MOBILE APPLICATION FOR EVENT UPDATES

by

SATYA SAGAR VANTEDDU

B.Tech., Jawaharlal Technological University, 2011

A REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2016

Approved by:

Major Professor
Dr. Mitchell Neilsen

Abstract

It is really tough for someone new to a place to search for new addresses or navigate from one building to another. Further, it would help the user to keep track of event at each building/location in the app itself rather than noting them down on a piece of paper. Keeping these in mind, the idea is to develop a mobile app through which one can easily search for addresses/buildings which might not be available in google maps and also should be getting updates regarding the events at that location. This can be achieved by storing the latitudes and longitudes of each building or any specific location through the website which in turn is updated in the mobile application. The user can further check the events at each building which will be updated from the website.

The application is divided into two parts, user module and admin module. The user would be using a mobile app to perform the required operations while the admin is responsible for storing the locations and events. Since there are a huge percentage of both android, iOS and windows users these days, it would be beneficial to build a hybrid app which would work cross platform rather than building a native app. I would be using the native browsers of these operating systems to achieve this.

The admin module is a website through which locations and events at specified locations can be stored based on dates in an attached relational database. This module can only be accessed by authorized users and is responsible for all the updates visible for the mobile app user.

The completely developed app can be used by any organization or institution to be used by their staff/students, etc.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to express my sincere gratitude to my Major Professor, Dr. Mitchell Neilsen for his encouragement and for trusting my abilities to complete this project.

I take immense pleasure in extending my heartfelt thanks to my committee members Dr. Daniel Andresen and Dr. Torben Amtoft for their encouragement and for taking the time to serve on my committee.

I take this opportunity to acknowledge the support and help received from the academic staff of the Department of Computing and Information Sciences.

I would like to thank my parents and friends for their immense love and belief.

Chapter 1 - Introduction

1.1 Project Description

The proposed application ‘Event Management’ is designed to provide a solution for problems faced by anyone new to a place. This application addresses a couple of very common problems. Finding a specific building, even though google maps locate most of the locations there are few buildings that cannot be located because of the organization’s compliance or other reasons. Another common problem anyone faces is being unaware of most of the events going on within the organization. With this app installed in your mobile device, the user can search for any building within the organization and also get updates about the events going on in all the buildings/ locations.

The system is built in two modules, Admin module, for the organization to provide data and User module for the user to access the data which the organization is providing. The Admin module is a secured website which can be only by the organization through which the locations can be updated and also the events on daily basis. This is achieved using PHP, HTML, Bootstrap and MySQL. The User module is a cross platform mobile application which would be available for users to download and use. Users would be able to search for buildings and check for events on daily basis. This is achieved using Cordova and JavaScript.

1.2 Motivation

Being an international student, me along with my friends faced these problems when we were new to Manhattan and the idea of building an app to provide a solution for this emerged. There are also several events going around in KSU which are generally communicated through e-mails long before the actual date which somehow are overlooked. The idea of user being able to see all the events happening on the present day got me excited to work on this application.

Chapter 2 - Background and Technologies Used

The two modules to build this system are mostly achieved using web application features. Admin module is built using a PHP framework called CodeIgniter. CodeIgniter is a powerful PHP framework with a small footprint, built for those who required a simple, yet elegant toolkit to build fully-featured web applications. It requires nearly zero configuration, with no restrictive coding rules, and offers simple solutions to complex application requirements. CodeIgniter features a rich set of libraries for commonly needed tasks, as well as a simple interface and logical structure to access these libraries. It's exceptionally fast, as its core system only requires a few small libraries, with additional libraries loaded dynamically upon requests, based on your needs for a given process. This means the base system is both lean and agile. As CodeIgniter uses the MVC controller approach, it allows for great separation between logic and presentation, particularly useful for projects in which designers are working on template files [1].

User module is built using Apache Cordova PhoneGap, jQuery mobile and JavaScript. PhoneGap provides access to the services and hardware of the phone by way of a JavaScript library. This library is the same for all phones, so you can really write a single application code base and use it on a range of different platforms [2]. This application interacts with the server through Ajax calls to get the necessary data that can be viewed by the user.

2.1 CodeIgniter

CodeIgniter is a PHP-driven Application Development Framework - a toolkit to build web sites using PHP [3]. CodeIgniter lets the user develop projects much faster since there is no need to write code from scratch since it has a rich set of libraries for commonly needed tasks and further has a simple interface to access these libraries. This framework has small footprint with and has very minimum to no server requirements and also is compatible with wide versions of PHP.

CodeIgniter uses the Model-View-Controller approach, which allows great separation between logic and presentation. We describe MVC in more detail later.

2.1.1 Application Flow Chart

The following graphic illustrates how data flows throughout the system in CodeIgniter:

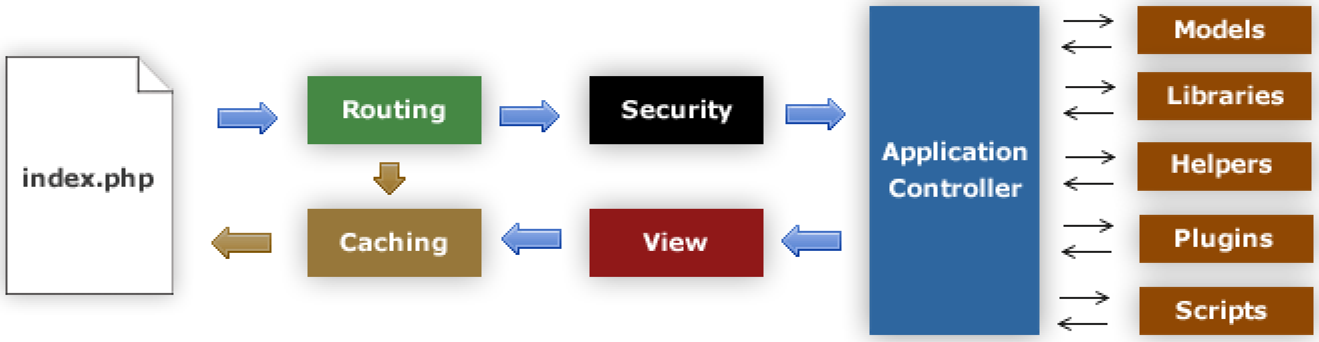


Figure 2-1 CodeIgniter-Application Flow Chart [10]

The “index.php” file serves as the front controller which initializes the base resources needed to run CodeIgniter [10]. The Router examines the HTTP request from “index.php” to determine the action to be taken and if a cache file exists, it is sent directly to the browser. Before the application controller is loaded, the HTTP request and any user submitted data is filtered for security. The Controller loads the model, core libraries, helpers, and any other resources needed to process the specific request. The finalized View is sent to the web browser and if caching is enabled, the view is cached first so that on subsequent requests it can be served.

2.1.2 Model-View-Controller

CodeIgniter is based on the Model-View-Controller development pattern. MVC is a software approach that separates application logic from presentation. In practice, it permits the web pages to contain minimal scripting since the presentation is separate from the PHP scripting.

- The Model represents the data structures. Typically, the model classes will contain functions that help retrieve, insert, and update information in our database.
- The View is the information that is being presented to a user. A View will normally be a web page, but in CodeIgniter, a view can also be a page fragment like a header or footer. It can also be an RSS page, or any other type of “page”.
- The Controller serves as an intermediary between the Model, the View, and any other resources needed to process the HTTP request and generate a web page.

CodeIgniter has a fairly loose approach to MVC since Models might not be always required. Based on the system that has to be built, we can ignore models and build the application minimally using Controllers and Views. CodeIgniter also enables us to incorporate existing scripts, or even develop core libraries for the system.

2.1.3 Security

There is always a concern about security when building a website as there are lot of hackers may get into the server through the website. Hacking is regularly performed by automated scripts written to scour the Internet in an attempt to exploit known website security issues in software. The security of the website can be breached by using attacks like XSS, SQL injection, CSRT, etc. These three are the most common attacks which can compromise a website. However, CodeIgniter framework provides a security class which has methods to tackle these attacks thereby ensuring the security of the website.

2.1.3.1 XSS Filtering

XSS means cross-site scripting. CodeIgniter comes with XSS filtering security. This filter will prevent any malicious JavaScript code or any other code that attempts to hijack cookie and do malicious activities. xss_clean() method is used to filter data through the XSS filter. This method is generally used only while submitting data.

2.1.3.2 SQL Injection

SQL injection is an attack made on database query. CodeIgniter provides inbuilt functions and libraries to prevent this. It follows ways like escaping queries, query biding and active record class.

- Escaping Queries: $this->db->escape() function automatically adds single quotes around the data and determines data type so that it can escape only string data.
- Query Biding: In this method, the query’s parameters are filled with question marks (?) which are later replaced with required parameters using query() function. The values passed through query() function automatically escape values there by producing safer queries.
- Active Record Class: Using active records, query syntax is generated by each database adapter. It also allows safer queries, since the values escape automatically.

2.1.3.3 Cross-Site Request Forgery (CSRF)

Cross-Site Request Forgery (CSRF) is an attack that forces end users to execute unwanted actions on a web application in which they're currently authenticated. CSRF attacks specifically target state-changing requests, not theft of data, since the attacker has no way to see the response to the forged request. CodeIgniter prevents this attack by enabling it in application/config/config.php file. When we are creating form using form_open() function, it will automatically insert a CSRF as hidden field. We can also manually add the CSRF using theget_csrf_token_name() and get_csrf_hash() function. The get_csrf_token_name() function will return the name of the CSRF and get_csrf_hash() will return the hash value of CSRF.

2.2 Apache Cordova

2.2.1 Web Hybrid App

Web hybrid mobile apps are those which wrapped in a webview with a thin native container used just as the bridge to native. No UI components are provided from the native side, it's simply a wrapper for native-to-webview communication. This kind of apps run as a locally installed app on the device built using HTML5/ JavaScript/ CSS. It is run in a webview(embedded web browser) which can access native API's like camera, contacts, etc.

2.2.2 Apache Cordova

Apache Cordova is an open-source mobile development framework [4]. It allows us to use standard web technologies - HTML5, CSS3, and JavaScript for cross-platform development. Applications execute within wrappers targeted to each platform, and rely on standards-compliant API bindings to access each device's capabilities such as sensors, data, network status, etc. Cordova uses Android SDK and JDK to generate the platform specific file (.apk in our case). What stops a web app from accessing the services of a mobile device is simply that the browser sandboxes it for security reasons. All Cordova PhoneGap does is remove the sandbox and provides a JavaScript API to access the system components. Figure 2.2 gives a brief idea about several components of Cordova application.

The Cordova-enabled *WebView* provides the application with its entire user interface. On some platforms, it can also be a component within a larger, hybrid application that mixes the WebView with native application components.

The *WebApp* is the part where the application code resides. The application itself is implemented as a web page, by default a local file named *index.html*, that references CSS, JavaScript, images, media files, or other resources are necessary for it to run. The app executes in a *WebView* within the native application wrapper, which can be distributed to app stores. This container has a very crucial file –config.xml file that provides information about the app and specifies parameters affecting how it works, such as whether it responds to orientation shifts. *Plugins* are an integral part of the Cordova ecosystem. They provide an interface for Cordova and native components to communicate with each other and bindings to standard device APIs. This enables us to invoke native code from JavaScript.

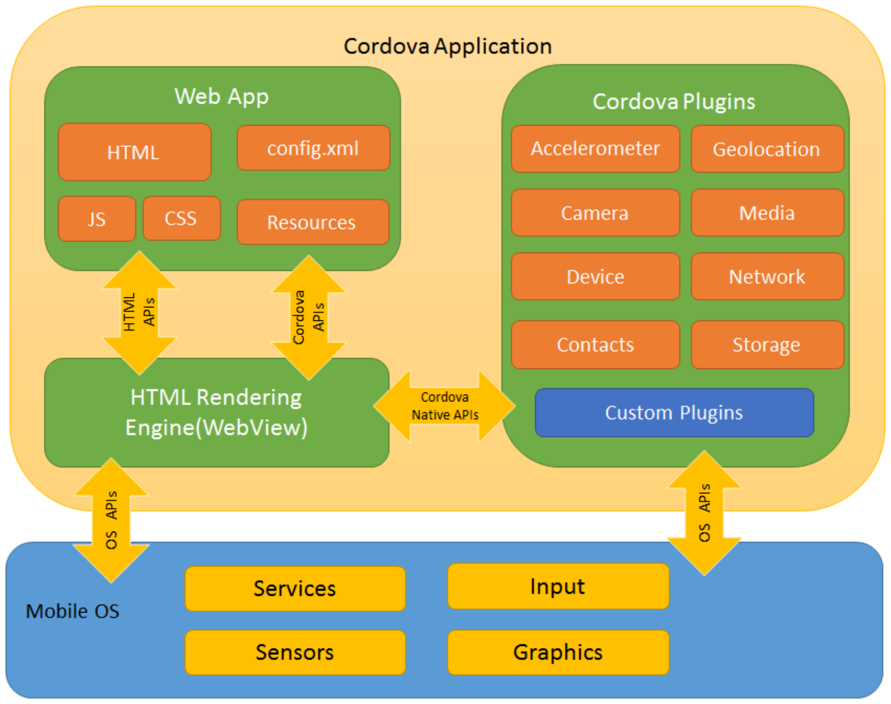


Figure 2-2 Cordova Application Architecture [4]

Chapter 3 - Requirements and System Design

3.1 Requirements

Requirement gathering is a very important part of software development. This phase plays a key role in what we are going to build. This is as important as developing the actual system. Unless a set of concrete requirements are available, the developer cannot get a clear idea about his/her approach to develop the system. To gather the requirements of this system, I initially thought about how a new user would face problems in a new place. With this thought in mind, I have defined the modules in the system and their expectations. The user would fetch data i.e. would only read data from the database while there should be someone who would provide data, that can be our Admin. The requirements gathered with help of Dr. Mitch Neilsen for these two modules are defined below:

The requirements of the user:

- User should be able to see KSU map by default.
- User should be able to search for buildings in KSU while the app predicts what the user is looking for.
- User should be able to see red markers on buildings where there are events on the present day.
- User should be able to check the events on that day by clicking on the red marker.
- User should be able to check all the events in a list format.
- User should have the flexibility to check his current location.

The requirements of the Admin are:

- Admin should be protected account i.e. should be able to access through username and password.

- Admin should be able to give access to new users to add locations/buildings or events.
- Admin should be able to add locations based on latitudes and longitudes.
- Admin should be able to modify location parameters.
- Admin should be able to add/modify events based on time and date to the buildings already stored in the database.
- Admin should be able to add events only to the buildings which are in the database.

It is required that the devices in which this system is implemented is connected to internet and geolocation.

3.2 System Design

A picture is worth a thousand words [6], this absolutely fits while discussing about system design. System design is a method of defining the modules, functions, etc., in a pictorial representation such that it satisfies the requirements. System design is dependent on the specified requirements. In a nut shell, system design is a graphical representation of requirements document which can be built using UML diagrams.

3.2.1 UML Diagrams

UML stands for Unified Modeling Language. UML diagrams represent both a static view and a dynamic view of the system. The static view focuses on the objects, attributes, operations and relationships. Structure diagrams show the static view of the system. The dynamic view focuses on the dynamic behavior of the system. It concentrates on collaborations among objects and various changes in the internal states of the objects. Behavior diagrams show the dynamic view of the system. Following UML diagrams give a brief idea about the present system.

3.2.1.1 Class Diagram

The class diagram is a static diagram otherwise known as structural diagram[7]. It represents the static view of an application. Class diagram is not only used for visualizing, describing and documenting different aspects of a system but also for constructing executable code of the software application. The class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object oriented systems because they are the only UML diagrams which can be mapped directly with object oriented languages. The class diagram shows a collection of classes, interfaces, associations, collaborations and constraints. The following figure represents the class diagram of this system. Since the system is developed in MVC architecture, the model, view and controller classes are represented separately along with their interactions.

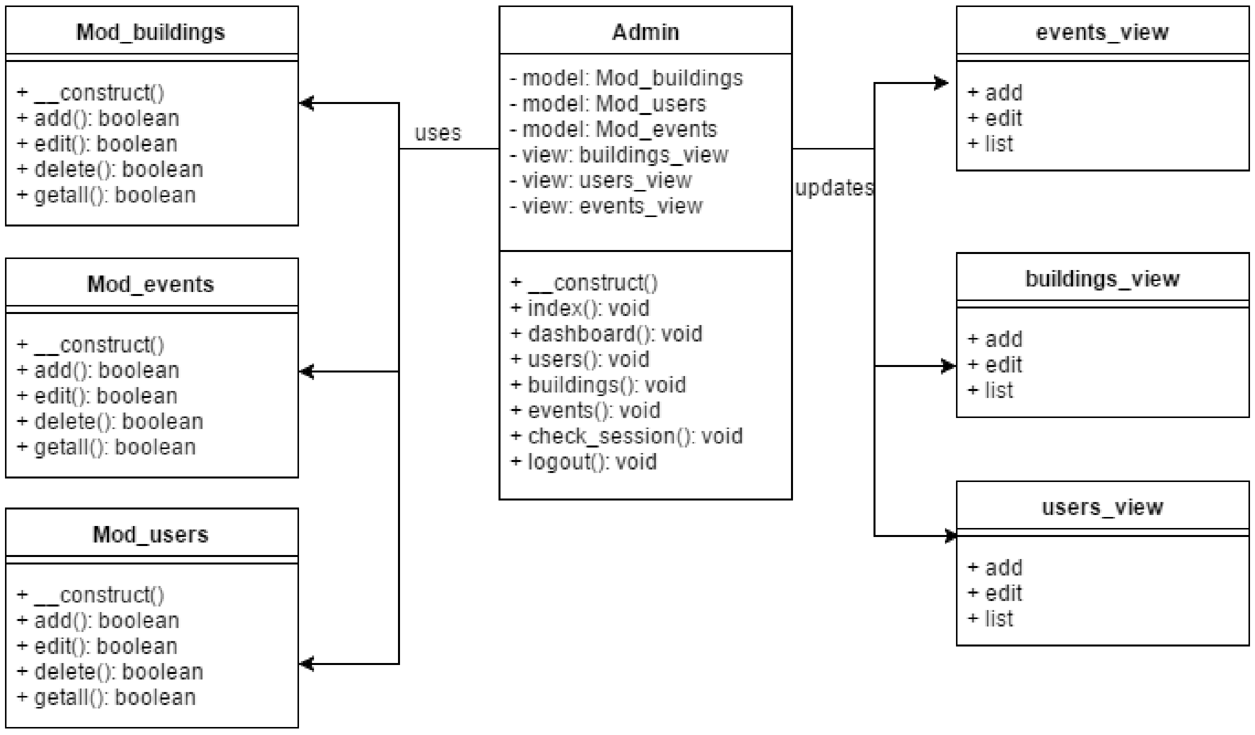


Figure 3-1 Class Diagram

3.2.1.2 Use Case Diagram

To model a system the most important aspect is to capture the dynamic behavior [8]. To clarify a bit in details, dynamic behavior means the behavior of the system when it is running /operating.

In UML there are five diagrams available to model dynamic nature and use case diagram is one of them. Use case diagrams are consists of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. The purpose of use case diagram is to capture the dynamic aspect of a system. The following diagram represents the use case diagram of this system where Admin and User are defined as actors and the operations they can perform are defined as use cases.

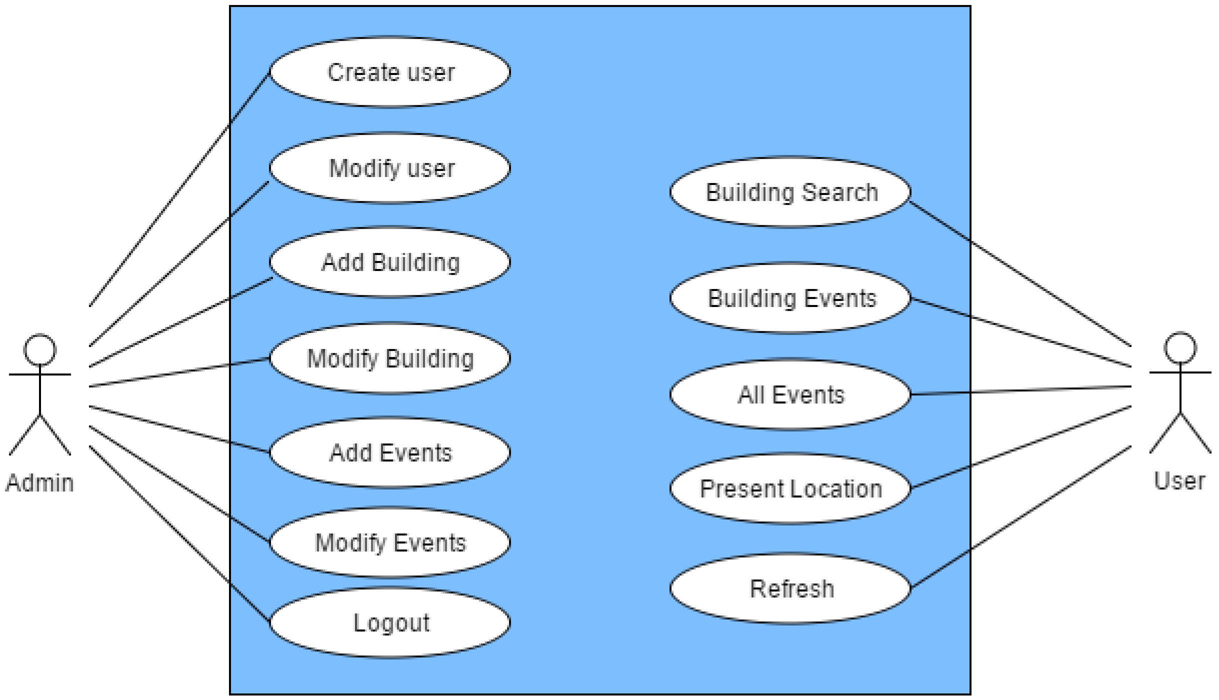


Figure 3-2 Use Case Diagram

3.2.1.3 Activity Diagram

Activity diagram is another important diagram in UML to describe dynamic aspects of the system [9]. Activity diagram is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent. Activity diagrams deals with all type of flow control by using different elements like fork, join etc. The activity diagram in this system is represented as shown in Figure 3-2. The activities are the operations that the system can perform in Admin module and User module.

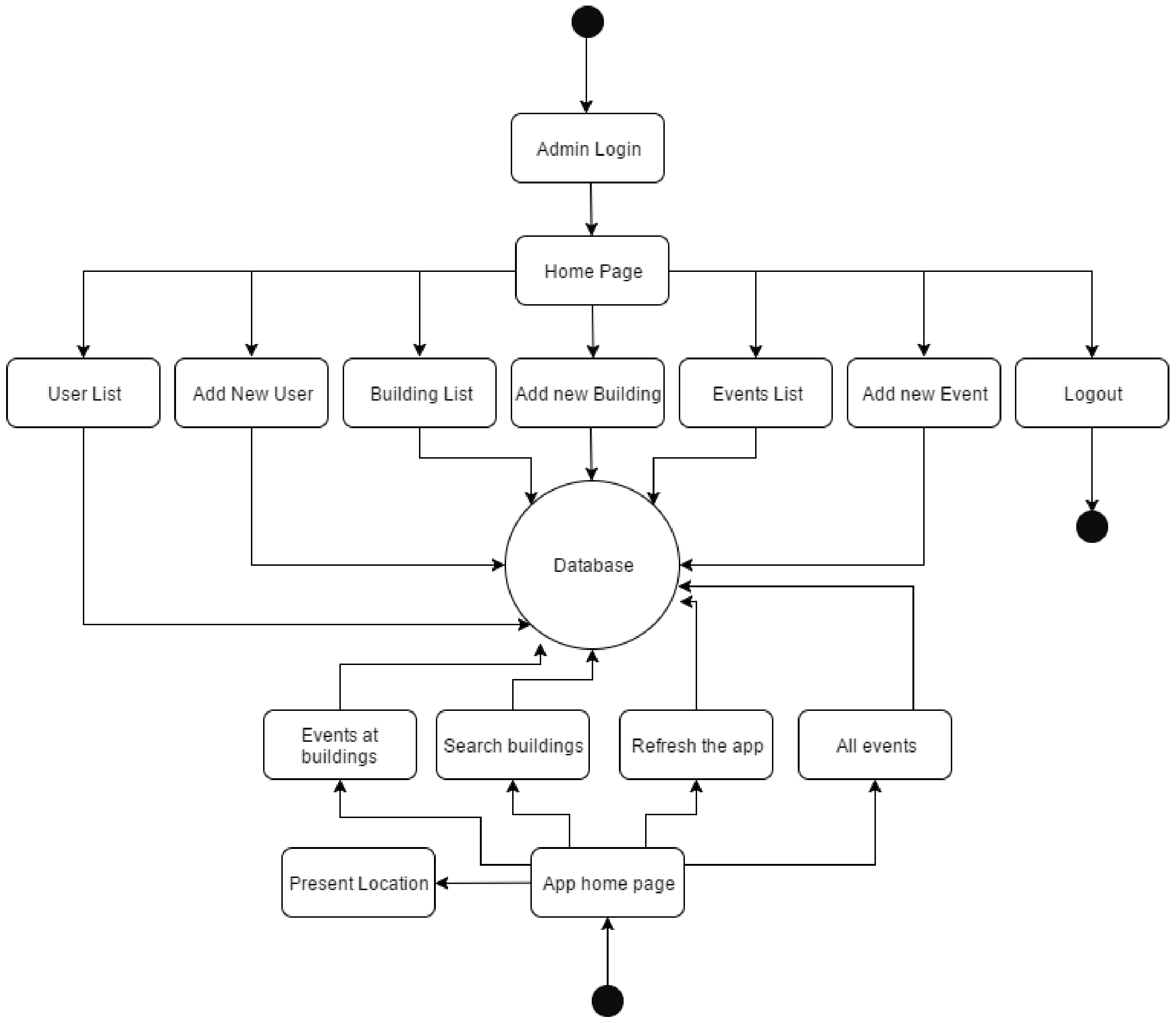


Figure 3-3 Activity Diagram

Chapter 4 - Building the System

This system is built in two different modules. Even though the user module is a mobile application, it is a hybrid app and is coded with jQuery mobile. Both the modules require extensive coding in web scripting languages like HTML5, JavaScript, CSS and PHP. The Admin module consists of a website which communicates with the database which in turn sends data to the mobile application. The system is built in such a way that the website is given read/write permissions over database and the mobile app has only read permissions. Below is a detailed explanation about each of these modules:

4.1 Admin Module

Admin module should provide all the functionalities for an organization to save location names, update events and also give authorization to specific users. To achieve this, all the data should be stored in a database. I chose MySQL for Data Management. HTML for content and structure, CSS for style and presentation, JavaScript for client side scripting and PHP for server side scripting. All these operations are built on a single framework called CodeIgniter. CodeIgniter follows a MVC approach which is explained further in detail.

4.1.1 Database

Based on the requirement, this module should have three entities namely Users, Buildings and Events. Users will have data about the users who are allowed to modify the database but are not related to buildings. This table specifically stores the users who can log in to the system. An E/R diagram shows the entities, attributes and their relation in a schema and can be later converted to a relational schema to maintain the project database. Below diagram shows the entity relationship between them.

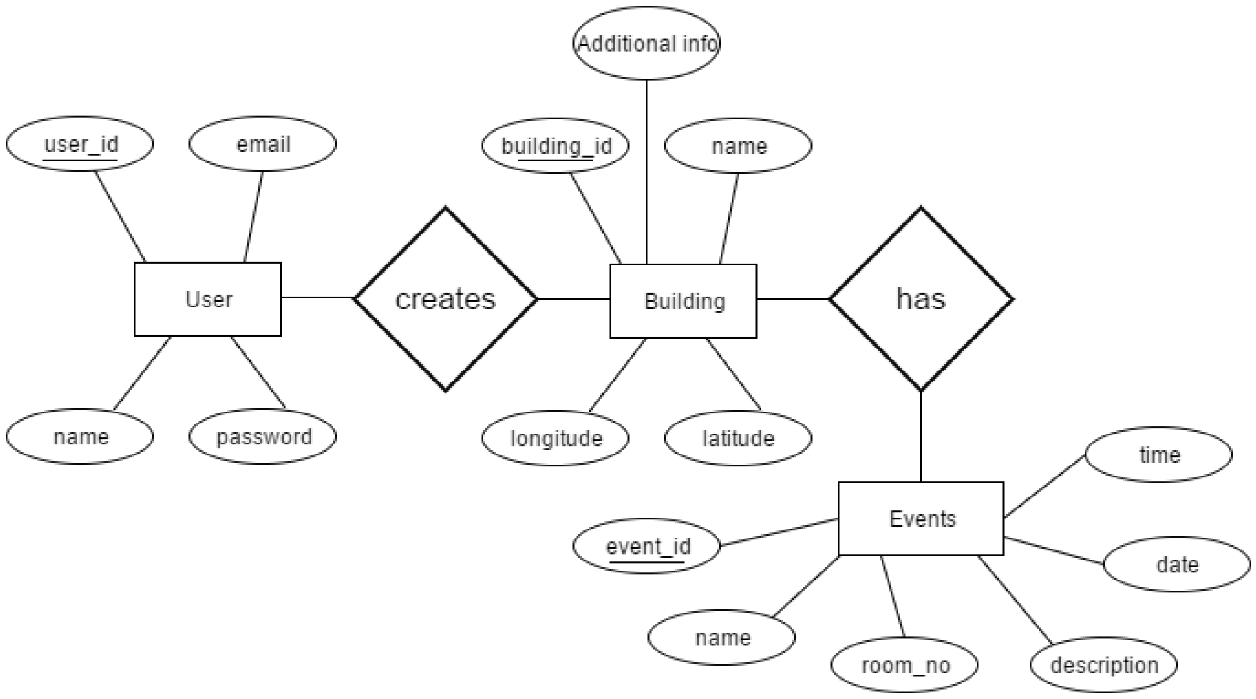


Figure 4-1 E/R diagram

This relational schema is maintained in MySQL database and is accessed through the model section of CodeIgniter. The relational schema is built aligning to the requirements, such that the events and buildings are related whereas the users table specifically stores user data who can use the system. Below relational schema gives a brief idea,

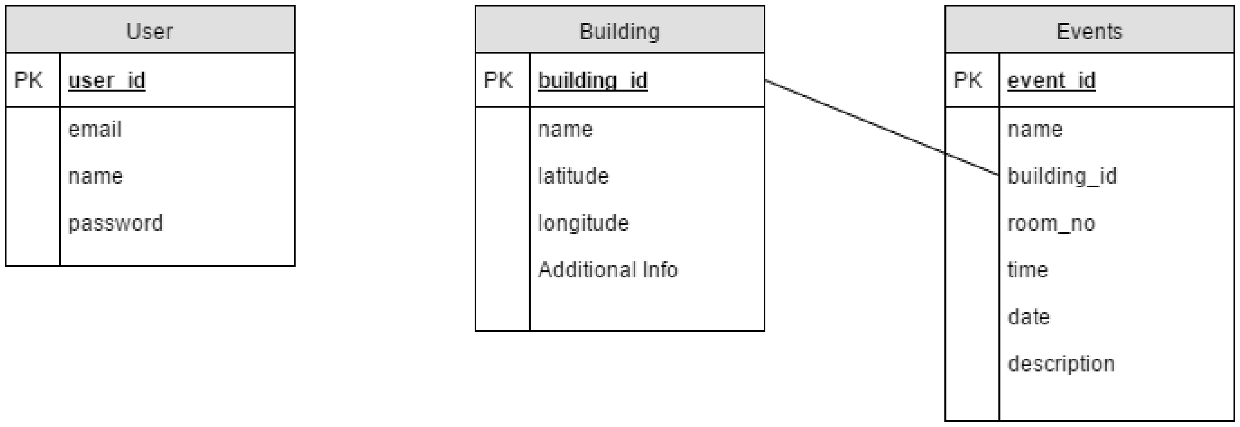


Figure 4-2 Relational Schema

4.1.2 Model

The model section is where the logic resides to communicate with the database and also update the view thereby making them visible to the user. Models are built for *Buildings, Events and Users* typically perform add, delete and getall data functions. In case of user model, an additional function for login validation is also present. Model is also responsible for the view to get updated based on user request coming through controller. All this logic is coded in PHP. This part of the code is in */application/models folder of CodeIgniter template.

4.1.3 View

The web pages that are visible to the user are all coded in view section. Typically there will be web pages for *User, Buildings and Events* section individually. Within these sections, each section would have a webpage to add new entry and also to view all the entries already present in the database. This part of the code is done in HTML, CSS and JavaScript. This part of the code is in */application/views/en/default/ folder of CodeIgniter template.

4.1.4 Controller

The Controller acts as an intermediary between model and view. The controller function is triggered whenever the user performs an action in the view. Controller thereby sends the information to model which interacts with the database and sends the information back to controller which in turn updates the view. Controlling functions are developed in PHP for *Users, Buildings and Events* which will triggered when user performs operations like add, delete, etc. This part of the code is in */application/controllers folder of CodeIgniter template.

4.2 User Module

The user module is a hybrid mobile application which is built using Cordova PhoneGap and jQuery mobile scripting language. The main purpose of this app is,

- To show the present day events at a particular building.
- Search the buildings he is looking for in google maps.
- Get a list of events that are going to happen.

All these is achieved using the google maps API and jQuery mobile scripting.

4.2.1. Google Maps API

The main section of the app is google maps which is first initialized with features like ‘My Location’ and ‘Refresh’. Map is initialized to a particular location by giving the default latitude and longitude as below,

```
<script async defer src="https://maps.googleapis.com/maps/api/js?key=AIzaSyAzQYXYGrK-
9Wv289cM5XGIXl0SH24ZNlg &callback=initMap" type="text/javascript"></script>
        function initMap() {
         var map = new google.maps.Map(document.getElementById('map'), {
          zoom: 8,
          center: new google.maps.LatLng(39.183608, -96.571669),
          mapTypeId: google.maps.MapTypeId.ROADMAP
        });
```

After the map initialization, the main focus was to show marker’s on all the locations where there are events on the present day. For this, a marker is initialized on all such locations. In addition to this, the marker should have an event listener when it is clicked such that it shows the events at that location. A sample of that code is below,

```
google.maps.event.addListener(marker, 'click', (function(marker, i) {
  return function() {
    infowindow.setContent(description);
    infowindow.open(map, marker);
  }
})(marker));
```

This mobile app receives data from the database through AJAX calls in json format.

```
var request = $.ajax({
   type: "POST",
   url: "http://creativeignite.com/samad/event_project/index.php/admin/get_data",
   dataType:'json'
 });
```

The Cordova plugin for geolocation [5] is used to get the present location of the user which is added to the framework before using it. Below is the sample,

```
navigator.geolocation.watchPosition(onSuccess, onError, { timeout: 5000, enableHighAccuracy:
true  });
```

4.2.3 JQuery Mobile – App Layout

JQuery mobile allows us to structure the app as we wish to. The Google map takes the center stage along with which the below components are embedded,

1. Current Location
2. Refresh
3. Search box
4. Clear search
5. Events page

The above components are bundled in a single page using jQuery Mobile. The current location gives the user’s present location. Refresh button would check if there are any updated events. Search box is to search for any location which has an autocomplete feature. Clear search would clear the markers from the map. Events page displays all the present and future events.

Chapter 5 - Using the System

As the requirements demands from the systems is used in two different modules and the target audience for each module are different. The first module, Admin module is targeted to be used by the administrator/organization whereas the second module is targeted to be used by the employees/students, etc. Below is a brief description about how to use both these modules,

5.1 Admin Module

The admin module will be able to perform all the operations that an administrator does, from logging in to adding events. Below are the screenshots of the webpages after developing. First and foremost is the login page through which a user can enter into the system to modify the database.

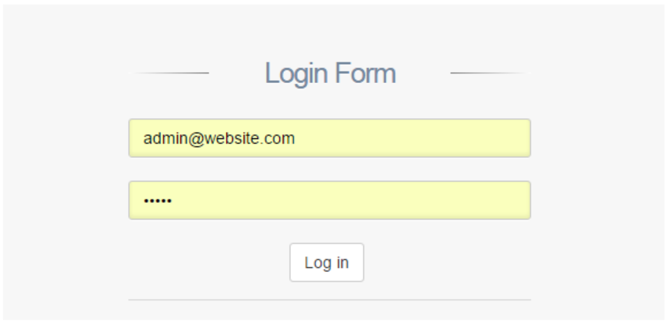
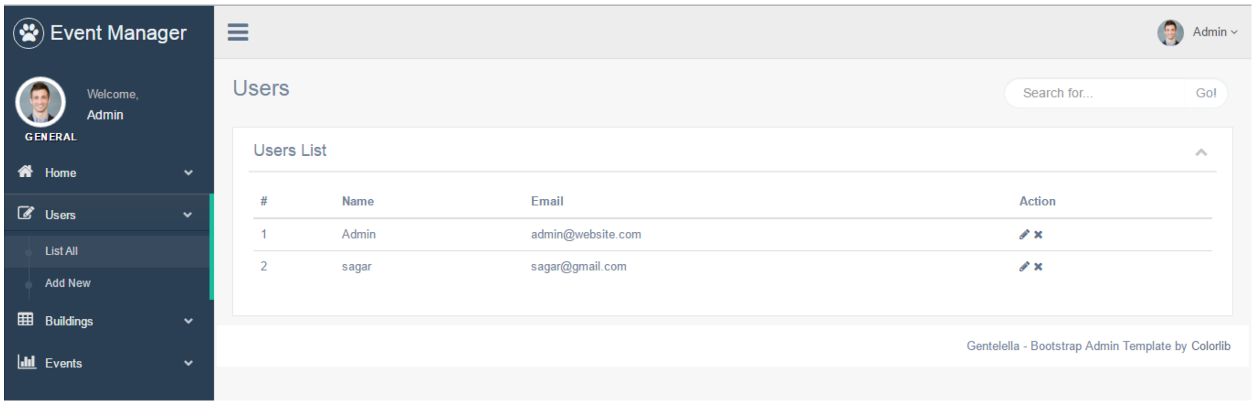


Figure 5-1 Login Page

5.1.1 Users

Users can be added to database or listed from the database in this section. User has fields like username and password. If a user has to be authorized to access this system, the admin should add his details through this web page. Figure 5-2 shows how the user details can be updated or deleted from the database from Action column.



Users

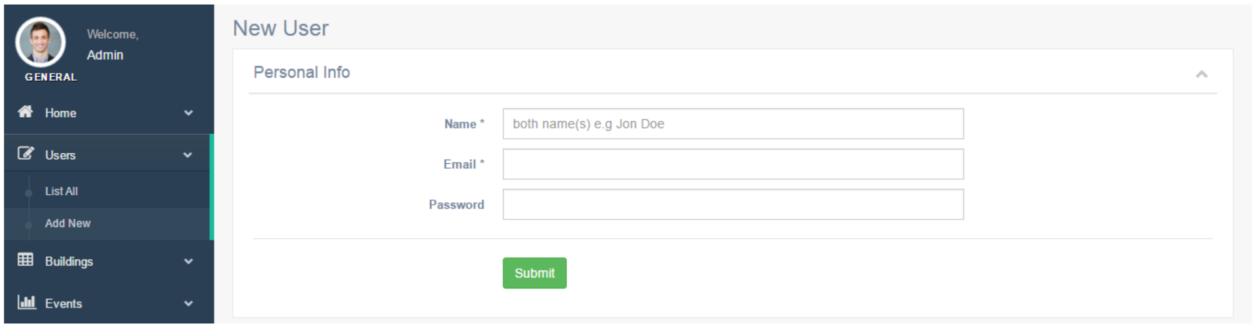
Users List

#	Name	Email	Action
1	Admin	admin@website.com	✎ ✖
2	sagar	sagar@gmail.com	✎ ✖

Gentelella - Bootstrap Admin Template by Colorlib

Figure 5-2 User List

New user can be added to the database from the Users > Add new section as shown in the below screenshot.



New User

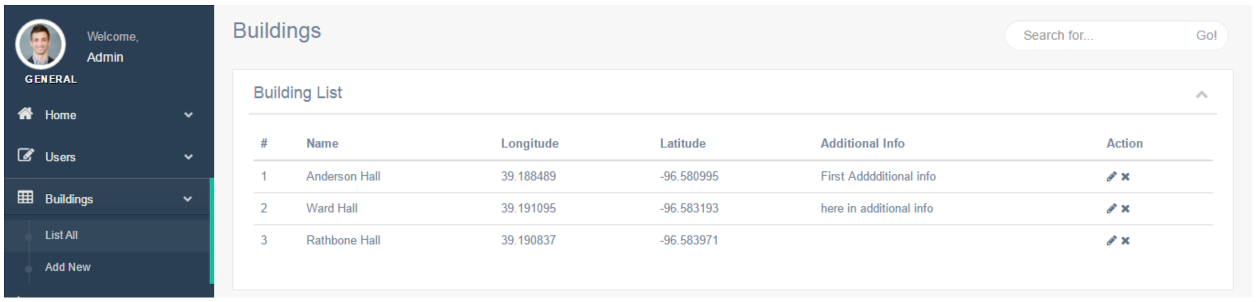
Personal Info

Name *	both name(s) e.g Jon Doe
Email *	
Password	

Figure 5-3 Adding New User

5.1.2 Buildings

Buildings are locations which can be stored with latitudes and longitudes. New buildings can be added or all the buildings can be listed out in this section. Figure 5-4 shows how the buildings can be modified or deleted from the database through Action column.



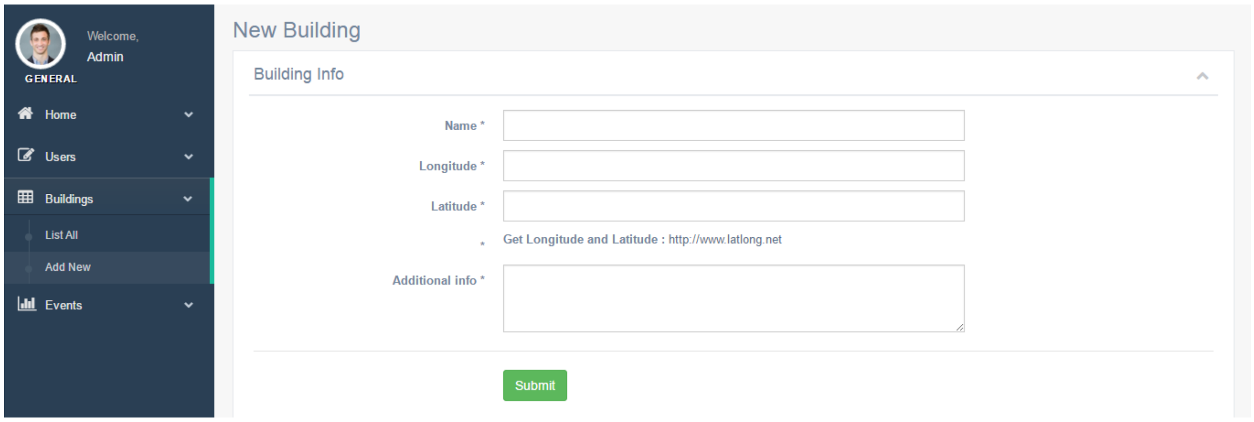
Buildings

Building List

#	Name	Longitude	Latitude	Additional Info	Action
1	Anderson Hall	39.188489	-96.580995	First Addddditional info	✎ ✖
2	Ward Hall	39.191095	-96.583193	here in additional info	✎ ✖
3	Rathbone Hall	39.190837	-96.583971		✎ ✖

Figure 5-4 Building List

New buildings can be added to database from Buildings > Add New section which is shown in the below screenshot. Latitude and Longitude can be picked from the given external website.

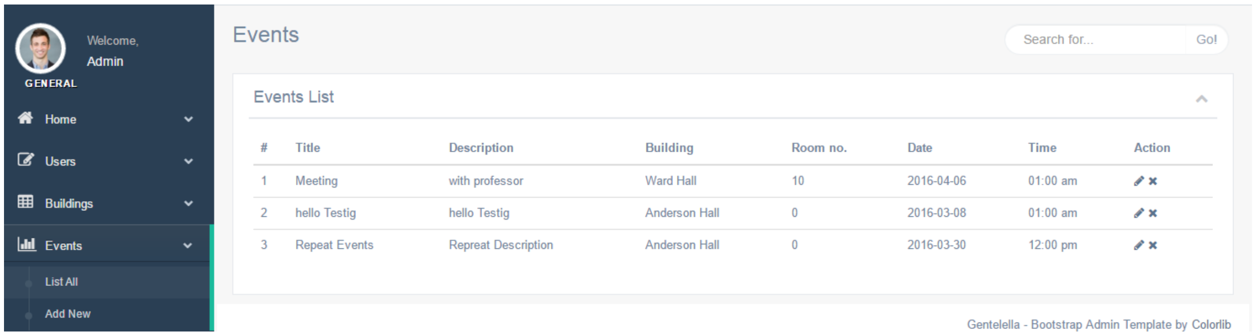


Name *	
Longitude *	
Latitude *	
	Get Longitude and Latitude : http://www.latlong.net
Additional info *	

Figure 5-5 Adding New Building

5.1.3 Events

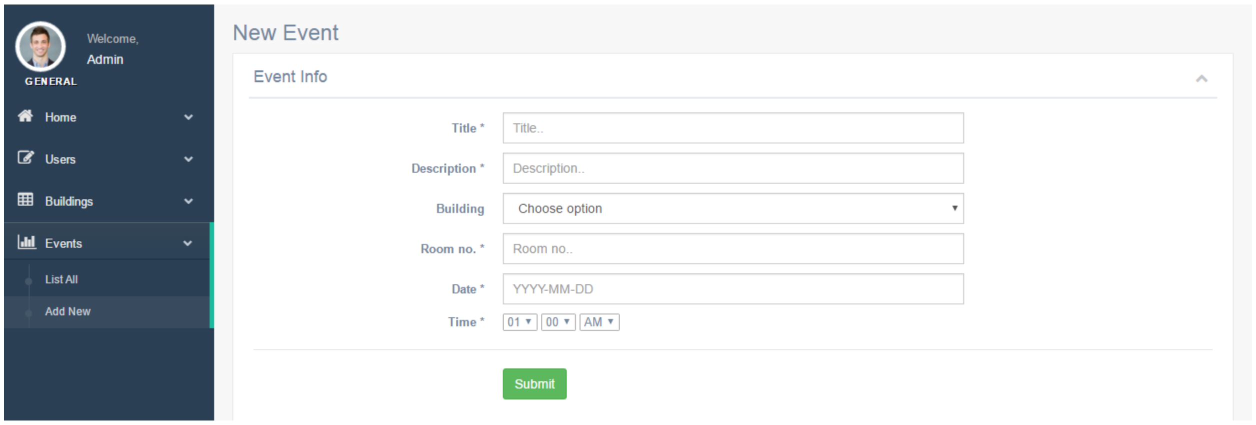
Events section is where events can be added to buildings based on date and time. Events can be added at each of the buildings present in the database along with its date and time. A short description and room number also can be added. Figure 5-6 shows how the events can be modified or deleted from the database using Action column.



#	Title	Description	Building	Room no.	Date	Time	Action
1	Meeting	with professor	Ward Hall	10	2016-04-06	01:00 am	✎ ✖
2	hello Testig	hello Testig	Anderson Hall	0	2016-03-08	01:00 am	✎ ✖
3	Repeat Events	Repreat Description	Anderson Hall	0	2016-03-30	12:00 pm	✎ ✖

Figure 5-6 Events List

Figure 5-7 shows how events can be added to database. An event has title, description, the building which can be selected from the dropdown, room number, date and time. New entries which we give here will be updated in the database which in turn would be visible in the mobile app.



New Event

Event Info

Title *	Title..
Description *	Description..
Building	Choose option
Room no. *	Room no..
Date *	YYYY-MM-DD
Time *	01 00 AM

Figure 5-7 Add New Event

After the necessary operations are performed, the user can logout of the account as shown in figure 5-8. Admin even has the flexibility to search with in the fetched list through the search box provided.

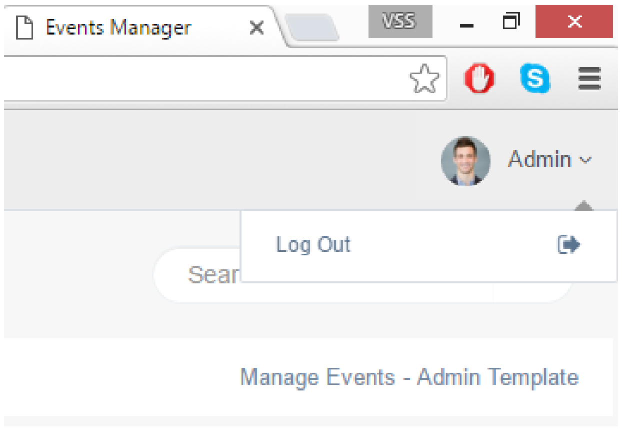


Figure 5-8 Logout

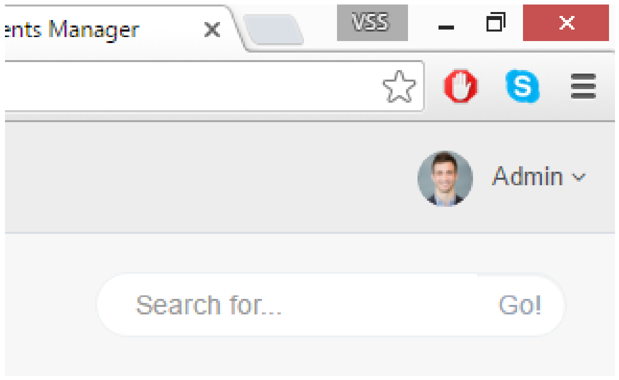


Figure 5-9 Search box

5.2 User Module

The mobile application when installed asks the user for permission to access internet and location which will be necessary to use this application. By default, the app looks like Figure 5-10. The home screen has a search box to search for buildings, a clear search button to clear any searched building, a 'my location' button and a refresh button.

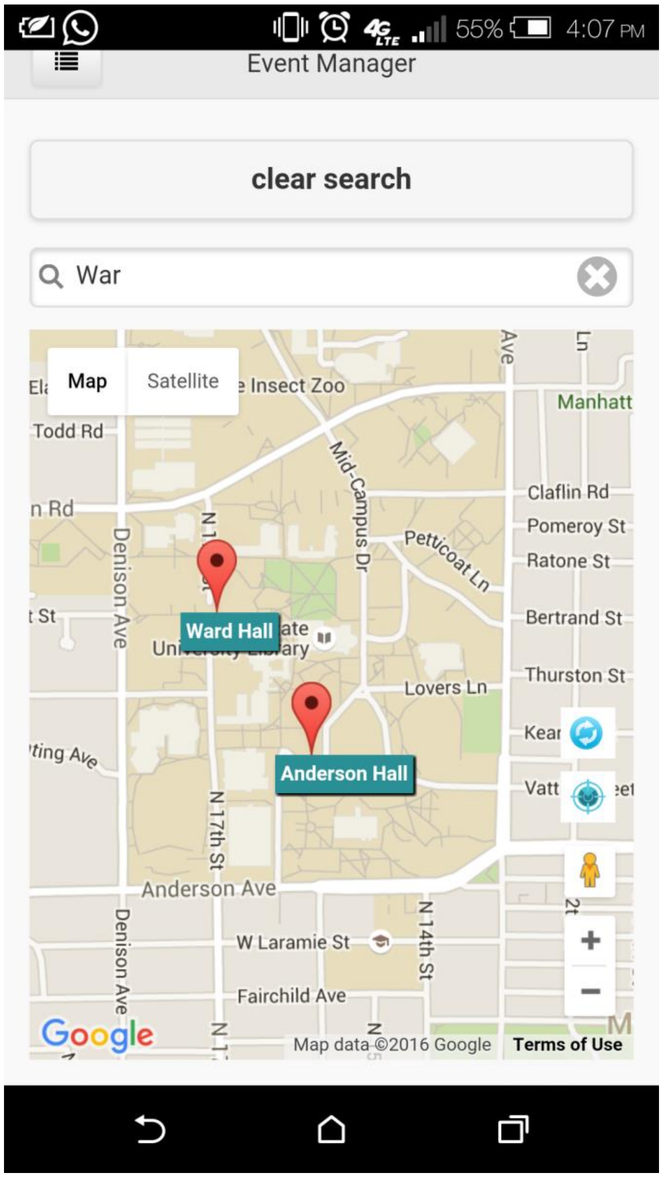
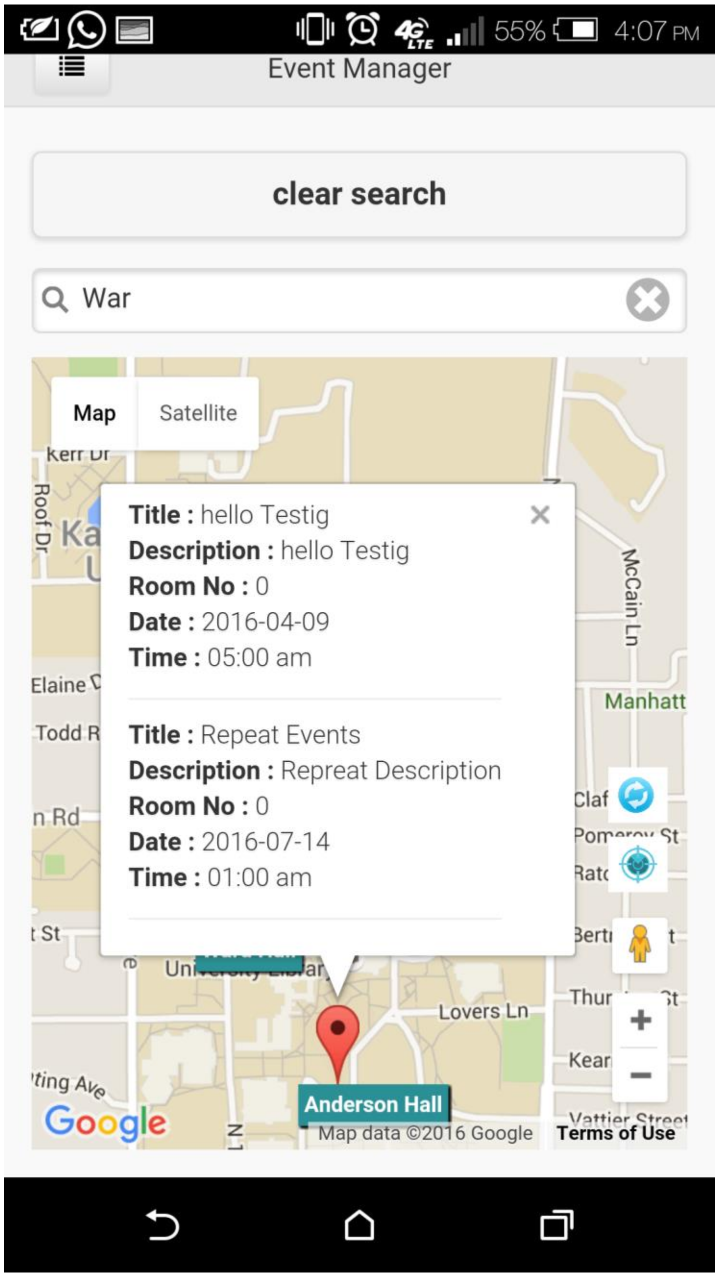


Figure 5-10 App Home Screen

The red markers which are visible by default are the places where there are events on that particular day. These markers are clickable and when done so, will display the list of events at that particular location as shown in Figure 5-11.



Title : hello Testig
Description : hello Testig
Room No : 0
Date : 2016-04-09
Time : 05:00 am

Title : Repeat Events
Description : Repreat Description
Room No : 0
Date : 2016-07-14
Time : 01:00 am

Figure 5-11 Events at a Building

The user can even search for a particular location he is looking for through the search box provided above the map. The app predicts the location you are looking for and suggests names accordingly. The building user is looking for will be viewed as a green marker. User can even look for events at this building by clicking on the green marker.

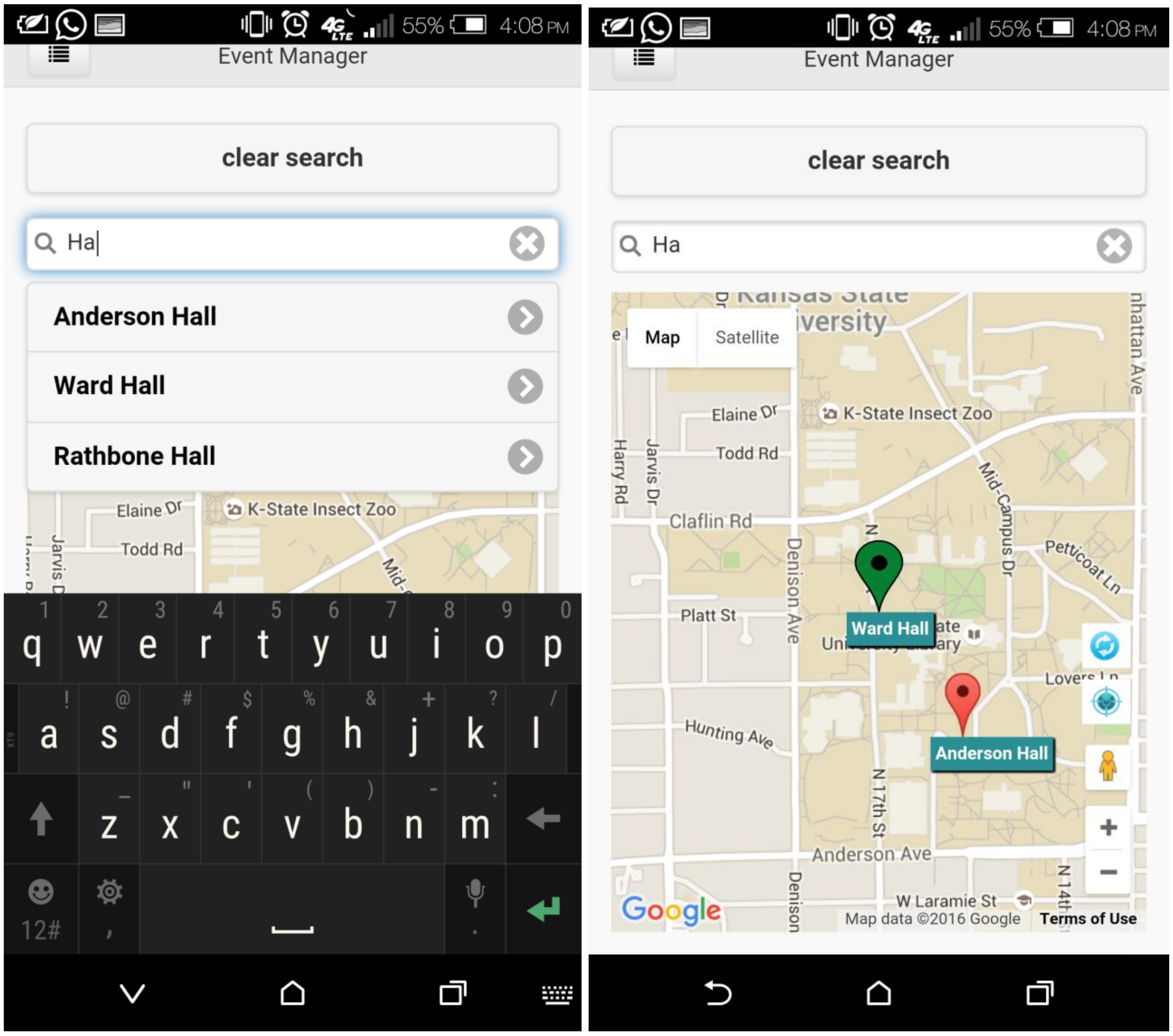
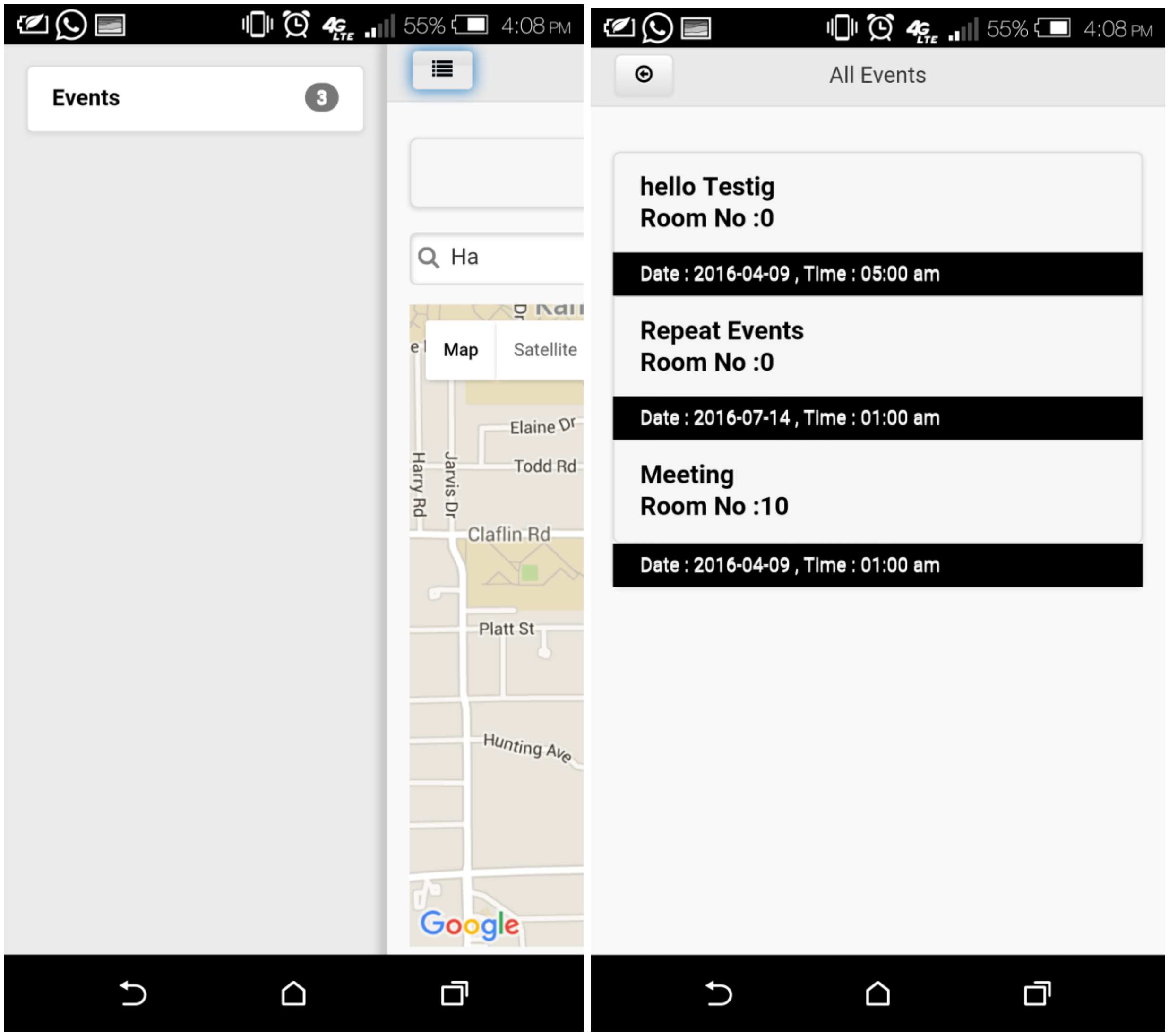


Figure 5-12 Building Search

If the user wants to check for all the events that are going to happen, he can simply click on the left top corner which will open a new page where all the future events are listed as shown in the Figure 5-13.



hello Testig
Room No :0
Date : 2016-04-09 , Time : 05:00 am

Repeat Events
Room No :0
Date : 2016-07-14 , Time : 01:00 am

Meeting
Room No :10
Date : 2016-04-09 , Time : 01:00 am

Figure 5-13 Events Page

User can even check his location by clicking the blue location button on the map. By doing this, user can check how far he is from the location he is searching for. The user can even refresh the map for new events by clicking the blue refresh button on the map. This feature would allow the user to get the modification of any events to be reflected on the app.

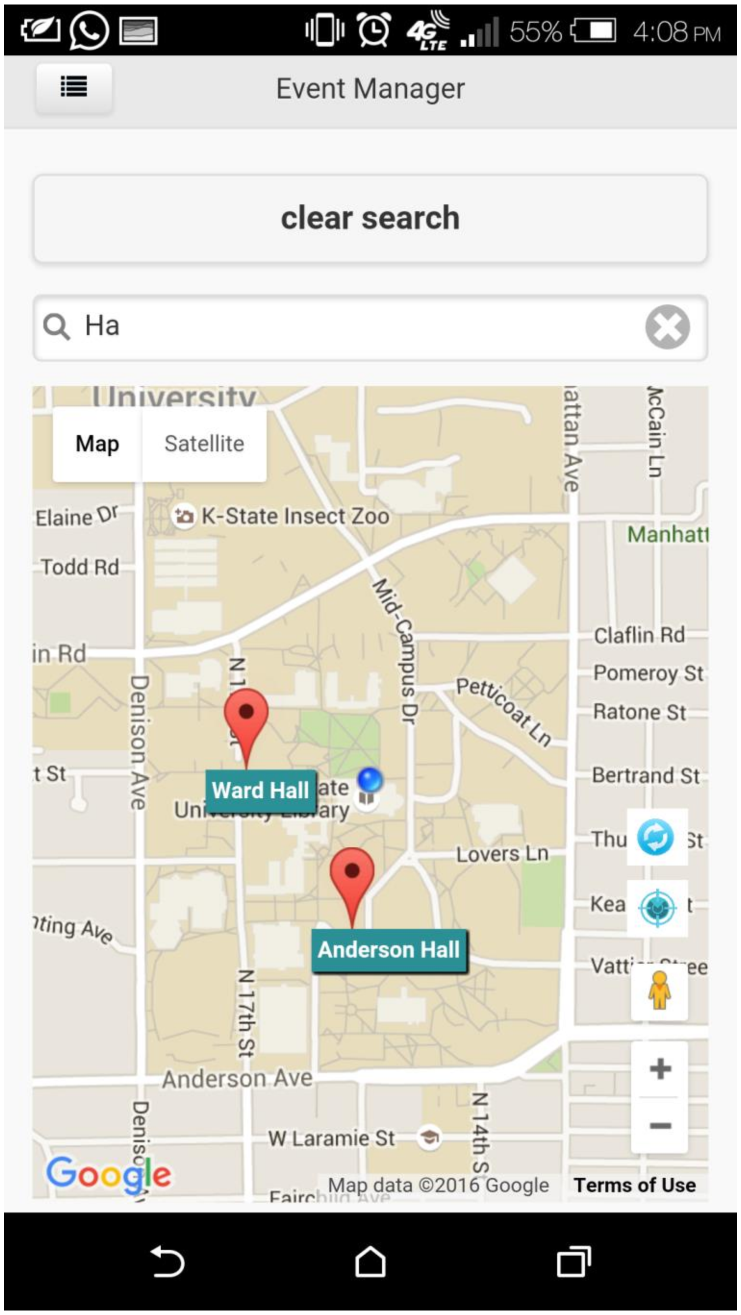


Figure 5-14 Current Location

Chapter 6 - Testing

In the software development process, errors can be injected at any stage during development. Therefore, software testing, which represents the ultimate review of specification, design, and coding, is a critical element of software quality assurance. Testing presents an interesting anomaly for the software engineer. The main objective of testing this system was to uncover any unusual activity the user was not expecting. Since this system consists of two modules, I have opted for Unit testing, Integration testing and Compatibility testing.

6.1 Unit Testing

Unit testing in a testing methodology wherein we test the individual components of the code. In this case, unit testing is performed on the web site before deploying the website using XAMPP server in windows system and also after deploying in google Chrome browser. Following table shows the operations conducted to ensure proper functionality,

S. No.	Test case	Expected Result	Result
1.	Open Login page	Login page should open.	Pass
2.	Admin should be able to login with right credentials	Admin should successfully login	Pass
3.	Admin shouldn’t to able to login with wrong credentials	Admin should not login	Pass
4.	Admin should be able to see all the users	User list should be populated	Pass
5.	Admin should be able to modify/delete a user	Modifications should be reflected in the user list	Pass

6.	Admin should be able to add users	New users should be listed in user list page	Pass
7.	Admin should be able to list all the buildings	All the buildings should be populated	Pass
8.	Admin should be able to modify/delete building details	The modifications should be reflected in list of buildings	Pass
9.	Admin should be able to add new buildings	New buildings should be listed in list of buildings	Pass
10.	Admin should be able to pick latitudes and longitudes from the link provided	Admin should be seeing a new website to pick latitudes and longitudes	Pass
11.	Admin should be able to list all the events	All the events should be populated	Pass
12.	Admin should be able to modify/delete events	Modifications should be reflected in the list of events	Pass
13.	Admin should be able to add new events based on date and time	New events should be visible in the list of events page	Pass
14.	Admin should be able to logout	Admin is redirected to login page	Pass

Table 6-1 Unit testing on Admin Module

The user module is tested by deploying the app in an android phone and the below operations are performed to ensure that all the functionalities are working as expected.

S. No.	Test case	Expected Result	Result
1.	Open the app	The app should load with default screen	Pass

2.	User’s location in the map	User should be able to identify his location	Pass
3.	Red markers on building where there are events	Markers should be visible if there are events at that building on the present day	Pass
4.	Present day events	On clicking the markers, the events at that building should be populated	Pass
5.	No marker for no events building	There should be no marker at a building if there are no events on that day	Pass
6.	Building search	User should get a green marker when he searches for a particular building	Pass
7.	Predict the building user is searching	The app should suggest names based on the text the user is giving	Pass
8.	Search multiple buildings	User should be able to search for multiple buildings	Pass
9.	Clear search	All the green markers should vanish when the user clicks ‘clear search’	Pass
10.	Refresh events	User should be able to refresh for updated events by clicking the refresh button	Pass

11.	List all events	User should be able to list all the future events with its details	Pass
12.	Page transition	User should be able to transit between event list page and map page without any trouble	Pass

Table 6-2 Unit Testing for User Module

6.2 Integration Testing

Integration testing is performed to ensure that all the components are working as expected after integrating all the components together. In this case, the admin module and user module should be able to interact without any problem and the data should be transferred without any glitches. Following tests are performed to ensure that the system is integrated as expected,

S. No.	Test case	Expected Result	Result
1.	New building in website	The app should be able to search for the new building.	Pass
2.	New event in a building	The app should show a marker if an event in added to a building	Pass
3.	Building modification	The marker should change its place based on modified latitude and longitudes	Pass
4.	Event date modification	Event should show marker based on its date	Pass
5.	Event modification	The event modification should be visible on refreshing the app	Pass

Table 6-3 Integration Testing

6.3 Compatibility Testing

Compatibility test is performed to check the devices the built system can work on. The website works on any browser and since the mobile app is a cross platform applications, the application is expected to work in various mobile platforms like android, iOS, windows, etc. Following tests are performed in android and iOS,

S. No.	**Test**	**Result**	
		HTC one M8	**Google Nexus**
1.	Install and Launch	Success	Success
2.	Stress Test	0 issues reported	0 issues reported
3.	Page transitions and button press	0 issues reported	0 issues reported
4.	Screen rotation	Success	Success

Table 6-4 Compatibility Testing

6.4 Assumptions

There are few prerequisites for the system to work without any problem. Following are the assumptions made,

- The Admin module is accessed on a device with internet.
- For the mobile application to run successfully, the device should be GPS enabled and should be connected to internet.

Chapter 7 - Conclusion and Future Work

7.1 Conclusion

In conclusion, this system would help a great deal for users who are new to any place. Any person who is new to a place would looks for places where there are events and where he wants to go. These places might not be available yet in google maps or internet. In a world where each individual owns a smartphone, this app would come in very handy for the users.

Personally, while developing this system, I got to learn about how a database, a website and a mobile application can be integrated. I have learnt technologies like PHP, JavaScript, and JQuery Mobile etc. in depth and also got a very good leaning about frameworks like CodeIgniter and Cordova PhoneGap.

7.2 Future Work

The mobile application can be developed even further by including plugins like Navigation, notification, etc. Navigation would let the user find the way to travel from one point to another within the organization. Notifications plugin would give the user notification beforehand about any event which he wants to attend.

Chapter 8 - References

[1] Mashable (n.d.). *Introduction to CodeIgniter*. Retrieved April 6, 2016, from http://mashable.com/2014/04/04/php-frameworks-build-applications/#xiYcCfdQ3Gqy

[2] I-Programmer (n.d.). *Intro to PhoneGap.* Retrieved April 6, 2016 from http://www.i-programmer.info/programming/mobile/3037-getting-started-with-phonegap.html

[3] CodeIgniter (n.d.). *CodeIgniter User Guide*. Retrieved April 6, 2016 from http://www.codeigniter.com/user_guide/general/welcome.html

[4] Cordova (n.d.). *Cordova overview*. Retrieved April 6, 2016 from https://cordova.apache.org/docs/en/latest/guide/overview/

[5] Cordova (n.d.). *Cordova geolocation plugin.* Retrieved April 7, 2016 from http://cordova.apache.org/docs/en/latest/reference/cordova-plugin-geolocation/index.html

[6] Tutorials point (n.d.). *UML diagram overview*. Retrieved April 8, 2016 from http://www.tutorialspoint.com/uml/uml_overview.htm

[7] Tutorials point (n.d.). *UML Class diagram*. Retrieved April 8, 2016 from http://www.tutorialspoint.com/uml/uml_class_diagram.htm

[8] Tutorials point (n.d.). *UML User Case diagram*. Retrieved April 8, 2016 from http://www.tutorialspoint.com/uml/uml_use_case_diagram.htm

[9] Tutorials point (n.d.). *UML Activity diagram.* Retrieved April 9, 2016 from http://www.tutorialspoint.com/uml/uml_activity_diagram.htm

[10] CodeIgniter (n.d.). *Application Flow Chart*, Retrieved April 6, 2016 from http://www.codeigniter.com/user_guide/overview/appflow.html