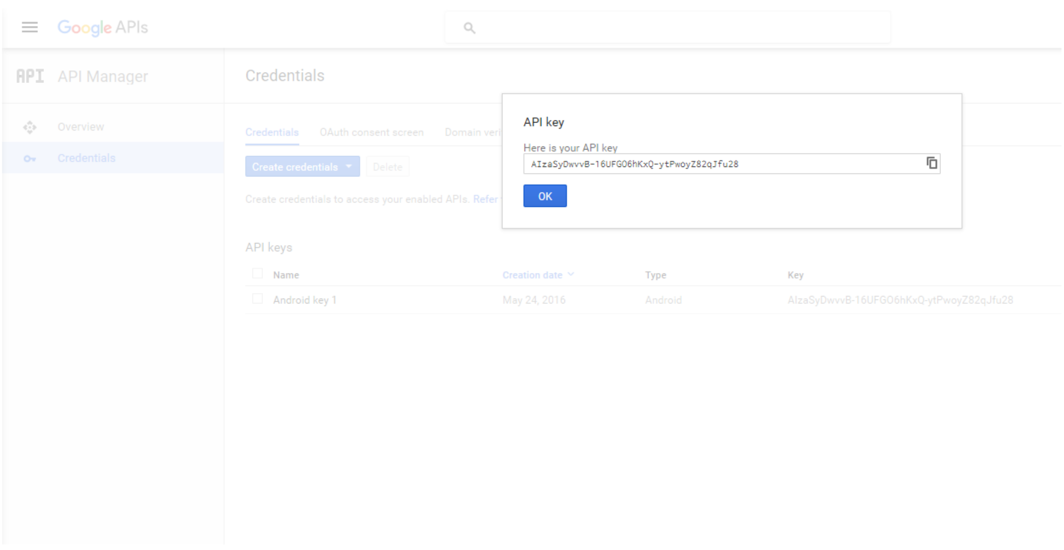
Name: Android key 1

Restrict usage to your Android apps (Optional)
Add your package name and SHA-1 signing-certificate fingerprint to restrict usage to your Android apps Learn more
Get the package name from your AndroidManifest.xml file. Then use the following command to get the fingerprint:

```
$ keytool -list -v -keystore mystore.keystore
```

Note: It may take up to 5 minutes for settings to take effect

Figure 52: Google Maps API Key Guide Screen Shot-6.



API key

Here is your API key

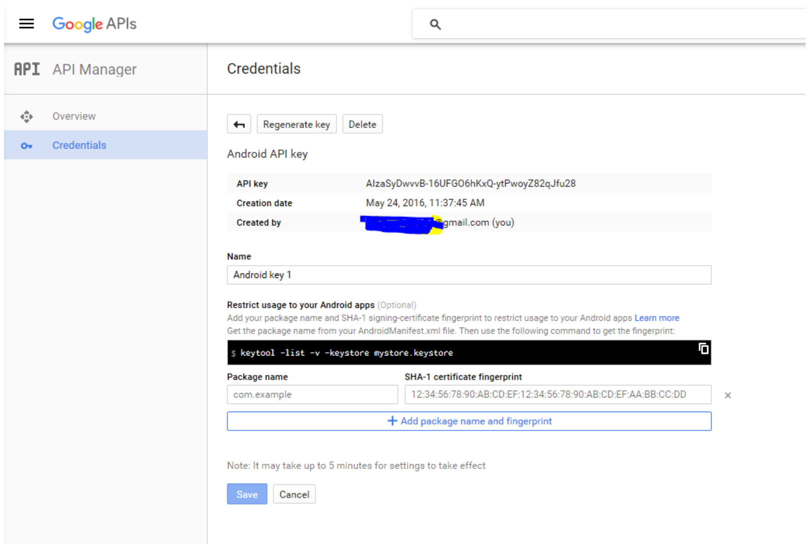
```
AIzaSyDwvvB-16UFGO6hKxQ-ytPwoyZ82qJfu28
```

API keys

Name	Creation date	Type	Key
Android key 1	May 24, 2016	Android	AIzaSyDwvvB-16UFGO6hKxQ-ytPwoyZ82qJfu28

Figure 53: Google Maps API Key Guide Screen Shot-7.

- The following command line can be used to create Certificate SHA-1 Finger Print in Windows OS if User's Android Project's file path is like this: *Windows Vista and Windows 7 min : C:\Users\your_user_name\.android*
  keytool -list -v -keystore "%USERPROFILE%\.android\debug.keystore" -alias androiddebugkey -storepass android -keypass android
- Add package name and SHA-1 certificate finger print and save



Android API key

API key	AlzaSyDwvvB-16UFGO6hKxQ-ytPwoyZ82qJfu28
Creation date	May 24, 2016, 11:37:45 AM
Created by	[redacted]gmail.com (you)

Name: Android key 1

Restrict usage to your Android apps (Optional)
Add your package name and SHA-1 signing-certificate fingerprint to restrict usage to your Android apps Learn more
Get the package name from your AndroidManifest.xml file. Then use the following command to get the fingerprint:

```
$ keytool -list -v -keystore mystore.keystore
```

Package name: com.example
SHA-1 certificate fingerprint: 12:34:56:78:90:AB:CD:EF:12:34:56:78:90:AB:CD:EF:AA:BB:CC:DD

Note: It may take up to 5 minutes for settings to take effect

Figure 54: Google Maps API Key Guide Screen Shot-8.

- The key can be implemented inside the google map xml file inside values folder.

```
<resources>
   <string name="google_maps_key_instructions" templateMergeStrategy="re-
place">
</string>

   <string name="google_maps_key" translatable="false" templateMergeStrat-
egy="preserve">
       AIzaSyCjPjRerGBsPrIZlUXdLt7O0AN94Cz6wlI
   </string>
</resources>
```

Code Snippet 19: Google Maps API Key integration.

8.16 Push Notification

When the Admin taps Send Notification Button, the application intent to Push Notification Activity. ParsePush class is used to send the Query using JSONObject. When the notification sent, toast message is seen on the UI. (http://parse.com/)

```
public class PushNotificationActivity extends Activity{
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.push_noti_layout);
      final EditText edittext = (EditText) findViewById(R.id.editTextSendPush);
      Button btnSend = (Button) findViewById(R.id.btnSendMessage);

      btnSend.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            ParseQuery<ParseInstallation> pushQuery = ParseInstallation.getQuery();
            // using parse data sdk
            String data1 = edittext.getText().toString();


            String data =  "{\n" +
                  "   \"data\": {\n" +
                  "      \"message\": \""+data1+"\",\n" +
                  "      \"title\": \"Colleges Nepal\"\n" +
                  "   }\n" +
                  "}";

            JSONObject jsondata = null;
            try {
                jsondata = new JSONObject(data);
            } catch (JSONException e) {
                e.printStackTrace();
            }
            ParsePush push = new ParsePush();
            push.setQuery(pushQuery);
            push.setData(jsondata);
            push.sendInBackground(new SendCallback() {
                @Override
                public void done(ParseException e) {
                    if (e == null) {
                        Toast.makeText(PushNotificationActivity.this, "Push Notification Send!!",
Toast.LENGTH_SHORT).show();
                    } else {
                        Toast.makeText(PushNotificationActivity.this, e.getLocalizedMessage(),
Toast.LENGTH_SHORT).show();
                    }
                }
            });

        }
    });

}
```

Code Snippet 20: Push Notification.

8.16.1 Generating Key Using Parse.com

With the use of parse.com and its key push notification are enables in this application. The project is setup and connect with parse.com. Here are some ways to use parse.com:

- Registration and login is made through parse.com website
- New application is created and give some name (for example: Testing)

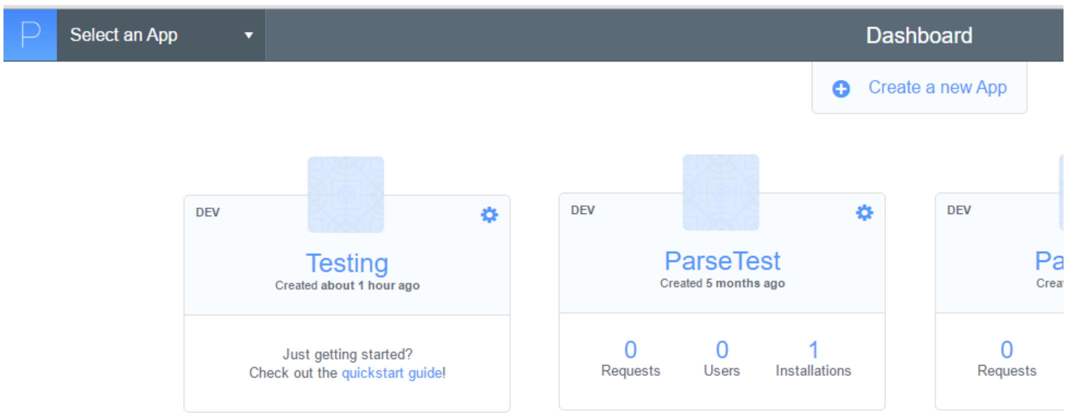
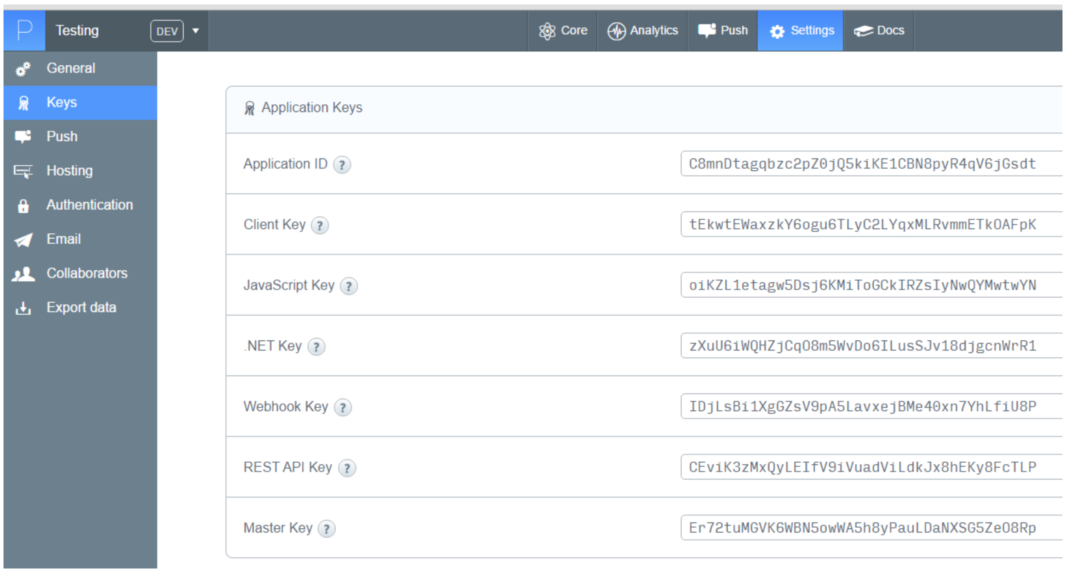


Figure 55: Parse.com Key Guide Screen Shot-1.

- Inside setting, keys can be found, Application ID and Client Key



Application Keys

Application ID	C8mnDtagqbzc2pZ0jQ5kiKE1CBN8pyR4qV6jGsdt
Client Key	tEkwtEWaxzkY6ogu6TLyC2LYqxMLRvmmETkOAFpK
JavaScript Key	oiKZL1etagw5Dsj6KMiToGCkIRZsIyNwQYMwtwYN
.NET Key	zXuU6iWQHZjCqO8m5WvDo6ILusSJv18djgcnWrR1
Webhook Key	IDjLsBi1XgGZsV9pA5LavxejBMe40xn7YhLfiU8P
REST API Key	CEviK3zMxQyLEIfV9iVuadViLdkJx8hEKy8FcTLP
Master Key	Er72tuMGVK6WBN5owWA5h8yPauLDaNXSG5ZeO8Rp

Figure 56: Parse.com Key Guide Screen Shot-2.

- These key is implemented inside Application project

public class AppDelegate extends Application implements Application.Activity-LifecycleCallbacks {

public static final String YOUR_APPLICATION_ID = "4hbcA4w3V4ZIUt-pJLC08ycUpnua1WJqE99OG8CvH";

```
    public static final String YOUR_CLIENT_KEY = "nK1zpZa2OuLfUNu4dCX5Vljjub-
DlGvNShJnIJRoz";


    @Override
    public void onCreate() {
        super.onCreate();
        instantiateManagers();
    }

    private void instantiateManagers() {
        ParseObject.registerSubclass(ArticleModel.class);
        Parse.initialize(this, YOUR_APPLICATION_ID, YOUR_CLIENT_KEY);
        SharedPreferenceManager.getSharedInstance().initiateSharedPrefer-
ences(getApplicationContext());
    }
```

Code Snippet 21: Parse.com Key Integration.

9 TESTING

The process of executing an application that aims to find the bugs is software testing. It is the process of validating and verifying a software, if the software meets the requirements, expectation of development or not. The Table 6 consists of most important test cases, their descriptions and improvements made:

Table 6: Testing.

#	**Testing Cases**	**Description and Problems**	**Improvements**
1	Using Android OS device version less than minimum 4.0 Ice Cream Sandwich	Application does not installed	This is Ok
2	Starting application	Application did not start	Using permission to use the internet from device
3	Splash Screen	Loading time takes longer than usual	Loading time is reduced to 3000 Milliseconds that is 3 seconds
4	Registration	If a user inputs same credentials for multiple times	User cannot use credentials for multiple times, if user wants to register the same email for multiple times, the application will toast message
5	Login	If a user inputs unregistered email and password to login	Application will check database if the credentials are registered or not to let user login to the application.
6	Visiting website	The data provided without HTTP:// did not work	Admin provide website link with starts with HTTP://

#	Testing Cases	Description and Problems	Improvements
7	Calling	Using non-international number format Application do not intent to dialer and did not call	Using international number format is reliable that everyone can call from every part of the world Using call_phone permission uses dialer that allows the user to confirm call by initiating the phone call.
9	View Map	Zoom in and zoom out was not working Location was not precise	CameraUpdateFactory functions to get pinch zoom in and out Using access_fine_loaction permission, that accesses application for precise location
10	Updating Information	If the user can update information or not because Admin also use the same application to update all the information provided	The user is not accessed to update any information because the application integrated with just a single credentials to Admin for login that can only access updating all the information to the application.
11	Adding College	Adds College even to non-existing University	Needs existing University that added already
12	Updating University	Cannot update University	Use of small letters so that application can rewrite the same letter to the database
13	Application size	Use of large size icon or other pictures	Using light size icons and pictures

#	Testing Cases	Description and Problems	Improvements
14	Application crashes	No matching id from one package to another package, database and PHP.	Checking and matching id that are inside every package and in database and in PHP

10 SUMMARY

The main aim of the project is to develop an application which provides information about the Colleges that is affiliated to several Universities. The other functionalities that are provided to this application are, to view College information, to make a call directly from the application using a dialer, to view a website using a web browser, to email to a college, to view the college's location on the map and to receive the notification message. Another main function of this application is to provide an administrator to add, to delete and to update a university's or a college's data and to send Push Notification.

On the completion of the project, the goals have been achieved. The project was tested on both an emulator and on an android device.

Moreover, the guidelines provided in this thesis could be helpful to the students who are eager to develop an android app.

11 CONCLUSIONS

Android development is a large platform to develop an application and the use of several programming languages like Java, XML, JSON, PHP and MySQL database. The development of Mobile applications is growing every day, and the use of these apps is comfortable and convenient.

This project helped me to learn many new things. Android development using Java especially challenged me to find out more about Java and the development of the application. I got too close with Android Architecture and its components. Also, I learned how does the power of programming language make so much difference in the area of technology.

11.1 Main Challenges

The main components used in this application are Google Maps API and Push notification. Learning about Google Map API and its implementation was a hard task but also interesting. Also making key from the parse.com and its implementation on the application. Use of JSON was also challenging for me because I was not so familiar with this language. Even the misplacement of a small dot or a comma crashes the application. The testing helped to figure out the problems and the solutions.

11.2 Future Tasks

Even though all the objected goals were achieved in the completion of the project there are many more things to be developed inside this application. Other future functions that can be develop for this application are:

- To make search functions using the search tab.
- To allow the user to validate and confirm his/her email address.
- To allow the user to fill in his/her other details like name, address and phone number that can make a user’s profile.
- To make other different setting features like forgot password and to delete the account permanently.

- To allow the administrator to upload pdf format prospectus and pictures of the College that can be downloaded by the user.

REFERENCES

/1/ What is Android. Accessed 25.03.2016
http://www.tutorialspoint.com/android/android_overview.htm

/2/ Android Architecture. Accessed 25.3.2016
http://www.eazytutz.com/android/android-architecture/

/3/ Android Activity Components. Accessed 26.3.2015
http://www.eazytutz.com/android/android-application-components/

/4/ Activity. Accessed 25.3.2016
https://developer.android.com/reference/android/app/Activity.html

/5/ Fragments. Accessed 26.3.2016
https://developer.android.com/guide/components/fragments.html

/6/ Thread vs Process and Service in Android – Part 1. Accessed 30.3.2016
https://msaudi.wordpress.com/2012/08/29/thread-vs-process-and-service-in-android-part-1/

/7/ Processes and Threads. Accessed 29.3.2016
https://developer.android.com/guide/components/processes-and-threads.html

/8/ UI Overview. Accessed 01.4.2016
https://developer.android.com/guide/topics/ui/overview.html

/9/ Android User Interface Design. Accessed 03.4.2016
http://www.eazytutz.com/android/android-user-interface-design/

/10/ Java Tutorial. Accessed 05.4.2016
http://www.tutorialspoint.com/java/

/11/ Java Basics for Android Development – Part 2. Accessed 05.4.2016
http://blog.teamtreehouse.com/java-basics-for-android-development-part-2

/12/ Java Basics for Android Development – Part 1. Accessed 06.4.2016
http://blog.teamtreehouse.com/java-basics-for-android-development-part-1

/13/ Java – Arrays. Accessed 07.4.2016
http://www.tutorialspoint.com/java/java_arrays.htm

/14/ Java – Inheritance. Accessed 07.4.2016
http://www.tutorialspoint.com/java/java_inheritance.htm

/15/ Java – Overriding. Accessed 10.4.2016
http://www.tutorialspoint.com/java/java_overriding.htm

/16/ Java – Polymorphism. Accessed 10.4.2016
http://www.tutorialspoint.com/java/java_polymorphism.htm

/17/ Java – Abstraction. Accessed 11.4.2016
http://www.tutorialspoint.com/java/java_abstraction.htm

/18/ Java – Encapsulation. Accessed 12.4.2016
http://www.tutorialspoint.com/java/java_encapsulation.htm

/19/ Michele E. Davis & Jon A. Phillips. 2007. Learning PHP and MySQL. 2nd ed. 1005 Gravenstein Highway North, Sebastopol, CA 95472. O'Reilly Media Inc.

/20/ PHP 5 Tutorial. Accessed 15.4.2016.
http://www.w3schools.com/php/default.asp

/21/ Database. Accessed 16.4.2016.
http://searchsqlserver.techtarget.com/definition/database

/22/ SQL Tutorial. Accessed 20.4.2016.
http://www.w3schools.com/sql/default.asp

/23/ MySQL Overview. Accessed 23.4.2016
https://www.mongodb.com/compare/mongodb-mysql

/24/ JSON Tutorial. Accessed 25.4.2016
http://www.w3schools.com/json/default.asp

/25/ Android – JSON Parser Tutorial. Accessed 27.4.2016
http://www.tutorialspoint.com/android/android_json_parser.htm

/26/ Org.json. Accessed 28.4.2016
https://developer.android.com/reference/org/json/package-summary.html

/27/ XML – Tutorial. Accessed 10.5.2016
http://www.tutorialspoint.com/xml/

/28/ Layouts. Accessed 12.5.2016
https://developer.android.com/guide/topics/ui/declaring-layout.html

/29/ Push Notification Accessed 12.5.2016
https://parse.com/docs/android/guide#push-notifications-setting-up-push

/30/ Introduction to the Google Maps Android API. Accessed 12.5.2016
https://developers.google.com/maps/documentation/android-api/intro#au-dience

/31/ Introduction to the Android, PHP, MySQL Databases, and JSON Mini Se-ries. Accessed 15.5.2016
http://www.mybringback.com/android-sdk/12924/android-tutorial-using-remote-databases-php-and-mysql-part-1/

/32/ Developer Workflow Basics. Accessed 16.5.2016
https://developer.android.com/studio/workflow.html?hl=id