DESIGN AND IMPLEMENTATION OF LOCATION BASED
INFORMATIONAL MOBILE APPLICATION

A thesis submitted
To Kent State University
Partial fulfillment of requirements
For the degree of Master of Science

By
Akhil Kumar Goud Koothal
May 2018
© Copyright
All rights reserved
Except for previously published materials
Thesis written by

Akhil Kumar Goud Koothal

B.Tech., Vignana Bharathi Institute of Technology, 2016

M.S., Kent State University, USA, 2018

Approved by

Cheng Chang Lu, Advisor

Javed I. Khan, Chair, Department of Computer Science

James L. Blank, Dean, College of Arts and Sciences
# TABLE OF CONTENTS

**LIST OF FIGURES** 

**ACKNOWLEDGEMENT**

## Chapter 1 INTRODUCTION

1.1 Location Based Web System .............................................. 1

1.2 Requirements ................................................................ 1

1.3 Overview application design and implement .................... 2

## Chapter 2 REQUIREMENTS ANALYSIS

2.1 User Requirement ................................................................. 3

2.1.1 System Environment Requirements ................................. 3

2.1.2 User Characteristics and Assumptions ............................ 3

2.2 Functional Requirements

2.2.1 Use Case Diagram .......................................................... 4

2.2.2 Guest User’s Use Case Descriptions ............................... 6

2.2.2.1 View Public Points ................................................... 6

2.2.3 Faculty User’s Use Case Descriptions ............................. 6

2.2.3.1 Public Points Information ......................................... 6

2.2.3.2 Users Log In .......................................................... 6

2.2.4 Student User’s Use Case Diagram ................................. 6

2.2.5 Admin User’s Use Case Diagram ................................. 7
Chapter 2.3 Architecture of Application

Chapter 2.4 Non-Functional Requirements

Chapter 3 CHOSEN TECHNOLOGIES

3.1 Platform

3.2 Third party Libraries

3.3 Map

3.4 Clustering

3.5 Networking

CHAPTER 4 APPLICATION IMPLEMENTATION

4.1 Development Environment

4.2 Used Design Patterns

4.3 Singleton

4.4 Delegate

Chapter 5 CONCLUSION

5.1 Current Progress

5.2 Future work

REFERENCES
LIST OF FIGURES

Figure 1 Use case diagram 5

Figure 2 Application architecture 7

Figure 3 Application third party libraries 10

Figure 4 Guest view in Application 11

Figure 5 Satellite view in map 12

Figure 6 Clustering Implantation in map 14

Figure 7 Application home screen 16

Figure 8 Forgot password screen 17

Figure 9 Recording user point in map 18

Figure 10 Recording points details screen 19

Figure 11 Validating the API calls in POSTMAN tool 20

Figure 12 Point information in annotation 21

Figure 13 Detailed information of the point 22

Figure 14 sharing information of the point 23

Figure 15 Internship Questions 24

Figure 16 MVC Architecture 27
ACKNOWLEDGEMENT

I would like to thank my thesis advisor Dr. Cheng-Chang Lu of the Computer Science at Kent State University. The professor always helps me out in the difficult situations, He understands the international student situations. I am very thankful to work under the guidance of Dr. Lu.

I also would like to thank Dr. Richard E. Ferdig for giving me this opportunity to join this project. His guides for the project requirements are very precise and very quick paced development allowed us to learn lot of things in a very short span of time. I am really fortune to have this kind of project manager.

I also would like to thank Dr. Austin Melton and Dr. Xiang Lian for the committee and feedbacks that enriched this thesis. I would also like to thank Pavan, Dilip and Sanath for providing a great flexibility of the work environment in the entire project with no hesitation. Sanath and Dilip helped me in giving the Backend service calls(API’s). While Pavan taught me skills required for the development of projects and very supportive in all situations.

Finally, I would like to thank my parents and sisters who support me with all their heart. This accomplishment would not have been without them. Thank you.
Chapter 1

Introduction

1.1 Location Based Mobile System

KSU-CPH is a geo based web and mobile application that provides information about Hospitals, Communities, Student Programs, Non-government organizations for the users. In this mobile application, a member of the CPH or the admin of CPH can contribute geographical related information at a particular point and show them to users. In this geo location is collected and sends data to the system in real time. It is an application to process the geo data and provide information based on that.

1.2 Requirements

LBS features

LBS features denote as ‘Admin’, ‘Faculty, and ‘Student. These three features will make use of GPS from a mobile device at a high frequency level. ‘Admin and ‘Faculty will pull data from REST API, that contains locations coordinate information and locations related information. Data will be pulled from the server, and then cached at a local documented database. Each ‘Faculty ’ or ‘Admin will locate a specific region. Normally, all POIs will be located around trails. Each POI information basically contains longitude, latitude, and radius. Every time users move into the geo-fence for each point, the application will push related notifications to them. There are two approaches to implement distance calculation, which are local based polling calculation and server based request calculation. When users’ motions are faster, the frequency of calculation will increase; if users’ motions are slowing down, the calculation frequency will decrease as well. In the case of users who are standing still, The. If
the application continues to use 3 highly accurate location, although the chips of CPU and GPS in mobile device could continue working well, the efficiency of the battery will decrease.

1.3 Overview application design and implement

This thesis will mainly focus on design and implementation of the mobile application on ios platform. For the implementation, we use objective-c as main development programming language for the main application. The application integrated with third party libraries to realize rapid development. Third-party libraries include, but are not limited to: AFNetworking, SDWebimage, GoogleMaps, etc. On the server end, PHP has been used as the development environment, and JSON has been used as the data exchange format through REST API. Through the third-party libraries which are open sources under MIT or BSD licenses, we could integrate them into the application without restrictions. GoogleMaps provides map visualization and map related features, and it provides highly customized map style, and stable performance. What’s more, GoogleMaps which will provide users with an offline map, when there is no Internet connection. Network library will mainly rely on ‘AFNetworking’, which provides PUT, GET, POST, and DELETE methods.. For interface design, we will implement user interface from a design team, and according to their design wireframe we will build a user interface.
CHAPTER 2
REQUIREMENTS ANALYSIS

In this chapter, the whole application requirements will be introduced completely. Requirement analysis stems from functional requirements and non-functional requirements.

2.1 User Requirement

User requirements are the statements of facts and assumptions that define the expectation of the system regarding the environment, user interface, and user characters.

2.1.1 System Environment Requirements

KSU-CPH is based on the website and mobile application. The software Requirements for mobile app are as following:

- iPhone with at least 9.0 as the version.

2.1.2 User Characteristics and Assumptions

Users occupy a broad spectrum of technical expertise and experiences. Users, however, are assumed to understand the basics of iPhone operation, such as: how to download and update, mobile application on the iPhone, etc. Beyond these basic assumptions, an instructional material will be provided to ensure successful operation.

2.2 Functional Requirements

Functional requirements define what a system is required to accomplish. User cases describe complete interactions between the software and users. There are three kinds of users in this system: admin users, normal registered users, and unregistered users. We analyzed functional requirements using Unified Modeling Language (UML),
2.2.1 Use Case Diagram

A Use Case Diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. Four kinds of users are using the system. The primary roles of each user groups are different. Admin users are usually professional informational specialists. They Show points on main map, Student/Faculty users management and add points, View the data at that point. Guest users are only allowed to view the informational points on the map. Faculty users are allowed to add the informational points assigning the intern questions to the points he added they can even create the normal point on the map. Faculty can add the intern points. Students can view the general guest map points and they can even answer the intern ship points questions create by their professor. Application directly grabs the users coordinates from the location services.users have to give the access permissions to grab the users coordinates.
Use case diagram are shown in Figure 1.
2.2.2 Guest User’s Use Case Descriptions

2.2.2.1 View Public Points

- Allow users to view all the data present on the map. Each location has specific information with their descriptive image as well.
- Precondition: None
- Post conditions: None

2.2.3 Faculty User’s Use Case Descriptions

2.2.3.1 Public Points Information

- Allow users to view all the data present on the map. Each location has specific information with their descriptive image as well.
- Precondition: None
- Post conditions: None

2.2.3.2 Users Log In

- Requirements: Allows a faculty to login
- Post Conditions: User gets a screen where they have 3 different options to go, view guest Map, add internship point and add normal point with category specification.

2.2.4 Student User’s Use Case Diagram

- Allows students to login
- They have 2 options to view the guest map and answer the intern point questions.
2.2.5 Admin User’s Use Case Diagram

- Allows admin to login
- Allows to create a point on the map.

2.3 Architecture of Application

Flow of data in each screen is show in storyboard in the XCode tool. An architecture is been Developed to control the data flow in between the two screens. Here is the picture of data flow.

Application architecture diagram are shown in Figure 2.
2.4 Non-Functional Requirements

Application needs to run on iOS platform. System version should be at least 9.0 to support the application. iPhone/iPod touch/iPad devices will be supported. Supports every screen size including 3.5, 4, 4.7, 5.5. This application is designed to work only on portrait device orientation.
Chapter 3

Chosen Technologies

3.1 Platform

iOS mobile application can be implemented in Objective-C or Swift programming language. In our research project, we have chosen Objective-C as development language, there are two reasons: 1. Lots of stable third parties libraries are using Objective-C as implementation language. Swift and Objective-C mix in one project will cause compiler performance degrades. Therefore, only Objective-C will be used in project. 2. Swift is a kind of interpretation language developed by Apple. Swift is under rapid development in recent years, in case language syntax confliction in future development or maintenance, we will choose stable Objective-C as development language.

Due different devices with needs different screen resolutions, Auto Layout will be used in Application. Auto Layout is a layout system that is descriptive and built on constraints. This simply means that you tell the layout engine how you want the user interface to be laid out. In other words, instead of telling the layout engine that a button needs to be positioned at a specific location, you describe where it should be positioned. Using Auto Layout in application, we can create dynamic and versatile UIs without concern about different size of devices.

Constraints are used to define or describe a layout. For example, you tell the layout engine that a button needs to be horizontally centered in its containing view. The layout engine takes a set of constraints, transforms them into equations, and sets the frames of the elements in the user interface. To make this work, you should add as many constraints to avoid any
3.2 Third Party Libraries

In application, most of UI components are constructed by CocoaTouch frame, some parts are based on third party libraries. Maps displayed part will base on ‘GoogleMaps’, which provides all the features present in google maps. If any features like google directions should be added we need to add ‘GoogleMapsDirections’ it gives all the features of the real time traffic in our application. Based on our requirements we need to import modules of that third party libraries. All the third party libraries used in application are listed below.

Application third party libraries are shown in Figure 3.

```ruby
# Uncomment this line to define a global platform for your project

source 'https://github.com/CocoaPods/Specs.git'

# Ignore all warnings from all pods
#inhibit_all_warnings!

target 'HealthCheck' do
  pod 'MagicalRecord', '-> 2.2'
  pod 'Masonry', '-> 0.5'
  pod 'IQKeyboardManager', '-> 3.2'
  pod 'AFNetworking', '-> 2.5'
  pod 'FastEasyMapping', '-> 0.5'
  pod 'GoogleMaps'
  pod 'NSDateString', '-> 1.0'
  pod 'Reachability'
  pod 'AlertView'
  pod 'MBProgressHUD', '-> 1.0.0'
  pod 'MarqueeLabel'
  pod 'RealReachability'
  pod 'JTFProgressHUD'
  pod 'SDWebImage'
  pod 'SCLAlertView-Objective-C', :git => 'git@github.com:changelog/SCLAlertView.git', :branch => 'develop'
  pod 'SDFraggableDialogView', :git => 'git@github.com:changelog/SDFraggableDialogView.git', :branch => 'master'
  pod 'DBAttachmentPickerController', '-> 1.1.0'
end

post_install do |installer|
  installer.pods_project.targets.each do |target|
    target.build_configurations.each do |config|
      config.build_settings['CLANG_WARNDocumentoTIONS_COMMENTS'] = 'NO'
    end
  end
end
```
3.3 Map

Map display for user will be a significant part in application. We used Google provide GoogleMap framework for developer. However, GoogleMaps is not sufficient for application requirements we have customized maps pointers. Lots of things are customized in GoogleMaps.

Like we have different modes of displays in GoogleMaps (Satellite, Map View). Here is a clear picture of the GoogleMaps implemented in particular module.

Guest view in Application shown in figure 4.
Satellite view in map show in figure 5
3.4 Clustering

Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters). It is a main task of exploratory data mining, and a common technique for statistical data.

Cluster is an easy map annotation clustering library. This repository uses an efficient method (QuadTree) to aggregate pins into a cluster.
Clustering Implantation in map shown in figure 6.
3.5 Networking
When application needs communication with server, iOS native framework provides a set of methods to access RESTful API. However, those native methods are not concise and not reusable for similar functions. In this application AFNetworking third party framework is used. It's built on top of the Foundation URL Loading System, extending the powerful high-level networking abstractions built into Cocoa. It has a modular architecture with well-designed, feature-rich APIs that are a joy to use. Therefore, AFNetworking is primary networking framework used in application. This framework encapsulates GET/POST/PUT/DELETE networking operations for developers.

Initial login screen has various options like faculty and admin login in the same screen and student login in another screen. If the credentials are correct then we proceed to the next screen else we get an error message from the server. Auto layout featured is been used so that we get the same UI with irrespective of the screen size.

Application home screen shown in figure 7.
Forgot password works with the email id specified and a notification will be sent to your email account to reset the password you have to follow the instructions. Then while login use your new password.
Forgot password screen show in figure 8.

Record a point on the map it has a record button and it directly grabs the user location. First it asks users for permission of the location services. We can add an image to this and the name of the location with the type specified like outreach or partnership. Coordinates are taken directly then click save option to save this point data in the data base.
Recording user point in map show in figure 9.

Recording points details screen in figure 10.
A POST API call of the user to get the users points on the maps each user has the unique userid and according to it the values from the database are retrieved in the form of JSON. We have
location coordinates title of that location point, path for that specific image, we have to parse the image url data. Location type of the user like partnership or outreach. Color of that pointer is also provided from the database.

Validating the API calls in POSTMAN tool shown in figure 11.

After the API call we visualized that data in the app in this way. We have the location coordinates and that location coordinate is of type outreach. We have an image for that point as we can see and the title is also specified. If we click on the information icon we get the detailed description of the point which we can see in other point.
Point information in annotation shown in figure 12.

After clicking the information button we have the detail information of that particular point. Initially we have the image in the image view and we use the UITableView to visualize the JSON data in the repeated cells. The title of that point will be scrolled in the navigation bar and the top right there is a share option we can even share the image using Airdrop, message, what’s app.

Detailed information of the point shown in figure 13.
sharing information of the point shown in figure 14.
Internship questions will be appeared in the student portal they can answer the questions from there itself.
Internship Questions shown in figure 15.
Chapter 4

Application implementation

4.1 Development Environment

iOS mobile application will be developed in macOS with Xcode IDE (Integrated development environment). Server side application will be developed in PHP and deploy in Apache web server. Third party library will be managed by Cocoapod which provides reliable third party libraries dependency. Under development testing process, iPhone simulator will be used. Simulator provides real iPhone or iPod touch application environment, but running on macOS. Simulator can simulate multiple devices, therefore under development we will be able to test mobile application on diversity screen size without real devices. When we need testing location related functions, simulator will provide controllable location simulation. Simulator provides function that allow developer input virtual GPS location by put GPX file as GPS signal input. Postman is a Chrome app which provides full REST API testing. When mobile application communicates with server application, Postman will be used to test API content correction and reachability. TestFlight is Apple test platform for application development team, it provides internal test, Beta test for different application version. In our development, TestFlight platform to distribute our latest developing application to test user.

4.2 Used Design Patterns
The Model View Controller (MVC) design pattern specifies that an application consist of a data model, presentation information, and control information. The pattern requires that each of these be separated into different objects.

MVC is more of an architectural pattern, but not for complete application. MVC mostly relates to the UI / interaction layer of an application. You’re still going to need business logic layer, maybe some service layer and data access layer.

MVC pattern is prefer fit for mobile application development. This pattern make model are reusable, and their interface tend to better defined. Application with MVC is more easily to extent new function without changing related function codes.

- The Model contains only the pure application data, it contains no logic describing how to present the data to a user.

- The View presents the model’s data to the user. The view knows how to access the model’s data, but it does not know what this data means or what the user can do to manipulate it.

- The Controller exists between the view and the model. It listens to events triggered by the view (or another external source) and executes the appropriate reaction to these events. In most cases, the reaction is to call a method on the model.
MVC Architecture shown in figure 16.

4.3 Singleton
Singleton pattern is kind of simple design pattern. This design pattern provides best way to create single instance object in application. LocationService and DatabaseManager class are using singleton design pattern under global environment. So whenever other class needs instance of this two class, there will only one copy existing in system. Only one class instance in memory guarantee that other class get same instance and get same data, for instance same GPS coordinates, or same database reference. Only one class instance guarantee related classes get same value real-time data instead inconsistent data.

4.4 Delegate

Delegation is a kind of simple design pattern which provides one object can behalf of another object. Delegate object will be inserted into delegating object. When delegating object needs some functions, it doesn’t need implement by inside, delegate object will hand over functions. This design pattern is widely used in application. For instance, when application service sending GPS coordinates to other class, there will be a delegate object receive GPS message and perform different actions. Even in UI component parts delegate are still widely used. UITableView are designed in delegation mode, UITableViewDataSource is a delegate needs to be implemented which will determine how much rows and sections in table.

Chapter 5
Conclusion

5.1 Current Progress

In this paper, we present the whole process of how to design and implantation of location-based informational mobile system. In order to build a high-performance map with geo-data in real-time, we design and implemented an effective algorithm to determine how to upload data to the dynamic map in different conditions. By applying the GeoJSON method to save multilevel geo-data, this kind of dataset not only loads data in GogoleMaps API efficiently but also helps to collect complicated user answer data easily and clearly. It is a well-designed object oriented application. The database currently contains more than 200 pointers related to Hospitals, communities, student programs, Internship points and non-government organizations. users can categorize the points by outreach, partnership, circular, alumni and students. Admin can categorize the points by the title, longitude, latitude, approved points, declined points, waiting points.

Moreover, we designed mobile RestAPIs, for the iOS to send and receive the data from server.

iOS mobile application KSU-CPH has been released on TestFlight, some features are still in progress. In TestFlight we are doing the internal testing with the dummy data. Application is working in real time.

All the 4 modules Admin, Guest, Student, Faculty are working in real time.

5.2 Future work

An increasing numbers of Faculty and students in intern points. The faculty cannot show
the same set questions to all the students in every intern point. Many of the questions changes if there exist more number of students. In the future, we will design a system where faculty can create a own set of questions for a specific points and to specific students.

REFERENCES


