

# **INDIAN HISTORY BUDDY FOR ANDROID**

Submitted in partial fulfillment of the requirements

of the degree of

**BACHELOR OF ENGINEERING**

**BY**

**SAGAR MANE 1014081**

**SANCHIT JAIN 1014086**

**SUSHIL THASALE 1014100**

Project Guide:

**Prof. Ms. Shailaja Gogate**



**DEPARTMENT OF INFORMATION TECHNOLOGY**

**K.J.SOMAIYACOLLEGE OF ENGINEERING, MUMBAI**

**UNIVERSITY OF MUMBAI**

**2013-2014**

Project Report Approval for B.E

**INDIAN HISTORY BUDDY FOR ANDROID**

by:

**Sagar Mane 1014081**

**Sanchit Jain 1014086**

**Sushil Thasale 1014100**

is approved for the degree of B. E. in Information Technology.

Internal Examiner

Name:

Sign:

External Examiner

Name:

Sign:

Project Guide

Name:

Sign:

Date:

Place:

# Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

1. -----

2. -----

3. -----

**1. Sagar Mane 1014081**

**2. Sanchit Jain 1014086**

**3. Sushil Thasale 1014100**

Date:

## **Abstract**

The main idea of this project is to design an application which will run on most the android handheld devices and will be helpful to find information about the monuments and museums all over India with their locations on the map using GPS. This application will allow users to create an account. It will give users various options to search for museums and monuments all over India. Also it will provide events going in and around the country throughout the year. It will give an additional functionality of setting reminder for the events on particular dates. It will also give users a function of viewing/adding comments.

# **Table of Contents**

1 Introduction .....	5
1.1 Existing System.....	6
1.2 Proposed System.....	6
1.3 Scope.....	6
2 Review of Literature.....	7
3 Report On Project Investigation.....	9
3.1 SPMP.....	10
3.1.1 Introduction.....	10
3.1.2 Project Organization.....	11
3.1.2.1 Software Process Model.....	11
3.1.3 Requirement Analysis and Definition.....	12
3.1.3.1 Functional Requirement.....	12
3.1.3.2 Non-functional Requirement.....	12
3.1.4 System and Software Design.....	13
3.1.5 Implementation and Unit Testing.....	14
3.1.6 Integration and System Testing.....	16
3.1.7 Roles and Responsibilities.....	16
3.1.7.1 Assumptions and Constraints.....	16
3.1.8 Technological Aspects.....	17
3.1.8.1 Android SDK.....	17
3.1.8.2 Eclipse.....	17
3.1.8.3 Xampp.....	18
3.1.8.4 SQLite.....	18

## Table of Contents continued...

3.1.9 Timeline Chart.....	19
3.2 Software Requirement Specification.....	20
3.2.1 Product Perspective.....	20
3.2.2 System Requirements.....	20
3.2.2.1 Hardware Requirements.....	20
3.2.2.2 Software Requirements.....	20
3.2.3 Communication Requirement.....	20
3.2.4 Software System Requirements.....	21
3.3 Software Design Documents.....	22
3.3.1 Design Overview.....	22
3.3.2 System Architecture Overview.....	22
3.3.3 Block Diagram of the System.....	23
3.3.3.1 UML Diagrams.....	23
3.3.3.1.1 Use Case Diagrams.....	23
3.3.4 User Interface.....	24-35
3.4 Implementation.....	36
3.4.1 Technologies Used.....	36
3.4.1.1 Hardware Used.....	36
3.4.1.2 Software Used.....	36
3.4.2 Source Code.....	37-61
3.5 Software test Plan and Test Cases.....	62
3.5.1 Introduction.....	62
3.5.1.1 System Overview.....	62

### **Table of Contents continued...**

3.5.1.2 Test Approach.....	62
3.5.1.2.1 Recovery Test.....	62
3.5.1.2.2 Beta Testing.....	63
3.5.1.2.3 User Acceptance Testing.....	63
3.5.2 Test Plan.....	63
3.5.2.1 Test Plan Objectives.....	63
3.5.2.2 Features to be tested.....	63
3.5.3 Testing tools and Environment.....	64
3.5.4 Test cases.....	64
4 Results and Discussions.....	67
5 Conclusion and Future Scope.....	68
5.1 Conclusion.....	69
5.2 Future Scope.....	69
6 Appendix.....	70-79
7 Literature Cited.....	81

**List of Figures and Tables :**

<b>Sr. No.</b>	<b>Name</b>	<b>Pg. No.</b>
1	Block Diagram	22
2	Flowchart	15
3	HLD and LLD	13-14
4	Use Case Diagram	23
5	Waterfall Model	11
6	User Interfaces	24-35
7	Timeline Chart	19
8	Test Cases	64-65



# **CHAPTER 1**

## **Introduction**

# **1. INTRODUCTION**

## **1.1 Existing System:**

There are various android applications which focuses on hotels all over India. There is no such application which provide information about museums and monuments all over India with maps and users location. Our application intrinsically provides information about museums and monuments all over India. Also it gives users information about events going in and around the country. Users can also set reminders for the same.

## **1.2 Proposed System:**

The heart of this system is successful database connectivity between user interface and MySQL database on Xampp server. We have used PHP to extract the server side data and display it on the user interface. The greatest advantage of Xampp server is that we donot need to write any code for creating various tables for various entities since this facility is provided by default. Our choice of language is PHP since it is easy to learn and easy to understand. In addition to that, it is very efficient and requires minimum lines of code.

## **1.3 Scope:**

This project is about developing an application on Android phones for the customers. The customers can have a look at all monuments/museums available under their choice of interest .The source code for this project is packaged to .apk format and ready for installation on android phone.

It helps to below specific user groups for their effective travel plan to visit these places:

1. **Educational Institutions** - Students to gain more information about historical monuments and museums.
2. **Tourists** - Our application can be used by tourists visiting India to plan their tour to these monuments and museums. This application will also help them to know the various upcoming events relative to their current location.

## **CHAPTER 2**

### **Review of Literature**

## **2. REVIEW OF LITERATURE**

1. Android city tour guide system based on Web service, by Li Lui,  
Dept. of Inf. & Eng., Shandong Jiao Tong University, China in April'2012.

Abstract:

In the paper, they propose the software development architecture based on Web services. This framework introduces the three-layer architecture of Web development into mobile phone software development. Based on the three-layer architecture, the android based Indian History Buddy is developed. The android based Indian History Buddy can realize to query information for monuments and museums all over India.

2. IEEE Paper –A Mobile Application to Access Remote Database using web Services in  
March'2012

Abstract:

This paper helped us to implement database connectivity using php scripting language. Using web services we could retrieve information on the Android GUI by accessing the xampp server.

## **CHAPTER 3**

### **Report On Project Investigation**

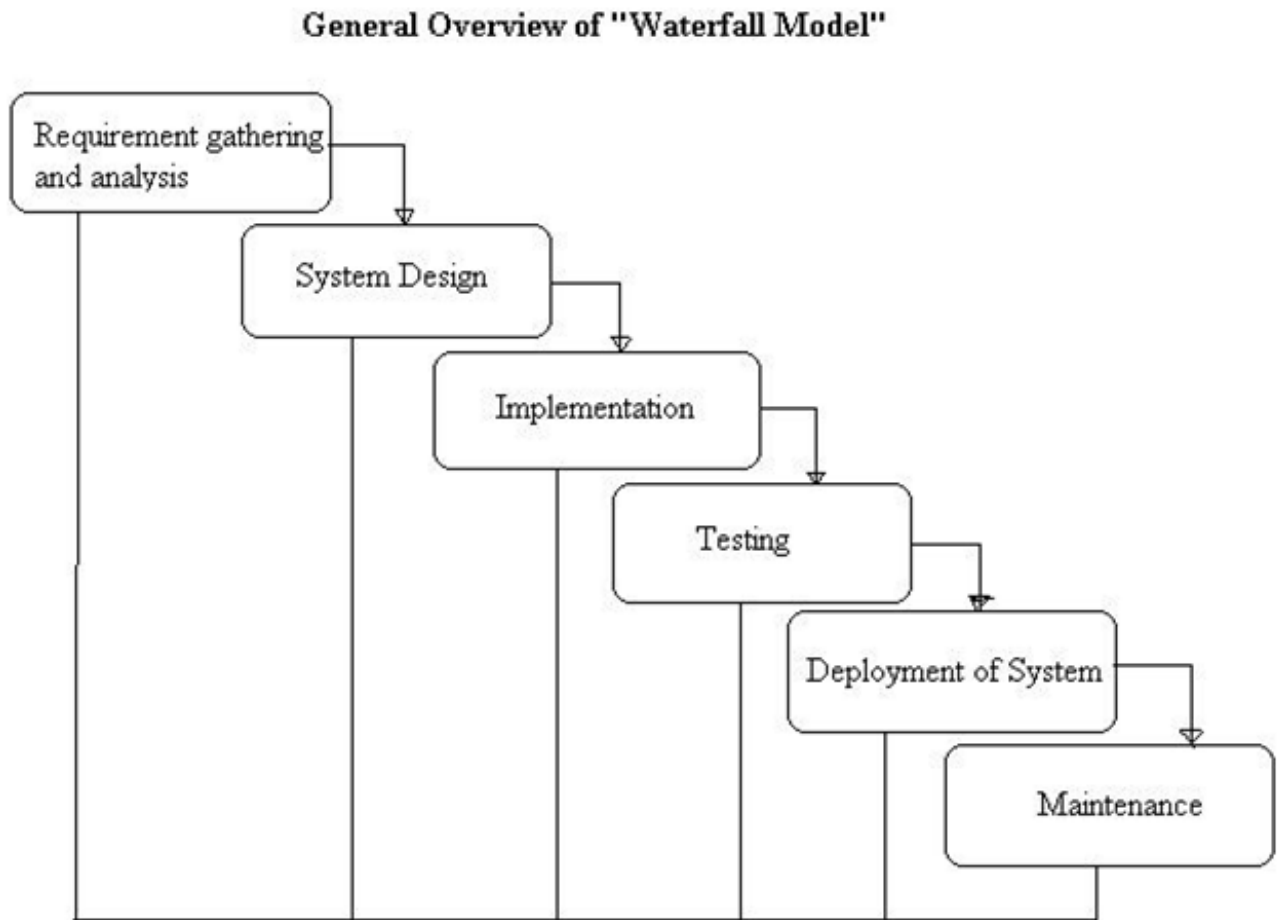
## **3.1 SOFTWARE PROJECT MANAGEMENT PLAN**

### **3.1.1 Introduction:**

This project is about developing an application on Android phones which provide an online information to public about historical monuments and museums in India on android handheld devices. This application also provides value added information about upcoming and current events in and around the city.

### 3.1.2 Project Organization:

#### 3.1.2.1 Software Process Model



The process model used is waterfall model. The waterfall model is a sequential software development model in which development is seen as flowing steadily downwards (like a waterfall) through several phases. This model combines elements of linear sequential model with iterative philosophy of prototyping.

### **3.1.3 Requirements Analysis and Definition**

#### **3.1.3.1 Functional Requirements:**

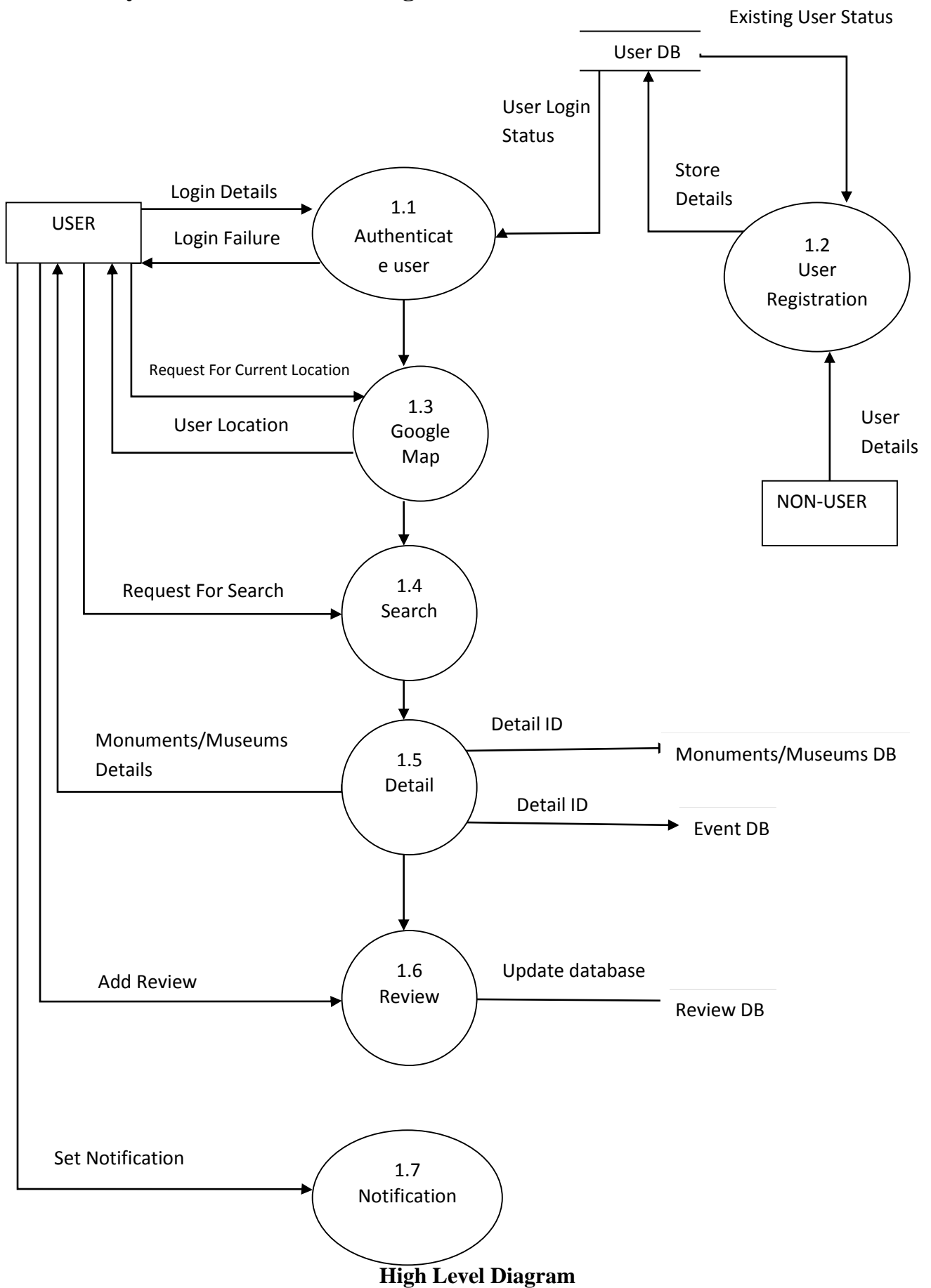
1. This application allows the user to create an account with valid password.
2. This application allows the user to search his/her desired historical monument or museums by location, name of the city, by state and by the name of the monument itself.
3. This application will notify users about the recent upcoming events happening in and around city.
4. This application will provide users to provide reviews related to the monuments/museums.
5. This application does not allow users to manipulate the data regarding the monuments/museums.
6. This application will allow users to view images and videos related to monuments and museums. It will also give emergency details.

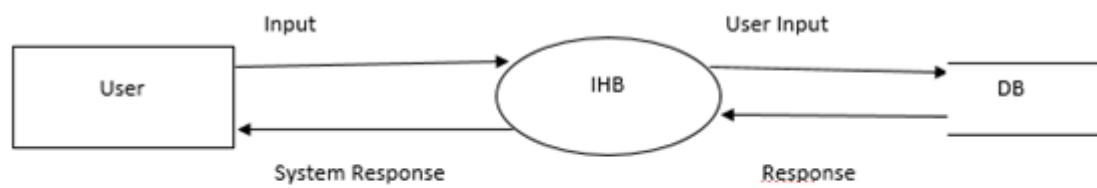
#### **3.1.3.2 Non-Functional Requirements:**

1. Installation: .apk will be available to all users and hence it is easy to install.
2. Availability: This Application will be available all the time because eventually it will be available on Google Play.
3. Security: Our application is highly secured since the passwords are stored in the encrypted format in the database.
4. Usability: The user interface is designed in such a way that any naive user will be able to operate and use the features of the application.



### 3.1.4 System and Software Design

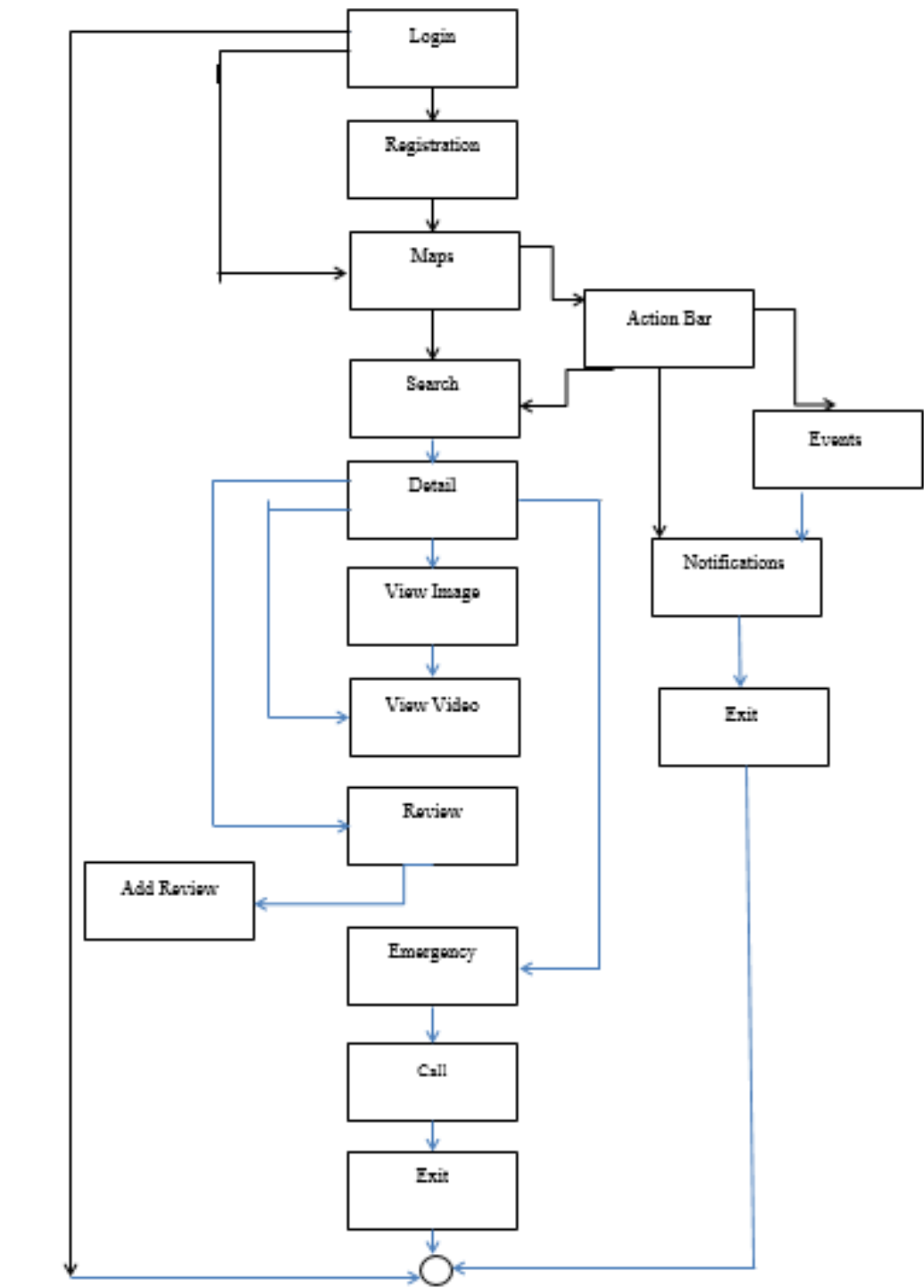




**Low Level Diagram**

### **3.1.5 Implementation and Unit Testing**

All the modules of our application have been developed and unit wise tested. The problems faced were API level requirement was high, coding errors, which were fixed. Thus due to successful testing it has been verified that all the modules meet user requirements. These modules were implemented on the emulator with API level 12 on Android sdk in Eclipse.



**Flowchart**

### **3.1.6 Integration and System Testing**

All the modules which were tested independently are linked using the intent object and later integrated with the database and server using PHP. Problems faced were linking between two activities, error in http connection, null pointer exception etc. major problem faced was linking an activity with listActivity and error in http connection which later solved by including permissions in manifest file and properly establishing the xampp server. After this system testing was performed which showed that the application runs successfully on all the platforms with android operating systems such as tablets, cell phones.

The units developed are integrated into a complete system during integration phase and tested to check if all the modules coordinate with each other and the system as a whole behaves as per the specification. After successfully testing the software, it is delivered to the customer.

### **3.1.7 Roles and Responsibilities:**

This project is done by a group of three people. All the three members have equally contributed to all the tasks in the project.

#### **3.1.7.1 Assumption and Constraint:**

Our project operated under the following assumptions:

- The users of this application are assumed to have Android based cell phones with internet connection in it.
- The server is up all the time so that data retrieval and storage is always efficient.
- The team had no human resources other than member of the group for project development.

We consider the following constraints:

- The time allotted for developing the project was not sufficient so we had to work on Saturdays and Sundays to meet the deadline.
- This application will not work on any other cell phone except Android based cell phones

### **3.1.8 Technological Aspect:**

Technologies used are as follows:

#### **3.1.8.1 Android SDK**

The Android SDK provides you the API libraries and developer tools necessary to build, test, and debug apps for Android.

If you're a new Android developer, we recommend you download the ADT Bundle to quickly start developing apps. It includes the essential Android SDK components and a version of the Eclipse IDE with built-in ADT (Android Developer Tools) to streamline your Android app development.

With a single download, the ADT Bundle includes everything you need to begin developing apps:

Eclipse + ADT plugin

Android SDK Tools

Android Platform-tools

The latest Android platform

The latest Android system image for the emulator

#### **3.1.8.2 Eclipse**

In computer programming, Eclipse is an integrated development environment (IDE). It contains a base workspace and an extensible plug-in system for customizing the environment. Written mostly in Java, Eclipse can be used to develop applications. By means of various plug-ins, Eclipse may also be used to develop applications in other programming languages: Ada, ABAP, C, C++, COBOL, Fortran, Haskell, JavaScript, Lasso, Natural, Perl, PHP, Python, R, Ruby (including Ruby on Rails framework), Scala, Clojure, Groovy, Scheme, and Erlang. It can also be used to develop packages for the software Mathematica. Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++ and Eclipse PDT for PHP, among others.

The Eclipse SDK includes the Eclipse Java development tools (JDT), offering an IDE with a built-in incremental Java compiler and a full model of the Java source files. This allows for advanced refactoring techniques and code analysis. The IDE also makes use of a workspace, in this case a set of metadata over a flat filesystem allowing external file modifications as long as the corresponding workspace "resource" is refreshed afterwards.

Eclipse implements widgets through a Java toolkit called SWT, whereas most Java applications use the Java standard Abstract Window Toolkit (AWT) or Swing. Eclipse's user interface also

uses an intermediate graphical user interface layer called JFace, which simplifies the construction of applications based on SWT.

### **3.1.8.3 Xampp**

XAMPP is a free and open source cross-platform web server solution stack package, consisting mainly of the Apache HTTP Server, MySQL database, and interpreters for scripts written in the PHP and Perl programming languages.

XAMPP requires only one zip, tar, 7z, or exe file to be downloaded and run, and little or no configuration of the various components that make up the web server is required. XAMPP is regularly updated to incorporate the latest releases of Apache, MySQL, PHP and Perl. It also comes with a number of other modules including OpenSSL and phpMyAdmin.

Self-contained, multiple instances of XAMPP can exist on a single computer, and any given instance can be copied from one computer to another.

It is offered in both a full, standard version and a smaller version.

### **3.1.8.4 SQLite**

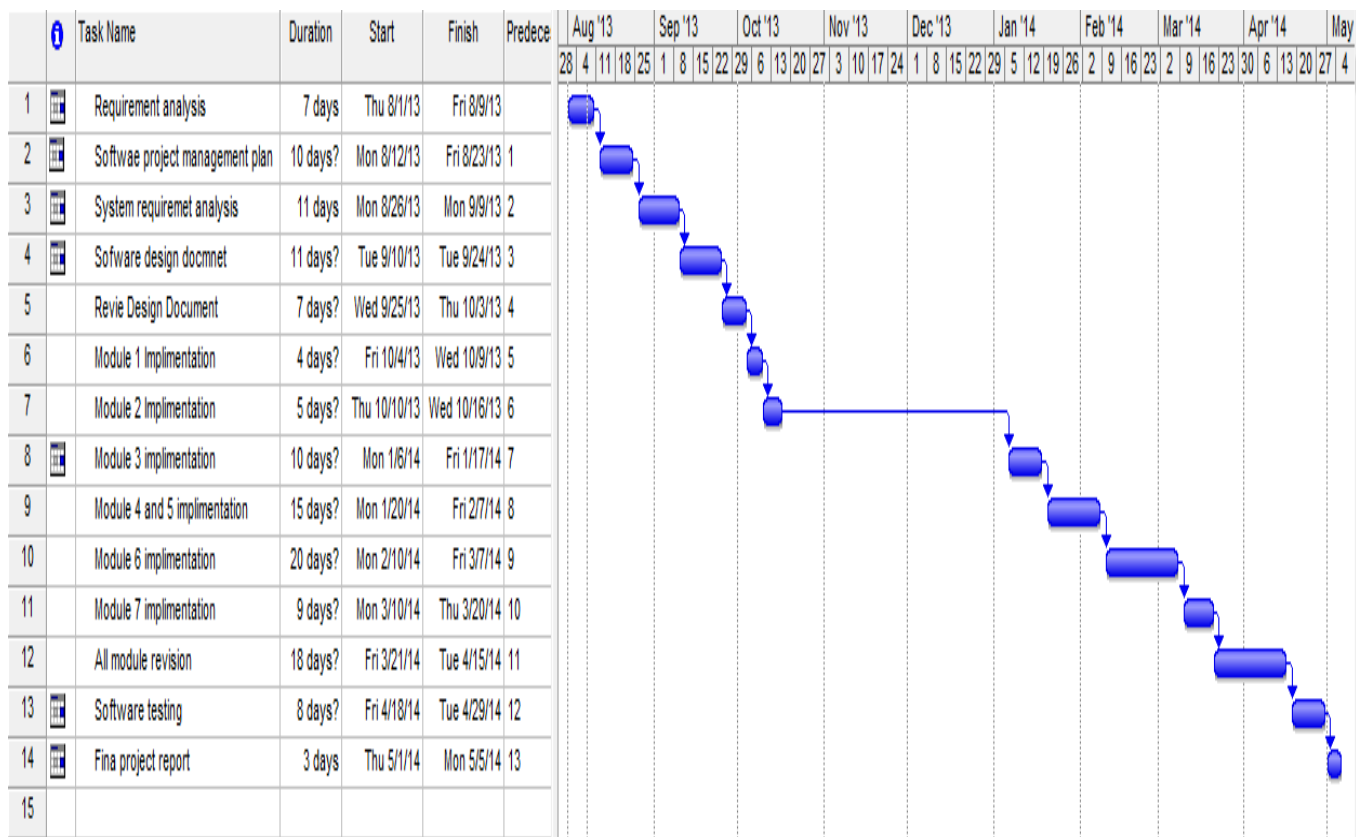
SQLite is an Open Source database. SQLite supports standard relational database features like SQL syntax, transactions and prepared statements. The database requires limited memory at runtime (approx. 250 Kbyte) which makes it a good candidate from being embedded into other runtimes

SQLite is embedded into every Android device. Using an SQLite database in Android does not require a setup procedure or administration of the database.

You only have to define the SQL statements for creating and updating the database. Afterwards the database is automatically managed for you by the Android platform.

Access to an SQLite database involves accessing the file system. This can be slow. Therefore it is recommended to perform database operations asynchronously

### 3.1.9 Timeline chart:



## **3.2 SOFTWARE REQUIREMENT SPECIFICATION**

### **3.2.1 Product Perspective:**

This application is a user - interactive application on the android platform. The project is its own entity and is hosted on android touch screen phones.

### **3.2.2 System Requirements:**

This application consists of a number of resources which will be bundled into an archive- an Android package(.apk). Programs are written in Java, built using the standard tools, and then the .apk is transferred via USB cable or Bluetooth and installed in phone using App installer software.

#### **3.2.2.1 Hardware Requirements:**

To develop the application the minimum hardware requirement:

- Pentium 4 and higher
- 1GB RAM and higher.

The physical characteristics of the application consists of various mobile devices that run the Android operating system. Mobile devices contain different hardware specifications, and the application will only require the minimal hardware requirements to operate fully.

#### **3.2.2.2 Software Requirements:**

- Operating System-Windows 7.
- Java Development Kit-Application will be developed in java programming language.
- Eclipse-An integrated Development Environment.
- Android SDK-Includes the Android JAR file as well as the Android documentation, tools, sample codes and emulator for simulation on computer before transferring the application on phone.

### **3.2.3 Communication Requirements:**

The application requires internet connection for accessing the information.



### **3.2.4 Software System Attributes:**

1. Installation: .apk file is available to all users and hence it is easy to install.
2. Availability: This Application will be available all the time because eventually it will be available on Google Play.
3. Security: Our application is highly secured since the passwords are stored in the encrypted format in the database.
4. Usability: The user interface is designed in such a way that any naive user will be able to operate and use the features of the application.

## **3.3 SOFTWARE DESIGN DOCUMENT**

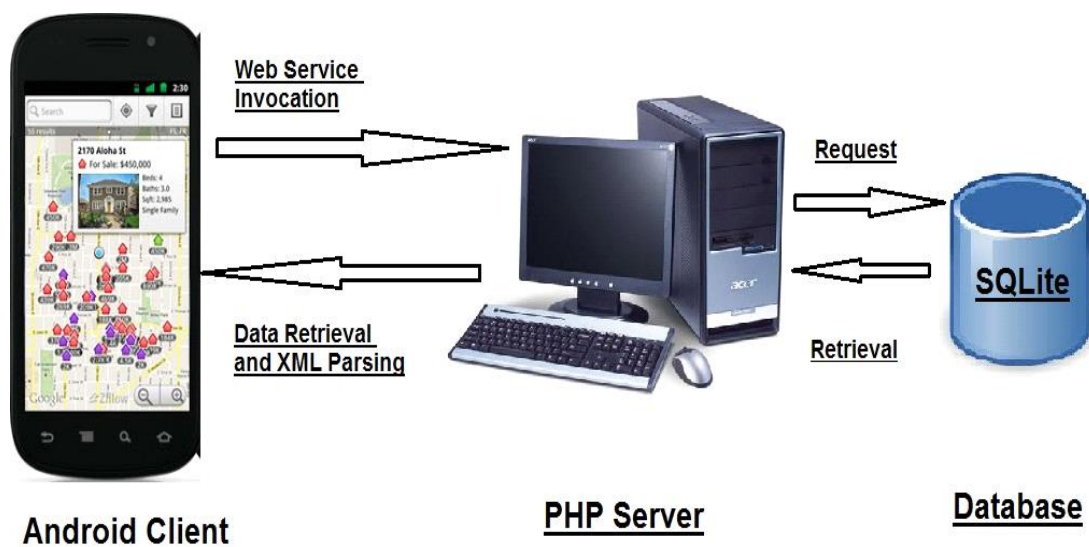
### **3.3.1 Design Overview:**

It is an android based application which provide an online information to public about historical monuments and museums in India on android handheld devices. This application also provides value added information about upcoming and current events in and around the city.

This application requires xampp server connectivity since the data feeding and retrieval are done on the database which lies on the server.

### **3.3.2 System Architecture Overview:**

The major components of our application is internet connection, which will connect our application to the server (Google store, laptop). All the information regarding monuments and museums, will be downloaded from the server and will be displayed to the user. In our application unlimited users will be able to use our application and connect to our server with their Tablet or Smartphone.



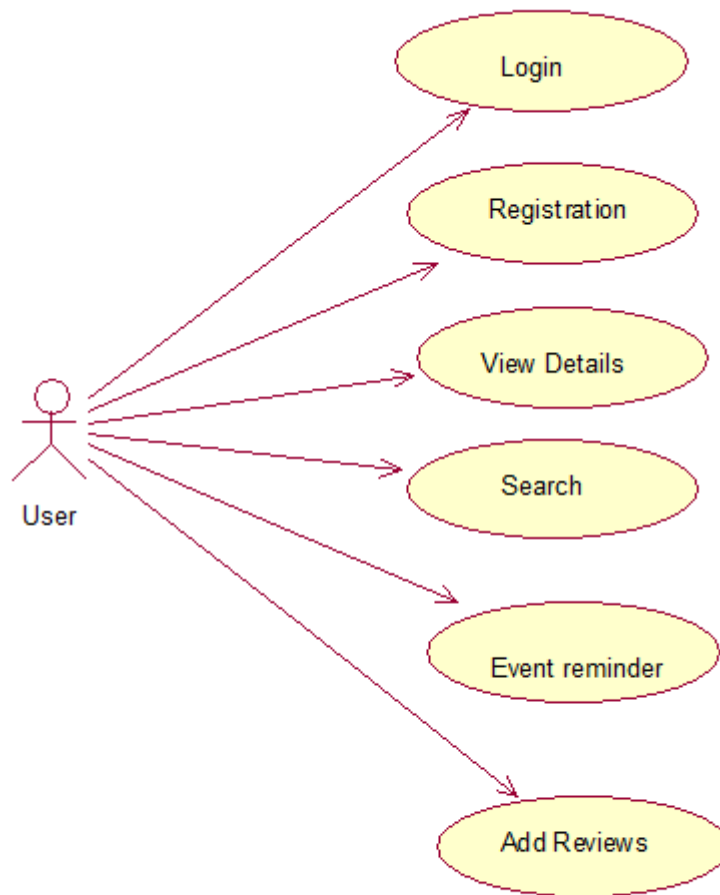
**Block Diagram of the System**

### 3.3.3 Block Diagram of the System

Indian History Buddy uses a three tier architecture where Android smartphone is at the client tier, php server at the second tier and Xampp server at the database tier. When the user clicks the Point of Interest (POI) on the maps a request is sent to the PHP web service through JSON object. The php web service then packs the information in a JSON object to send it back to the client side. At the client side XML parsing is used to extract the information stored in JSON objects and it is displayed.

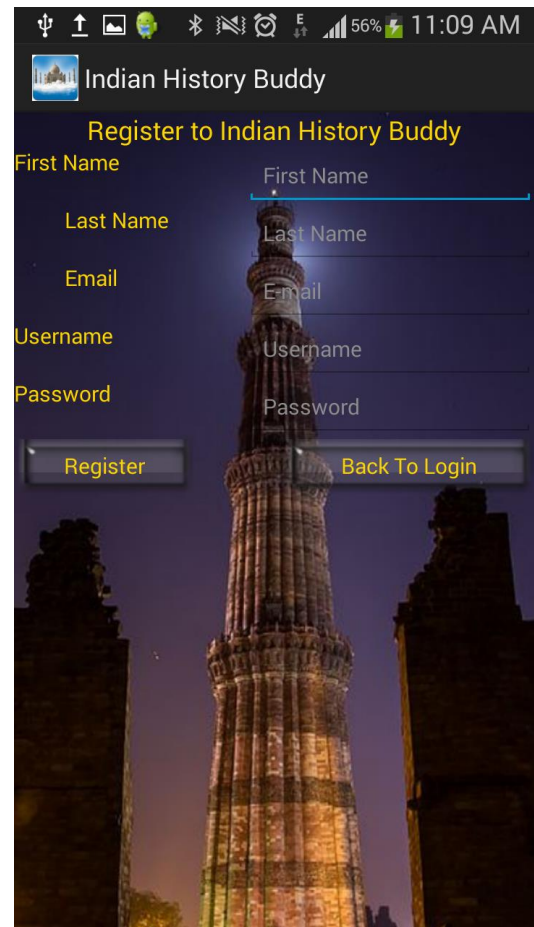
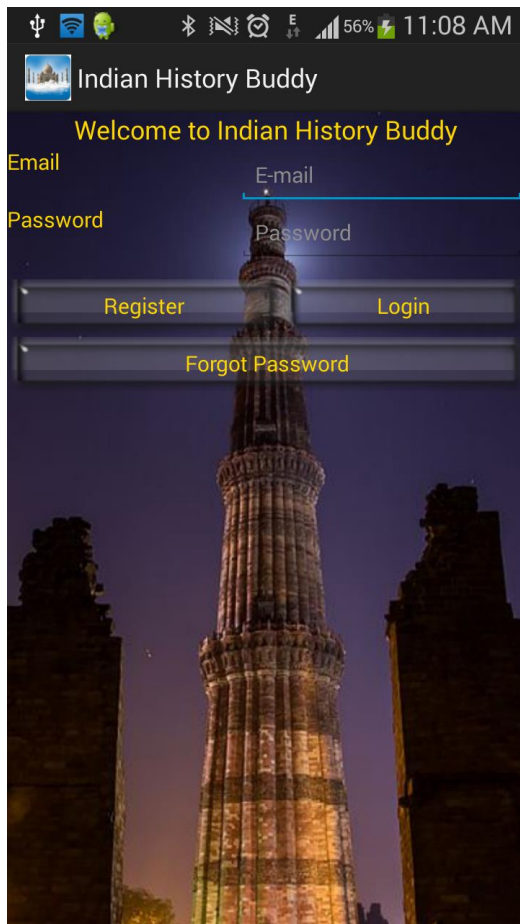
### 3.3.1 UML DAIGRAMS

#### 3.3.1.1 Use Case Diagram:



### 3.3.4 USER INTERFACES

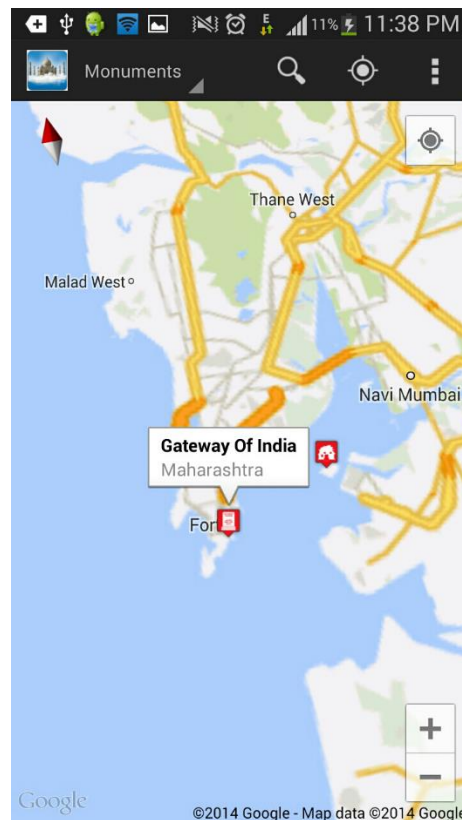
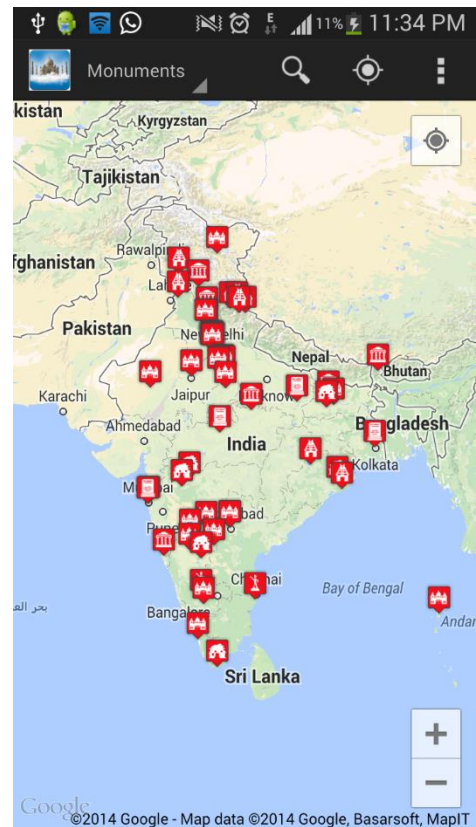
#### 1. Module1- User Login and Registration :



**Name:** Login and Registration

**Function:** The UI of the login and registration page is developed using xml. Its functionality is developed using java. When the user enters the details and clicks the login button a php web service is called which verifies the username and the corresponding password. If the username and password are correct the user will be allowed to view the maps. When the user registers the user details will be passed to php webservice and stored and stored in the users database.

## 2. Module2-Google Maps and Markers :



**Name:** Google Maps and Markers

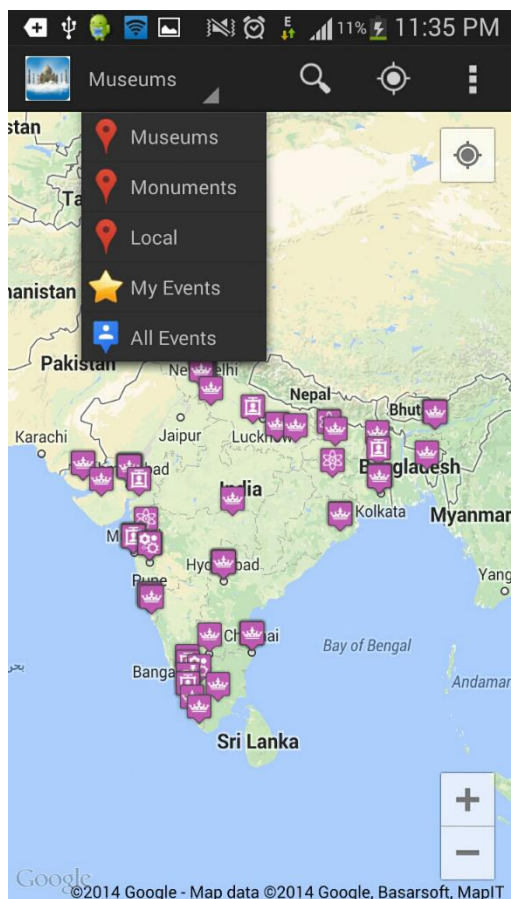
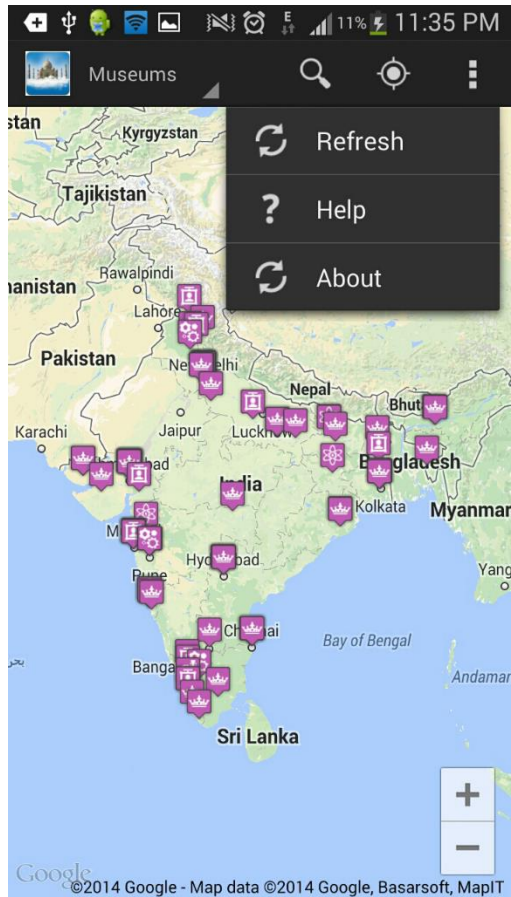
**Function:** Android maps and markers module is responsible for displaying the google maps and the markers of different monuments and museums .It will also display the users current location.

Android Maps are developed using a google webservice named Google Maps API v2.The UI is designed using a map fragment in the xml file.

MainActivity.java contains the functionality of this module and the applications manifest file contains the permissions required to access Google maps api v2.

When the user clicks on the marker info window for the corresponding marker will be displayed with a tittle and a snippet.Clicking the info window will take the user to Monuments/Museums Details Activity

### 3. Module3- Action Bar :



**Name:** Action Bar

**Function:** Action bar is a window feature that offers user actions and navigation modes. Using the action bar the user can navigate to museums activity, events activity or search activity. Action bar is called using the function `onCreateOptionsMenu()` which inflates corresponding design stored in xml file. Action bar consists of navigation items and action items. The click events of the navigation items are handled through `onNavigationItemSelected()` function. The click events of the action items on the action bar are handled using `onOptionsItemSelected()` function.



#### 4. Module4- Search activity :



11:38 PM

Search

BY STATE

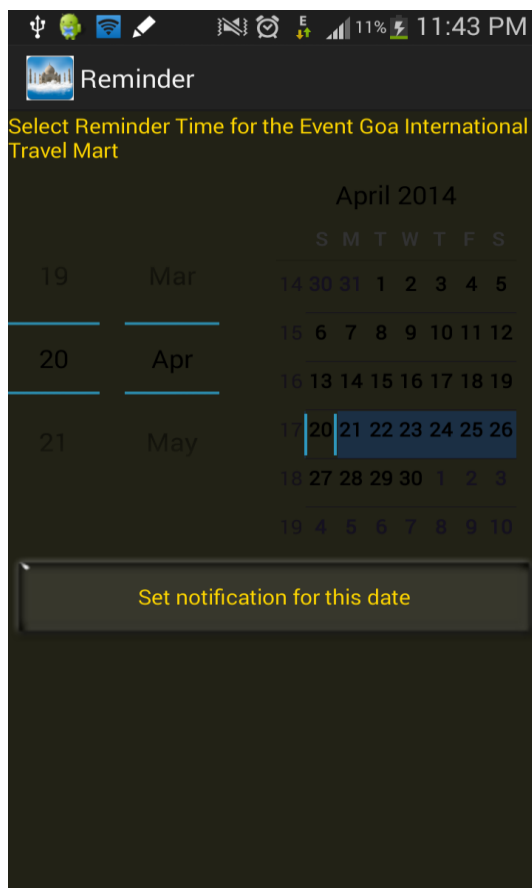
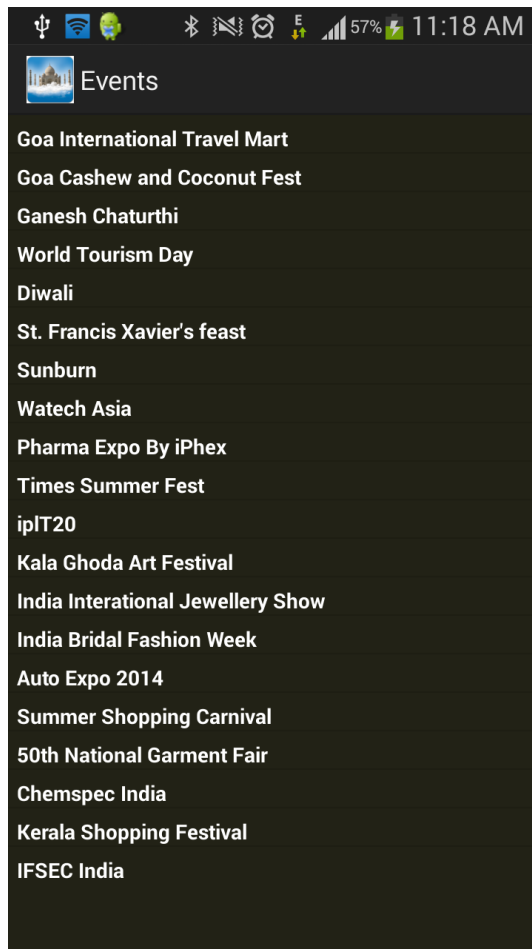
BY NAME



**Name:** Search Activity

**Function:** This module is responsible for downloading information from the database server and displaying the information about all the upcoming events. moreover the user can set reminder for the specific event for a particular date. The user will get a reminder notification about the upcoming events on the specified date. this is implemented using alarm manager API. This class provides access to the system alarm services. These allow you to schedule your application to be run at some point in the future. When an alarm goes off, the Intent that had been registered for it is broadcast by the system, automatically starting the target application if it is not already running. Registered alarms are retained while the device is asleep (and can optionally wake the device up if they go off during that time), but will be cleared if it is turned off and rebooted. When the user sets a reminder, the details of the particular event will be stored in SQLite database. When the user clicks on the notification, application will navigate to NotificatiEventDeatils.java activity, which extracts information from the SQLite database and display it.

## 5. Module5- Reminder Notification :




**Name:** Reminder Notification

**Function:** This module is responsible for retrieving the requested data from the database. The database is hosted by the xampp server. The android application is connected to the database server through php web services. When the user sends a request the request parameters are packaged in a JSON object. The JSON object calls the corresponding php webservice. The webservice connects to the database and retrieves the requested information. This information is then packed in a JSON object and sent as a response to the user's smartphone.

## 6. Module6- Database Connectivity with Android GUI :





## Details

NEW Delhi. Taj Mahal is the tomb of the Mughal Empress Mumtaz Mahal. After 22 years, and the mutual effort of over 20,000 labors and master craftsmen, the tomb was finally completed in 1648 AD on the banks on the river Yamuna in Agra, the 17th century capital of the Mughal monarchs. The colors of the Taj Mahal change at different hours of the day and during different seasons. The Taj Mahal of Agra is pinkish in the morning, milky white in the evening and golden when the moon shines. Like a jewel, the Taj Mahal sparkles in moonlight when the semi-precious stones inlaid into the white marble on the main tomb catch the shine of the moon. These changes, they say, represents the different moods of woman. The source of the name "Taj Mahal" is not understandable. Court histories from Shah Jahan reign only call it the tomb of Mumtaz Mahal. It is generally believed that "Taj Mahal" is shortened version of her name, Mumtaz Mahal.

[VIEW IMAGE](#)

[VIEW VIDEO](#)

[VIEW REVIEW](#)

[EMERGENCY](#)



## Images

Taj Mahal



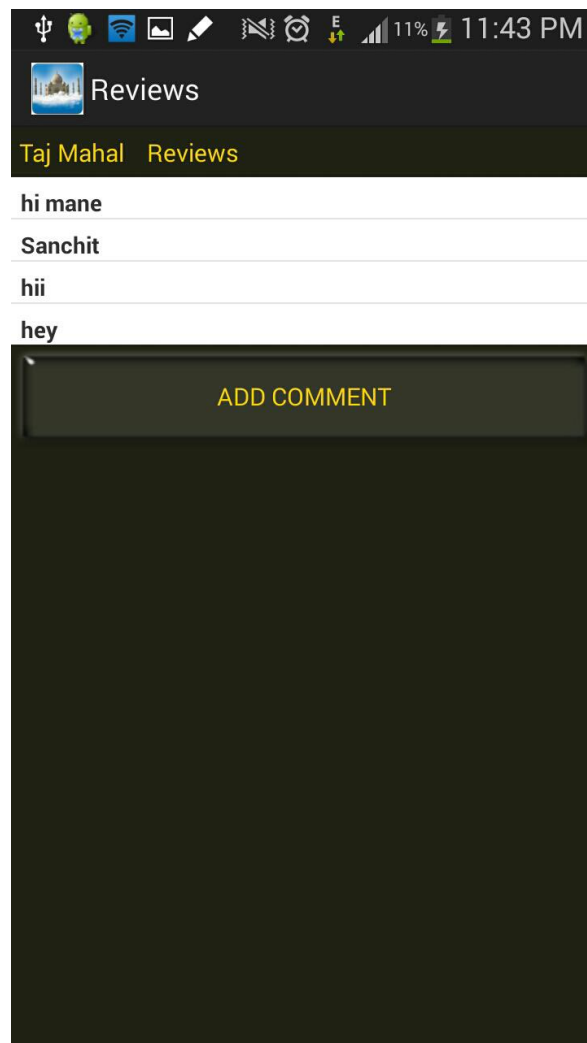
Taj Mahal



**Name:** Database Connectivity with Android GUI

**Function:** Using this module the user can search monuments or museums by state or by its name. If the user selects by state option, all states will be displayed in an expandable list view. When the user selects a particular state the particular list item will expand to display the monuments or museums within that state. If the user selects by name option all the monuments or museums all over India will be displayed in a list view.

#### 7. Module7- User Reviews :



**Name:** User Reviews

**Function:** This module is responsible for displaying all the reviews for the particular monument or museum in a listview. Moreover it allows the user to add reviews.

## **3.4 IMPLEMENTATION**

### **3.4.1 Technologies used**

#### **3.4.1.1 Hardware Used:**

To develop the application the minimum hardware requirement:

- Pentium 4 and higher
- 1GB RAM and higher.

The physical characteristics of the application consists of various mobile devices that run the Android operating system. Mobile devices contain different hardware specifications, and the application will only require the minimal hardware requirements to operate fully.

#### **3.4.1.2 Software Used:**

- Operating System-Windows 7.
- Java Development Kit-Application will be developed in java programming language.
- Eclipse-An integrated Development Environment.
- Android SDK-Includes the Android JAR file as well as the Android documentation, tools, sample codes and emulator for simulation on computer before transferring the application on phone.

#### **Source Code**

There are seven modules in our project. The project is been developed in eclipse environment. The user interface of these modules is developed in XML and are linked to each other using java which uses intent object for linking. Also the connectivity code for connecting the user interface to the database is written in java file of the activity. The database for the project is created on Xampp server using PHP as the server side coding. The PHP files stores and retrieves the data to server and from the database. The database contains tables for storing all the information about monuments and museums.



### 3.4.2 Source Code:

#### MainActivity.java

```
package com.example.actionbarnew;

import info.actionbar.model.SpinnerNavItem;
import info.actionbar.adapter.TitleNavigationAdapter;

import android.app.ActionBar;
import java.util.ArrayList;

import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.MapFragment;
import com.google.android.gms.maps.GoogleMap.OnInfoWindowClickListener;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.CameraPosition;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;

import android.app.ActionBar;
import android.os.AsyncTask;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.Menu;
import android.view.MenuItem;

public class MainActivity extends Activity implements ActionBar.OnNavigationListener{

    private static final String TAG_NAME = "name";
    private static final String TAG_PID = "pid";
```

```

/*final String[] monuments = {

    "Taj Mahal",
    "Gateway of India",
    "Buland Darwaja",
    "Char Minar",
    "Gol Gumbad",
    "Statue of Gomateswara",
    "Hampi",
    "Humayun Tomb",
    "India Gate",
    "Khajuraho Temples",
    "Mahabalipuram Rathas",
    "Mattancherry Palace",
    "Mysore Palace",
    "Vivekananda Rock",
    "Nalanda",
    "Qutub Minar",
    "Safdarjung Tomb",
    "Sanchi Stupa",
    "Dhamekh Stupa",
    "Victoria Memorial",
    "Cellular Jail",
    "Hawa Mahal",
    "Leh Palace",
    "Agra Fort",
    "Ajanta Caves",
    "Akbar's Tomb",
    "Basilica of BOM JESUS",
    "Ellora Caves",
    "Golconda Fort",
    "Golden Temple",
    "Gwalior Fort",
    "Tughlaqabad fort",
    "Purana Quila",

```

```

        "Jama Masjid",
        "Red Fort",
        "Barabar Caves",
        "Khuda Bakhsh Oriental Library",
        "Kurukshetra",
        "Jyotisar",
        "Panchkula",
        "Dharamshala",
        "Vaishno Devi",
        "Badami",
        "Gulbarga",
        "Raichur",
        "Elephanta Caves",
        "Konark Sun Temple",
        "The Leaning Temple of Huma",
        "Udayagiri and Khandagiri Caves",
        "Jodhpur",
        "Rumtek Monastery",
        "Badrinath",
        "Kedarnath",
        "Gangotri",
        "Yamunotri"}; */

```

```
// action bar
```

```
private ActionBar actionBar;
```

```
// Title navigation Spinner data
```

```
private ArrayList<SpinnerNavItem> navSpinner;
```

```
// Navigation adapter
```

```
private TitleNavigationAdapter adapter;
```

```
// Refresh menu item
```

```
private MenuItem refreshMenuItem;
```

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    // all map activities
    // Get a handle to the Map Fragment
    GoogleMap map = ((MapFragment) getFragmentManager()
        .findFragmentById(R.id.map)).getMap();

    //my location enabled
    map.setMyLocationEnabled(true);

    //enable zoom functionality
    map.getUiSettings().setZoomControlsEnabled(true);

    //compass enabled
    map.getUiSettings().setCompassEnabled(true);

    //my location button
    map.getUiSettings().setMyLocationButtonEnabled(true);

    //enable rotate gesture
    map.getUiSettings().setRotateGesturesEnabled(true);

    //moving camera to a location
    CameraPosition cameraPosition = new CameraPosition.Builder().target(
        new LatLng(23.481285,77.739738)).zoom(4).build();
    map.animateCamera(CameraUpdateFactory.newCameraPosition(cameraPosition));

    map.setMapType(GoogleMap.MAP_TYPE_NORMAL);
    //map.setMapType(GoogleMap.MAP_TYPE_HYBRID);
    //map.setMapType(GoogleMap.MAP_TYPE_SATELLITE);
    //map.setMapType(GoogleMap.MAP_TYPE_TERRAIN);

```

```

//map.setMapType(GoogleMap.MAP_TYPE_NONE);

//monuments markers

map.addMarker(new MarkerOptions()
    .position(new LatLng(27.175009,78.0420906))
    .title("Taj Mahal")
    .snippet("Uttar Pradesh")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(18.92186,72.83449))
    .title("Gateway Of India")
    .snippet("Maharashtra")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.memorial)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(27.09441,77.662886))
    .title("Buland Darwaja")
    .snippet("Uttar Pradesh")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(17.361564,78.474664))
    .title("Char Minar")
    .snippet("Andhra Pradesh")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(16.830136,75.735068))
    .title("Gol Gumbad")
    .snippet("Karnataka")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(12.854042,76.484558))
    .title("Statue of Gomateswara")
    .snippet("Karnataka")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.statue)));
map.addMarker(new MarkerOptions()

```

```

.position(new LatLng(15.333333,76.466667))
.title("Hampi")
.snippet("Karnataka")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.rockhouse)));
map.addMarker(new MarkerOptions()
.position(new LatLng(28.5927639,77.2498549))
.title("Humayun Tomb")
.snippet("Delhi")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.memorial)));
map.addMarker(new MarkerOptions()
.position(new LatLng(28.61280,77.23006))
.title("India Gate")
.snippet("Delhi")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.memorial)));
map.addMarker(new MarkerOptions()
.position(new LatLng(24.852971,79.919664))
.title("Khajuraho Temples")
.snippet("Madhya Pradesh")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.temple)));
map.addMarker(new MarkerOptions()
.position(new LatLng(12.608925,80.189512))
.title("Mahabalipuram Rathas")
.snippet("Tamil Nadu")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.statue)));
map.addMarker(new MarkerOptions()
.position(new LatLng(9.958291,76.259408))
.title("Mattancherry Palace")
.snippet("Kerala")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
.position(new LatLng(12.305135,76.655148))
.title("Mysore Palace")
.snippet("Karnataka")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()

```

```

.position(new LatLng(8.07815,77.555323))
.title("Vivekananda Rock")
.snippet("Tamil Nadu")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.rockhouse)));
map.addMarker(new MarkerOptions()
.position(new LatLng(25.203805,85.509734))
.title("Nalanda")
.snippet("Bihar")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
.position(new LatLng(28.526407,77.187028))
.title("Qutub Minar")
.snippet("Delhi")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
.position(new LatLng(28.589308,77.210547))
.title("Safdarjung Tomb")
.snippet("Delhi")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.memorial)));
map.addMarker(new MarkerOptions()
.position(new LatLng(23.481285,77.739738))
.title("Sanchi Stupa")
.snippet("Madhya Pradesh")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.memorial)));
map.addMarker(new MarkerOptions()
.position(new LatLng(25.380902,83.024515))
.title("Dhamekh Stupa")
.snippet("Uttar Pradesh")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.memorial)));
map.addMarker(new MarkerOptions()
.position(new LatLng(22.544546,88.343024))
.title("Victoria Memorial")
.snippet("West Bengal")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.memorial)));
map.addMarker(new MarkerOptions()

```

```

.position(new LatLng(11.674192,92.747807))
.title("Cellular Jail")
.snippet("Andaman")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
.position(new LatLng(26.923936,75.826744))
.title("Hawa Mahal")
.snippet("Rajasthan")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));

map.addMarker(new MarkerOptions()
.position(new LatLng(34.165772,77.586565))
.title("Leh Palace")
.snippet("Jammu And Kashmir")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
.position(new LatLng(27.179533,78.021112))
.title("Agra Fort")
.snippet("Uttar Pradesh")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
.position(new LatLng(20.551868,75.703262))
.title("Ajanta Caves")
.snippet("Maharashtra")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.rockhouse)));
map.addMarker(new MarkerOptions()
.position(new LatLng(27.219107,77.949593))
.title("Akbar's Tomb")
.snippet("Uttar Pradesh")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.memorial)));
map.addMarker(new MarkerOptions()
.position(new LatLng(15.500788,73.911599))
.title("Basilica of BOM JESUS")
.snippet("Goa")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.temple)));

```



```

map.addMarker(new MarkerOptions()
    .position(new LatLng(20.032343,75.175548))
    .title("Ellora Caves")
    .snippet("Maharashtra")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.rockhouse)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(17.3853626,78.4041297))
    .title("Golconda Fort")
    .snippet("Andhra Pradesh")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(31.61998,74.876485))
    .title("Golden Temple ")
    .snippet("Punjab")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.templehindu)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(26.221761,78.1664189))
    .title("Gwalior Fort")
    .snippet("Madhya Pradesh")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(28.5169126,77.2591938))
    .title("Tughlaqabad Fort")
    .snippet("Delhi")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(28.6153509,77.2368603))
    .title("Purana Quila")
    .snippet("Delhi")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(28.651162,77.234129))
    .title("Jama Masjid")
    .snippet("Uttar Pradesh")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.temple)));

```

```

map.addMarker(new MarkerOptions()
    .position(new LatLng(28.655813,77.241952))
    .title("Red Fort")
    .snippet("Delhi")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(25.0130556,85.0519444))
    .title("Barabar Caves")
    .snippet("Bihar")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.rockhouse)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(25.619169,85.16282))
    .title("Khuda Bakhsh Oriental Library")
    .snippet("Bihar")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.temple)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(29.9695121,76.878282))
    .title("Kurukshetra")
    .snippet("Haryana")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(29.9615995,76.7697201))
    .title("Jyotisar")
    .snippet("Haryana")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(30.6942091,76.860565))
    .title("Panchkula")
    .snippet("Haryana")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.temple)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(32.2616629,76.3067954))
    .title("Dharamshala")
    .snippet("Himachal Pradesh")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.temple)));

```

```

map.addMarker(new MarkerOptions()
.position(new LatLng(33.0255377,74.9384135))
.title("Vaishno Devi")
.snippet("Jammu And Kashmir")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.templehindu)));
map.addMarker(new MarkerOptions()
.position(new LatLng(15.91494,75.676811))
.title("Badami")
.snippet("Karnataka")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
.position(new LatLng(17.329731,76.8342957))
.title("Gulbarga")
.snippet("Karnataka")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
.position(new LatLng(16.20702,77.354362))
.title("Raichur")
.snippet("Karnataka")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
.position(new LatLng(18.963253,72.931444))
.title("Elephanta Caves")
.snippet("Maharashtra")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.rockhouse)));
map.addMarker(new MarkerOptions()
.position(new LatLng(19.887595,86.094536))
.title("Konark Sun Temple")
.snippet("Odisha")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.templehindu)));
map.addMarker(new MarkerOptions()
.position(new LatLng(21.45243,83.967133))
.title("The Leaning Temple Of Huma")
.snippet("Odisha")
.icon(BitmapDescriptorFactory.fromResource(R.drawable.templehindu)));

```

```

map.addMarker(new MarkerOptions()
    .position(new LatLng(20.263115,85.785725))
    .title("Udayagiri and Khandagiri Caves")
    .snippet("Odisha")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.rockhouse)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(26.2389469,73.0243094))
    .title("Jodhpur")
    .snippet("Rajasthan")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.palace)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(27.288671,88.561518))
    .title("Rumtek Monastery")
    .snippet("Sikkim")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.temple)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(30.7433085,79.4937634))
    .title("Badrinath")
    .snippet("Uttarakhnda")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.templehindu)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(30.735232,79.066898))
    .title("Kedarnath")
    .snippet("Uttarakhnda")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.templehindu)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(30.9946945,78.9398402))
    .title("Gangotri")
    .snippet("Uttarakhnda")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.templehindu)));
map.addMarker(new MarkerOptions()
    .position(new LatLng(30.9968657,78.4616526))
    .title("Yamunotri")
    .snippet("Uttarakhnda")
    .icon(BitmapDescriptorFactory.fromResource(R.drawable.templehindu)));

```

```

//handling info window click
map.setOnInfoWindowClickListener(new OnInfoWindowClickListener() {
    @Override
    public void onInfoWindowClick(Marker marker) {

        String monument_selected=marker.getTitle();
        String pid="";

        String[] str = {

            "Taj Mahal",

            "Gateway of India",

            "Buland Darwaja",

            "Char Minar",

            "Gol Gumbad",

            "Statue of Gomateswara",

            "Hampi",

            "Humayun Tomb",

            "India Gate",

            "Khajuraho Temples",

            "Mahabalipuram Rathas",

            "Mattancherry Palace",

```

"Mysore Palace",

"Vivekananda Rock",

"Nalanda",

"Qutub Minar",

"Safdarjung Tomb",

"Sanchi Stupa",

"Dhamekh Stupa",

"Victoria Memorial",

"Cellular Jail",

"Hawa Mahal",

"Leh Palace",

"Agra Fort",

"Ajanta Caves",

"Akbar's Tomb",

"Basilica Of Bom Jesus",

"Ellora Caves",

"Golconda Fort",

"Golden Temple",

"Gwalior Fort",

"Hawa Mahal",

"Tughlaqabad fort",

"Purana Quila",

"Jama Masjid",

"Red Fort",

"Barabar Caves",

"Khuda Bakhsh Oriental Library",

"Kurukshetra",

"Jyotisar",

"Panchkula",

"Dharamshala",

"Vaishno Devi",

"Badami",

"Gulbarga",

"Raichur",

"Elephanta Caves",

"Konark Sun Temple",

"The Leaning Temple of Huma",

"Udayagiri and Khandagiri Caves",

"Jodhpur",

"Rumtek Monastery",

"Badrinath",

"Kedarnath",

"Gangotri",

"Yamunotri"

};

for(int i=0;i<str.length;i++)

{

if(str[i].equals(monument\_selected))

{ int n=i+1;

pid="" + n;

System.out.println("pid in for is " + pid);

break;}

}



```

        Intent in = new Intent(getApplicationContext(),
                                   MonumentsInfoActivity.class);

        // sending pid to next activity
        in.putExtra(TAG_NAME, monument_selected);
        in.putExtra(TAG_PID, pid);

        // starting new activity and expecting some response back
        startActivityForResult(in, 1000);

    }
});

```

```

//all action bar activities
actionBar = getActionBar();

// Hide the action bar title
actionBar.setDisplayShowTitleEnabled(false);

// Enabling Spinner dropdown navigation
actionBar.setNavigationMode(ActionBar.NAVIGATION_MODE_LIST);

// Spinner title navigation data
navSpinner = new ArrayList<SpinnerNavItem>();
navSpinner.add(new SpinnerNavItem("Monuments", R.drawable.ic_location));
navSpinner.add(new SpinnerNavItem("Museums", R.drawable.ic_location));

navSpinner.add(new SpinnerNavItem("Local", R.drawable.ic_location));
navSpinner.add(new SpinnerNavItem("My Events",
R.drawable.ic_my_places));
navSpinner.add(new SpinnerNavItem("All Events", R.drawable.ic_latitude));

```

```

        // title drop down adapter
        adapter = new TitleNavigationAdapter(getApplicationContext(),
            navSpinner);

        // assigning the spinner navigation
        actionBar.setListNavigationCallbacks(adapter, this);

        // Changing the action bar icon
        // actionBar.setIcon(R.drawable.ico_actionbar);

    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.actions, menu);
        return true;
    }

    /**
     * On selecting action bar icons
     */
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Take appropriate action for each action item click
        switch (item.getItemId()) {
            case R.id.action_search:
                // search action
                search();
                return true;
            case R.id.action_location_found:
                // location found

                return true;
        }
    }

```

```

        case R.id.action_refresh:
            // refresh
            refreshMenuItem = item;
            // load the data from server
            new SyncData().execute();

            return true;
        case R.id.action_help:
            // help action
            return true;
        case R.id.action_check_updates:
            // check for updates action
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}

/**
 * Launching new activity
 * */
private void search() {
    Intent i = new Intent(MainActivity.this, SearchMonuments.class);
    startActivity(i);
}

/**
 * ActionBar navigation item select listener
 * */
@Override
public boolean onNavigationItemSelected(int itemPosition, long itemId) {
    // Action to be taken after selecting a spinner item
    switch(itemPosition)
    {
        case 0: break;
    }
}

```

```

        case 1 : museums();
        break;
        case 3 : myEvents();
        break;
        case 4: allEvents();
        break;

    }

    return false;
}

private void myEvents() {
    Intent i = new Intent(MainActivity.this, MyEvents.class);
    startActivity(i); }

private void allEvents() {
    Intent i = new Intent(MainActivity.this, AllEventsActivity.class);
    startActivity(i); }

private void museums() {
    Intent i = new Intent(MainActivity.this, Museums.class);
    startActivity(i); }

/**
 * Async task to load the data from server
 * **/
private class SyncData extends AsyncTask<String, Void, String> {
    @Override
    protected void onPreExecute() {
        // set the progress bar view
        refreshMenuItem.setActionView(R.layout.action_progressbar);
    }
}

```

```

        refreshMenuItem.expandActionView();
    }

    @Override
    protected String doInBackground(String... params) {
        // not making real request in this demo
        // for now we use a timer to wait for sometime
        try {
            Thread.sleep(3000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        return null;
    }

    @Override
    protected void onPostExecute(String result) {
        refreshMenuItem.collapseActionView();
        // remove the progress bar view
        refreshMenuItem.setActionView(null);
    }
};

}

```

activity\_main.xml

```

<?xml version="1.0" encoding="utf-8"?>
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/map"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:name="com.google.android.gms.maps.MapFragment"/>

```

AlarmTask.java

```

package com.example.actionbarnew.alarm;
import java.util.Calendar;

import android.app.AlarmManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;

import com.example.actionbarnew.services.NotifyService;

public class AlarmTask implements Runnable{
    // The date selected for the alarm
    private final Calendar date;
    // The android system alarm manager
    private final AlarmManager am;
    // Your context to retrieve the alarm manager from
    private final Context context;

    public AlarmTask(Context context, Calendar date) {
        this.context = context;
        this.am = (AlarmManager)
context.getSystemService(Context.ALARM_SERVICE);

        this.date = date;
    }

    @Override
    public void run() {
        // Request to start are service when the alarm date is upon us
        // We don't start an activity as we just want to pop up a notification into the
system bar not a full activity
        Intent intent = new Intent(context, NotifyService.class);
        intent.putExtra(NotifyService.INTENT_NOTIFY, true);
        PendingIntent pendingIntent = PendingIntent.getService(context, 0, intent, 0);

```

```

        // AlarmManager alarmManager = (AlarmManager)
getSystemService(ALARM_SERVICE);
        // Sets an alarm - note this alarm will be lost if the phone is turned off and on
again
        am.set(AlarmManager.RTC, date.getTimeInMillis(), pendingIntent);
        am.setRepeating(AlarmManager.RTC,date.getTimeInMillis(),(24 * 60 * 60 *
1000),pendingIntent);
    }
}

```

NotifyService.java

```

package com.example.actionbarnew.services;

```

```

import android.R;
import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.app.Service;
import android.content.Intent;
import android.os.Binder;
import android.os.IBinder;
import android.util.Log;

import com.example.actionbarnew.SecondActivity;

public class NotifyService extends Service {

    /**
     * Class for clients to access
     */

    public class ServiceBinder extends Binder {
        NotifyService getService() {
            return NotifyService.this;
        }
    }
}

```

```

        // Unique id to identify the notification.
        private static final int NOTIFICATION = 123;

        // Name of an intent extra we can use to identify if this service was started to create a
notification
        public static final String INTENT_NOTIFY =
"com.example.services.INTENT_NOTIFY";

        // The system notification manager
        private NotificationManager mNM;

        @Override
        public void onCreate() {
            Log.i("NotifyService", "onCreate()");
            mNM = (NotificationManager)
getSystemService(NOTIFICATION_SERVICE);
        }

        @Override
        public int onStartCommand(Intent intent, int flags, int startId) {
            Log.i("LocalService", "Received start id " + startId + ": " + intent);

            // If this service was started by out AlarmTask intent then we want to show our
notification
            if(intent.getBooleanExtra(INTENT_NOTIFY, false))
                showNotification();

            // We don't care if this service is stopped as we have already delivered our
notification
            return START_NOT_STICKY;
        }

        @Override
        public IBinder onBind(Intent intent) {
            return mBinder;
        }

        // This is the object that receives interactions from clients

```



```

private final IBinder mBinder = new ServiceBinder();

/**
 * Creates a notification and shows it in the OS drag-down status bar
 */
private void showNotification() {
    // This is the 'title' of the notification
    CharSequence title = "Reminder From IHB";
    // This is the icon to use on the notification
    int icon = R.drawable.ic_dialog_info;
    // This is the scrolling text of the notification
    CharSequence text = "You Have Upcoming Events";
    // What time to show on the notification
    long time = System.currentTimeMillis();

    Notification notification = new Notification(icon, text, time);

    // The PendingIntent to launch our activity if the user selects this notification
    PendingIntent contentIntent = PendingIntent.getActivity(this, 0, new
Intent(this, SecondActivity.class), 0);

    // Set the info for the views that show in the notification panel.
    notification.setLatestEventInfo(this, title, text, contentIntent);

    // Clear the notification when it is pressed
    notification.flags |= Notification.FLAG_AUTO_CANCEL;

    // Send the notification to the system.
    mNM.notify(NOTIFICATION, notification);

    // Stop the service when we are finished
    stopSelf();
}
}

```

## **3.5 SOFTWARE TEST PLAN AND TEST CASES**

### **3.5.1 Introduction:**

#### **3.5.1.1 System Overview:**

This project is about developing an application on Android phones which provide an online information to public about historical monuments and museums in India on android handheld devices. This application also provides value added information about upcoming and current events in and around the city. The project provides set of functions to the users such as creating an account, adding, updating, and uploading the images.

The source code for this project is packaged to .apk format and ready for installation on android phone.

#### **3.5.1.2 Test Approach:**

We have first conducted tests on each individual functions in this application as separate entities using the Android Emulator supplied by the Android Software Development Kit(SDK). Once each individual checkboxes, radio buttons are tested thoroughly, the package is built together and tested as a whole. All known valid inputs as well as known invalid inputs. A more comprehensive overview of our testing strategies is included in our testing specification documentation. Test strategies consists of a series of different tests that will fully exercise the software. The primary purpose of these tests is to uncover the systems limitations and measure its full capabilities. Also the database connectivity is tested properly to check whether the data is being retrieved properly or not. A list of the various executed tests and a brief explanation follows below.

##### **3.5.1.2.1 Recovery Test**

Recovery tests will force the system to fail in a various ways and verify that the recovery is properly performed. If our application fails under high load, it will just force closed and data will be retrieved if user again opens the application. If our application crashes the user can just reinstall the .apk file. There will be no loss of data since the data is stored on the server. Therefore the recovery test was successfully performed on our project except in case of server crash.

### **3.5.1.2.2 Beta Testing**

The team will perform Beta test for the application and will report any defects they find during the initial phases of the development. This will subject the system to tests that could not be performed in our test environment.

### **3.5.1.2.3 User Acceptance Testing**

Once the application was ready for implementation, the users performed User acceptance Testing. All the basic functionalities of the application such as viewing, uploading and deleting were working properly and meeting all the user requirements. The application was installed in 10-15 android based cell phones. The users of these cell phones accessed the application simultaneously to check for the response of the application and the latency time. In this case, the response from the application was slow with latency of around 2 minutes. Thus our application successfully passed the user acceptance testing.

## **3.5.2 Test Plan**

### **3.5.2.1 Test Plan Objectives:**

This Test Plan document for software supports the following objectives:

- Identify existing project information and software components to be tested.
- List recommended requirements for Test (high level).
- Recommend and describe testing strategies to be employed.
- Identify required resources and provide a test effort estimate.
- List the test project deliverable elements.
- Define the activities required to prepare for and conduct System, Beta and User Acceptance testing.
- Communicate to all responsible parties the System Test strategy.
- Define deliverables and responsible parties.

### **3.5.2.2 Features To Be Tested:**

The following is a list of functions that will be tested:

- User Interface –The user interface is tested successfully in unit and integration testing.
- Memory Constraints testing.

- All the functionalities are tested by the developers in system testing and the users in user acceptance testing.
- Database connectivity-It is tested to check whether the data from server is retrieved successfully or not. Also to check whether the data was successfully stored on the server in integration and system testing.

### 3.5.3 Testing Tools and Environment

#### 3.5.3.1 Development Environment

- Android Platform
- Android SDK
- Eclipse IDE

### 3.5.4 TEST CASES

Test Case ID	Test Case	Input	Expected Output	Actual Output	Pass/Fail
1.	This application should resume in case of intermediate calls or messages.	An incoming call or a text message.	The application should resume itself after the end of call or after receiving message.	Application resumes.	Pass
2.	Verify the screen resolution	Basic GUI of the application.	The scene should fit in properly in the limited area of the phone screen	The resolution supports the scenes of the application very well.	Pass
3.	Application should properly regain control	Forcing an error in an application.	The application should properly	The functionalities and application controls are	Pass

	after an error message.		resume control.	properly resumed.	
4.	Tap on the screen ten times at different positions, the application should not freeze.	Forcing certain things that can result in application freezing.	Application should not hang.	Application working as expected.	Pass
5.	Applications confirming to the features described in the documents.	Re-evaluating mentioned features that confirm to it.	All major functions should confirm to the features in the document.	All major functionalities confirming to documentation.	Pass
6.	Alteration or corruption of data when handset is suddenly powered off.	Switch off the cell when the application is on.	The application should not corrupt or alter any of the phone data.	Phone functions and data unaltered.	Pass
7.	To check whether the app suffers when multiple applications are running.	Run multiple phone applications at once.	There should not be any major effects in the application functionality.	No memory constraints due to multiple applications turned on.	Pass

## **CHAPTER 4**

### **Results and Discussions**

## **4. RESULTS AND DISCUSSIONS**

Our project INDIAN HISTORY BUDDY has been successfully implemented with the following features:

1. Integration of the xampp server:

The functionality of our application to store and retrieve the data has been successfully been implemented by making use of the xampp server as the storage system and a successful connection has also been obtained.

2. Use of buttons for the processing:

Any user can very easily navigate through the application since very easily understood terms are used over the buttons to instruct for the guidance.

## **CHAPTER 5**

### **Conclusion and Future Scope**



## **5. CONCLUSIONS AND FUTURE SCOPE**

### **5.1 Conclusion:**

It was a great experience to do this project. We got to learn many things as team work led to knowledge sharing, indeed making the entire process very informative. Also we faced many challenges while developing the project, which we are happy to overcome successfully. We enjoyed a lot working with android and are satisfied with our efforts since we could implement our idea successfully. Also our knowledge in Android increased after working on this project. Although there are many modifications that can be done to make the project more efficient and user friendly. Many new features can also be incorporated in the current application to build the project on a larger scale for future scope.

### **5.2 Future scope:**

In future, this project can be extended to national as well as international level. However for that lot of research work is required, database will be huge and management of such a database will be a herculean task. In addition to that various advertisements and videos regarding properties can be added. This application will be uploaded on Google Play which is the market for android applications so that anyone can download the application when required and use it.

## **CHAPTER 6**

### **APPENDIX**

## **6. APPENDIX**

Source Code:

JSONParser.java

```
package com.example.actionbarnew;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.net.URI;
import java.net.URISyntaxException;
import java.util.List;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.client.utils.URLEncodedUtils;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONException;
import org.json.JSONObject;

import android.util.Log;

public class JSONParser {

    static InputStream is = null;
    static JSONObject jObj = null;
    static String json = "";
```

```

// constructor
public JSONParser() {

}

// function get json from url
// by making HTTP POST or GET 2method
public JSONObject makeHttpRequest(String url, String method,
                                   List<NameValuePair> params) {

    // Making HTTP request
    try {

        // check for request method
        if(method == "POST"){
            // request method is POST
            // defaultHttpClient
            DefaultHttpClient httpClient = new DefaultHttpClient();
            HttpPost httpPost = new HttpPost(url);
            httpPost.setEntity(new UrlEncodedFormEntity(params));

            HttpResponse httpResponse = httpClient.execute(httpPost);
            HttpEntity httpEntity = httpResponse.getEntity();
            is = httpEntity.getContent();

        }else if(method == "GET"){
            // request method is GET
            DefaultHttpClient httpClient = new DefaultHttpClient();
            String paramString = URLEncodedUtils.format(params, "utf-8");

            url += "?" + paramString;
            System.out.println("url is" + url);

            HttpGet httpGet = new HttpGet(url);

```

```

        System.out.println("httpget is"+httpGet);

        try {
            httpGet.setURI(new URI(url));
        } catch (URISyntaxException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

        HttpResponse httpResponse = httpClient.execute(httpGet);
        System.out.println("httpresponse is is"+httpResponse);

        HttpEntity httpEntity = httpResponse.getEntity();
        is = httpEntity.getContent();
    }

} catch (UnsupportedEncodingException e) {
    e.printStackTrace();
} catch (ClientProtocolException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}

try {
    BufferedReader reader = new BufferedReader(new
InputStreamReader(

        is, "iso-8859-1"), 8);
    StringBuilder sb = new StringBuilder();
    String line = null;
    while ((line = reader.readLine()) != null) {
        sb.append(line + "\n");
    }
    is.close();
    json = sb.toString();
}

```

```

        } catch (Exception e) {
            Log.e("Buffer Error", "Error converting result " + e.toString());
        }

        // try parse the string to a JSON object
        try {
            jsonObj = new JSONObject(json);
        } catch (JSONException e) {
            Log.e("JSON Parser", "Error parsing data " + e.toString());
        }

        // return JSON String
        return jsonObj;
    }
}

```

DatabaseHandler.java

```

package com.example.actionbarnew;

import java.util.HashMap;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DatabaseHandler extends SQLiteOpenHelper {

    // All Static variables
    // Database Version
    private static final int DATABASE_VERSION = 1;

```

```

// Database Name
private static final String DATABASE_NAME = "android_api";

// Login table name
private static final String TABLE_LOGIN = "login";

// Login Table Columns names
private static final String KEY_ID = "id";
private static final String KEY_NAME = "name";
private static final String KEY_EMAIL = "email";
private static final String KEY_UID = "uid";
private static final String KEY_CREATED_AT = "created_at";

public DatabaseHandler(Context context) {
    super(context, DATABASE_NAME, null, DATABASE_VERSION);
}

// Creating Tables
@Override
public void onCreate(SQLiteDatabase db) {
    String CREATE_LOGIN_TABLE = "CREATE TABLE " + TABLE_LOGIN
+ "("
        + KEY_ID + " INTEGER PRIMARY KEY,"
        + KEY_NAME + " TEXT,"
        + KEY_EMAIL + " TEXT UNIQUE,"
        + KEY_UID + " TEXT,"
        + KEY_CREATED_AT + " TEXT" + ")";
    db.execSQL(CREATE_LOGIN_TABLE);
}

// Upgrading database
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    // Drop older table if existed
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_LOGIN);
}

```

```

        // Create tables again
        onCreate(db);
    }

    /**
     * Storing user details in database
     * */
    public void addUser(String name, String email, String uid, String created_at) {
        SQLiteDatabase db = this.getWritableDatabase();

        ContentValues values = new ContentValues();
        values.put(KEY_NAME, name); // Name
        values.put(KEY_EMAIL, email); // Email
        values.put(KEY_UID, uid); // Email
        values.put(KEY_CREATED_AT, created_at); // Created At

        // Inserting Row
        db.insert(TABLE_LOGIN, null, values);
        db.close(); // Closing database connection
    }

    /**
     * Getting user data from database
     * */
    public HashMap<String, String> getUserDetails(){
        HashMap<String,String> user = new HashMap<String,String>();
        String selectQuery = "SELECT * FROM " + TABLE_LOGIN;

        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.rawQuery(selectQuery, null);
        // Move to first row
        cursor.moveToFirst();
        if(cursor.getCount() > 0){
            user.put("name", cursor.getString(1));

```



```

        user.put("email", cursor.getString(2));
        user.put("uid", cursor.getString(3));
        user.put("created_at", cursor.getString(4));
    }
    cursor.close();
    db.close();

    // return user
    return user;
}

/**
 * Getting user login status
 * return true if rows are there in table
 * */
public int getRowCount() {
    String countQuery = "SELECT * FROM " + TABLE_LOGIN;
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor cursor = db.rawQuery(countQuery, null);
    int rowCount = cursor.getCount();
    db.close();
    cursor.close();

    // return row count
    return rowCount;
}

public void resetTables(){
    SQLiteDatabase db = this.getWritableDatabase();
    // Delete All Rows
    db.delete(TABLE_LOGIN, null, null);
    db.close();
}
}

```

fetchbyidmuseums.php

<?php

\$response = array();

// include db connect class

require\_once \_\_DIR\_\_ . '/db\_connect.php';

// connecting to db

\$db = new DB\_CONNECT();

// check for post data

if (isset(\$\_GET["pid"])) {

    \$pid= \$\_GET['pid'];

// get a product from products table

\$result = mysql\_query("SELECT \*FROM museums WHERE pid=\$pid");

if (!empty(\$result)) {

    // check for empty result

    if (mysql\_num\_rows(\$result) > 0) {

        \$result = mysql\_fetch\_array(\$result);

        \$product = array();

        \$product["name"] = \$result["name"];

        \$product["state"] = \$result["state"];

        \$product["visitor\_info"] =

\$result["visitor\_info"];

        \$product["description"] =

\$result["description"];

```

$response["success"] = 1;

// user node
$response["product"] = array();

array_push($response["product"], $product);

// echoing JSON response
echo json_encode($response);
} else {
    // no product found
    $response["success"] = 0;
    $response["message"] = "No product found";

    // echo no users JSON
    echo json_encode($response);
}
} else {
    // no product found
    $response["success"] = 0;
    $response["message"] = "No product found";

    // echo no users JSON
    echo json_encode($response);
}
} else {
    // required field is missing
    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";

    // echoing JSON response
    echo json_encode($response);
}
?>

```

**CHAPTER 7**  
**LITERATURE CITED**

## **7. LITERATURE CITED**

### **Papers:**

1. “A Mobile Application to Access Remote Database using Web Services”, an IEEE paper by Karan Balkar, Shweta Tripathi, Department of Computer Engineering, Fr.C.Rodrigues Institute of Technology, Vashi.
2. “Remote Access of Building Management System on Windows                      Mobile Devices”, an IEEE paper by Onderj Krejcar, Department of measurement and control, VSB Technical Institute of Ostrava Czech Republic.

### **Books:**

1. Android Application Development by Lauren Darcy.
2. The Busy Coder's Guide to Android Development by Mark L Murphy.
3. Beginning Android Application Development by Wei-Meng Lee.

### **Websites:**

1. <http://www.android-x86.org/>
2. [http:// developer .android.com/index.html](http://developer.android.com/index.html)
3. <http://www.androidpolice.com/2011/08/23/learning-android-development-here-is-a-200-episode-almost-20-hours-tutorial-series-all-for-free-videos/>