北京郵電大學

Queen Mary University of London

Undergraduate Project Report
2018/19

A Geocaching smart phone app

Name:	Wang Ran
School:	International School
Class:	2015215119
QM Student No.:	151006766
BUPT Student No.:	2015213404
Programme:	Internet of Things Engineering

Date: 06-05-2019

Table of Contents

Abstract

The purpose of the project was to develop an Android smart phone app that can assist people to participate in the activity of geocaching and add enjoyment to it. The achieved features of the app mainly included searching and viewing geocaches, posting geocaches, setting geocaches as targets, locating geocaches on a map, and viewing users' history records. One special feature of this app was that users could chat with other users and publish status updates, like a social media platform. The outcomes of the project contained an Android smart phone app, a server, and a database. Amap API was applied to develop the app in Android Studio. RESTful web services were chosen to realize the communication between the app and the server, and the model Spring Boot was used to develop the server. The project also carried out the testing of the system and drew a conclusion on the pros and cons of the app compared with other geocaching apps on the market. Future work could include providing some offline features of the app and presenting the path lines the user traveled.

Keywords: Android smart phone app, geocaching, RESTful web services, Spring Boot

摘要

本项目旨在研发一款能协助人们更好地参与地理寻宝活动的安卓智能手机程序。本项目研发的手机程序主要包含以下几个功能：寻找及查看宝藏、发布藏宝信息、设置宝藏目标、在地图上定位宝藏、以及查看用户历史记录。本项目的特色功能是允许用户和其他用户聊天，并在公开布告栏上发布状态更新。本项目主要有三个成果：安卓手机程序、服务器端程序以及数据库。本项目在 Android Studio 中开发手机程序时使用到了高德地图 API，在开发服务器端程序时应用了 RESTful Web 服务和 Spring Boot 模型。本项目完成了系统测试，并结合市场上已有的地理藏宝手机程序对本项目的成果做了优缺点的总结。未来工作包括提供一些手机程序的线下功能，以及显示用户走过的路径。

关键词：安卓智能手机程序，地理寻宝游戏，RESTful Web 服务，Spring Boot

Chapter 1: Introduction

This project aims to develop an Android smart phone app which can assist users to participate in geocaching (O'Hara, 2008), which is a treasure hunting outdoor activity. A database is needed to organize and manage the data in the project, and a server should be designed to deal with the requests from the app. Web services are used to deliver requests and give responses between the app and the server. AMap API(Application Programming Interface) (Alibaba, 2002) is used to deliver a map in the app and help users navigate to the target geocache during participating this activity. Development tools used in this project are Android Studio 3.2, IntelliJ IDEA 2018.3.3 and MySQL Workbench 6.3. Development language used in this project is Java. RESTful(Representational State Transfer) web services (Vinoski, 2008), Spring Boot (Pivotal Software, Inc., 2019), Spring Data JPA (Java Persistent API) and Hibernate (JavaTpoint, 2011) models are applied when developing the server.

1.1 Motivation and aims

The aim of this project is to develop a smart phone app that can help users to participate in the activity of geocaching more easily and can help users to enjoying the fun of it. The Geocaching smart phone app can be regarded by users as a guide for geocaching, a platform to search and post geocaches, a database to store their history records of posts and findings, a tool to help them navigate to the target geocache, as well as a social media platform to share experience and information. The app should make it more convenient for people to start the activity of geocaching and satisfy people’s needs to communicate and share experience with others.

1.2 Features of the system

1.2.1 User requirements

A Geocaching app should assist users to participate in the activity by offering the following basic functions (Sullivan, 2016):

- Allow users to search for a geocache, view detailed information of geocaches, set one interested geocache as a target and provide users with a map to help them navigate to the target geocache. Allow users to mark the target geocache as found or lost.
- Allow users to post a geocache by providing cache description, location, and hints that could help other users find the geocache more easily. Allow users to save a draft of the cache to be posted and edit it later.

- Allow users to view user information and history records of the posted, found, and failed-to-find geocaches.
- Allow users to register and log in/out.
- Allow users to share and communicate with others to express their thrilled feelings after finding a difficult-to-find geocache or to report a damaged or missing geocache. To implement that, the app should allow users to send messages to a specific user by offering the username or to share their experience and feelings on a public virtual bulletin board.
- Provide an introduction of geocaching and some instructions on how to participant in the activity for novices.

1.2.2 Achieved Functionalities

Users are supposed to register or login the system before they can get into the functional pages of the app. There are 4 items in the bottom navigation bar in the main page and each item refers to a functional page: “History”, “Share”, “Geocaching” and “More”.

On the page “History”, users can view some basic information like username, user level, joining date and the numbers of geocaches posted or set as targets. Users can also view their own history records of posts and findings of the geocaches. On the page “Share”, users are allowed to communicate with other users by exchanging messages or posting status updates onto the public virtual bulletin board to express their feelings and share experience with other users. On the page “Geocaching”, users can search for a geocache, view geocaches in different orders, choose a geocache as a target on the shown map, and use the map as a tool to help them locate to the target. After users finish a journey, they can mark the geocache as found or lost and view this record later on the page “History” when they want to. Users can also post a geocache by providing the basic information like cache ID, cache position, cache description and hints for finding the cache. On the page “More”, users can log out, give feedback to the app, and view introduction of geocaching and suggestions on how to participate in this activity. Users can also edit the draft of the geocache to be posted and then publish them.

1.3 Structure of the report

The report first introduced the motivation and aims of this project, as well as the user requirements and achieved functionalities. The second chapter presented the background of this project, for example, the history and aims of the activity of geocaching, the existing geocaching

apps on the market, and the techniques and technologies utilized in this project, such as GPS, AMap, and web services. The third chapter introduced how the system was designed and how different parts of the system were implemented, such as the app, the server and the database. This chapter also introduced the software, the hardware and the specific map API utilized in the project. On chapter four, testing and the results were presented, as well as the analysis of the developed system. The last chapter drew a conclusion on the whole project, including the purpose, the achieved outcomes, the problems and solutions, and some changes in the project. The future work could include some improvements to the system.

Chapter 2: Background

2.1 Geocaching

2.1.1 History of geocaching

The idea of geocaching was derived from a 160-year-old game named letterboxing. In this game, participants locate to the landmarks by utilizing some hints and clues (Vartiainen et al, 2013). In 2000, the technology of GPS developed rapidly and made a huge progress, which implemented and realized the idea of geocaching. Instead of posting descriptions or clues, the GPS locations of the 'treasures' were posted on the websites, which allowed participants to navigate to the target in a more accurate way with the help of GPS devices.

Geocaching is an outdoor, real-world treasure hunting game, where participants can either choose a published "treasure" as a target to find, or publish a "treasure" with its GPS location. The "treasures" are called geocaches in the activity. Traditional geocaches are containers that can hold a logbook, a pen or pencil, and trade items. Participates who find the geocache can sign the logbook to record the success of this journey, and exchange a trade item in the container.

2.1.2 Aims of geocaching

The aims of geocaching are to enrich people's lives, encourage people to go outside and enjoy the fun of touching with the nature. Most of the geocaches are located in parks, which makes the treasure hunting itself a relaxation and a fun journey. Signing the logbook gives the participants a sense of achievement and accomplishment. Exchanging trade items means that participants can keep a trinket as a souvenir and put in a trinket to get involved. For people who like to hole up at homes and don't have many social activities, geocaching is a good way to start. Additionally, geocaching also helps improve people's skill of reading a map and develop a sense of direction. Geocaching can make people feels involved and encourage participants to communicate and share with others.

2.2 Geocaching Apps

2.2.1 Literature review

The website of Geocaching was established in 2000, in which year the accuracy of GPS technology improved tenfold. A variety of Geocaching apps came to the market in recent years due to the vapid development of the smart phone technologies and the Internet.

Table 1 summaries the common features of four existing geocaching apps, which indicates the important features that a geocaching app should possess. Almost all geocaching apps have the features of searching for geocaches, fetching geocache information to the smart phone, providing a map, and reading or keeping records in the database (The Geocaching Junkie, 2017).

Table 1: Common features of the four existing geocaching apps

Apps / Features	Geocaching	Cachly	Geooh Live	c:geo
iOS / Android	iOS & Android	iOS	Android	Android
Search for a cache	√	√	√	√
Provide a map for navigation	√	√	√	√
View history records	√	√	√	√
Log in and out	√	√	√	√
View description of the geocache	√	√	√	√

Table 2 contains some special features of the existing geocaching apps.

Table 2: Special features of the four existing geocaching apps

Apps / Features	Geocaching	Cachly	Geooh Live	c:geo
Free to use	primary version		√	√
Post a cache	√	√		
Provide instructions for geocaching	√			
View the map without the Internet	√	√		√
Send messages to the publisher of the geocache	√			
Share a cache by email or through other ways	√			√
Remind the caches not found		√	√	
Traffic information shown on a map			√	
Log caches without an internet connection	√	√		√
Provide virtual souvenirs after a cache is found on a specific place or day	√	√		

2.2.2 Pros and cons of the existing geocaching apps

The existing geocaching apps help users participate in the activity of geocaching more conveniently. Geocaching is the official app created by Groundspeak, Inc., which also manages the official website of Geocaching (Groundspeak, Inc., 2000). The app Geocaching allows users to save cache information to the phone, so the users can still view the information they need without the Internet. When navigating to a geocache, Geocaching has a nice compass feature, which shows the current location, the cache location coordinates, and the distance to the geocache. Geocaching also has a help center that gives introduction to this activity and suggestions on how to participate in it. Cachly (Zed Said Studio LLC, 2015) is an app that only works with the iOS operating system, and it has virtual souvenirs for users after they have found a geocache in a specific place or on a special day. To make it more convenient for the users, Cachly provides attributes for each geocache, such as parking information, estimated time consuming and whether dogs are allowed in that area.

However, the existing apps are limited in some ways. Only a few apps have the feature of allowing users to communicate with each other. The official Geocaching app allows users to send messages to a specific user by offering the username. Besides, many apps are not free. Some apps provide a charged premium version that has more powerful functions.

This project intended to develop a free app for geocaching participants without using the official geocaching API provided by Groundspeak, Inc. In addition to fulfilling the basic features to assist users to hide and seek a geocache, this app also intended to encourage users to communicate and share with others by offering two options. Users could add other users as friends by inputting their usernames and then chat with them, the history records of the messages could be presented on the chatting page. Users could also choose to post a statues update to share their feelings and experience on a virtual bulletin board opened to all the users.

2.3 GPS

The Global Positioning System (GPS) is a U.S.-owned system that provides users with three-dimensional position, speed and other information based on 24 positioning satellites on a global scale. The system consists of three segments: the control segment, the space segment, and the user equipment segment. The control segment is responsible for maintaining the satellites in their proper orbits and making adjustment to the satellite clocks. This segment consists of the control station, the antennas, the monitoring station and the communication system. The space

segment has 24 operating satellites distributed in 6 different orbital planes, which are responsible for transmitting one-way signals that could provide position and time to the current GPS satellite. The user equipment segment is the GPS receiver that could receive the signals from the GPS satellites and then present the users' position and time. The positioning accuracy of civil use can reach within 10 meters (NOAA, 2017). In this project, participants should downloaded the app on an Android smart phone that contains a GPS receiver. Only in this case can the map show the positioning of the participant, and can the participant further utilize the map as a tool to navigate to the target geocache.

On May 2, 2000, Selective Availability, a technique to limit the accuracy of GPS deliberately, was removed from the Global Positioning System, which allowed for more accurate positioning of small containers to be placed and located. Geocaching was conceived shortly after the accuracy of GPS technology improved tenfold (Wikipedia, 2019). Participants use devices embedded with GPS receiver to help them locate and find the geocaches.

2.4 Amap

2.4.1 Literature review

Amap (Alibaba, 2002) was developed by the company named Gaode, which was China's leading service provider of digital maps, navigation and positioning services. In February 2014, Alibaba completed an acquisition of Amap.

Amap is a powerful app that contains many functions, such as map rendering, online navigation, AR virtual reality, route planning, real-time broadcast of traffic conditions and taxi services. In this project, Android map and positioning SDK of Amap is downloaded and utilized. Amap provides an open source API(Application Program Interface) platform for developers to utilize the services of Amap. The open platform contains SDK for Android and iOS platform, as well as JS API for web developers. Developers are allowed to use these tools as long as they register the system and apply for a key. Tools and documents of Amap are free to download for developers.

2.5 Web services

A web service is a client and server application typically delivered over HTTP(Hyper Text Transport Protocol), and the components can be deployed and executed on distinct devices. The concept of client in web services is also known as a consumer or a requester. A web service can be hosted on a server and consumed on PCs, smart phones, and other devices. There are two

types of web services: SOAP-based and REST-style (Belqasmi et al, 2012).

2.5.1 SOAP-based web services

SOAP-based web services have standards, toolkits, and abundant software libraries and they are special cases of RESTful web services. SOAP (Simple Object Access Protocol) was designed before REST (Representational State Transfer) to make it simpler for programs built on different platforms and in different programming languages to exchange data. A WSDL (Web Service Description Language) file is included in SOAP to describe the availability of the service. SOAP messages are all in XML (EXtensible Markup Language) format and it requires more bandwidth for transferring a large amount of data in SOAP messages. In a SOAP-based web service, a client sends a request in SOAP document transparently, and the web service will return another SOAP document as a response.

2.5.2 RESTful web services

A web service designed on the principles of REST is a RESTful web service. REST (Representational State Transfer) is an architectural style where consumers access a shared resource repository. REST uses URI (Uniform Resource Identifier) to represent business logic. REST is a concept and it permits different data format such as XML, HTML, JSON, while SOAP only supports XML. For example, a client who sends a standard HTTP request may receive a response of XML document (Sheehy et al, 2010). In this project, the RESTful web service is chosen to use. Because of the lightweight infrastructure, there is a very low barrier for developers to adopt it, and the complexity of developing a RESTful web service can be reduced by many development tools.

2.6 Spring Boot

Spring Boot is an integrated framework provided by Pivotal Software, Inc., and it was designed to simplify the development process of a Spring application (Gutierrez, 2017). Spring is an open source framework designed by Rod Johnson to deal with the complexity of J2EE design and development. Spring provides an light-weight application framework for JavaBean, and Spring Boot makes it more convenient for developers to build a web application. In this project, Spring Boot was used in the development of the server. There are many advantages to use this framework, for example, the Tomcat is embedded in Spring Boot and there is no need to deploy the WAR file, the Maven configuration is simplified and the Spring is automatically configured. The advantages make Spring Boot a useful and novice-friendly framework to adopt in developing a server or any web application.

Chapter 3: Design and Implementation

3.1 System architecture

There are mainly three parts in the system: server, app and database as shown in Figure 1.

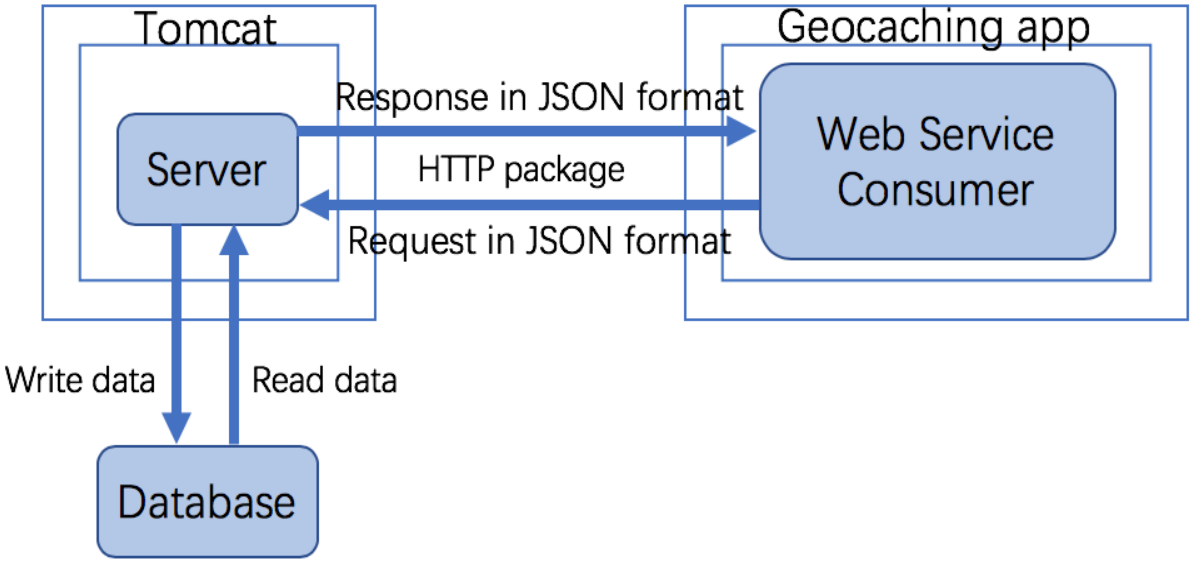

Figure 1 System architecture

The server is developed on IntelliJ IDEA and deployed on Tomcat. The database is developed on MySQL Workbench and it stores all the data in the system. The app is developed on Android Studio and it contains all the GUIs(Graphical User Interface) and the web service consumer, which is also known as the client. The web service consumer makes requests in JSON format via HTTP to the server. The server receives the requests, operates on the data in the database according to the requests, and gives responses back to the web service consumer in JSON format through HTTP. In this architecture, the server and the database can be deployed on a PC, and the app can be deployed on different mobile devices, which realized a distributed system.

3.2 GUI and design of the app

3.2.1 GUI and features

The main page has a navigation bar with 4 items and each item indicates a functional page. There are 4 main functional pages: “History”, “Share”, “Geocaching”, and “More”. Figure 2 shows one of the main features: allowing users to view user information and history records.

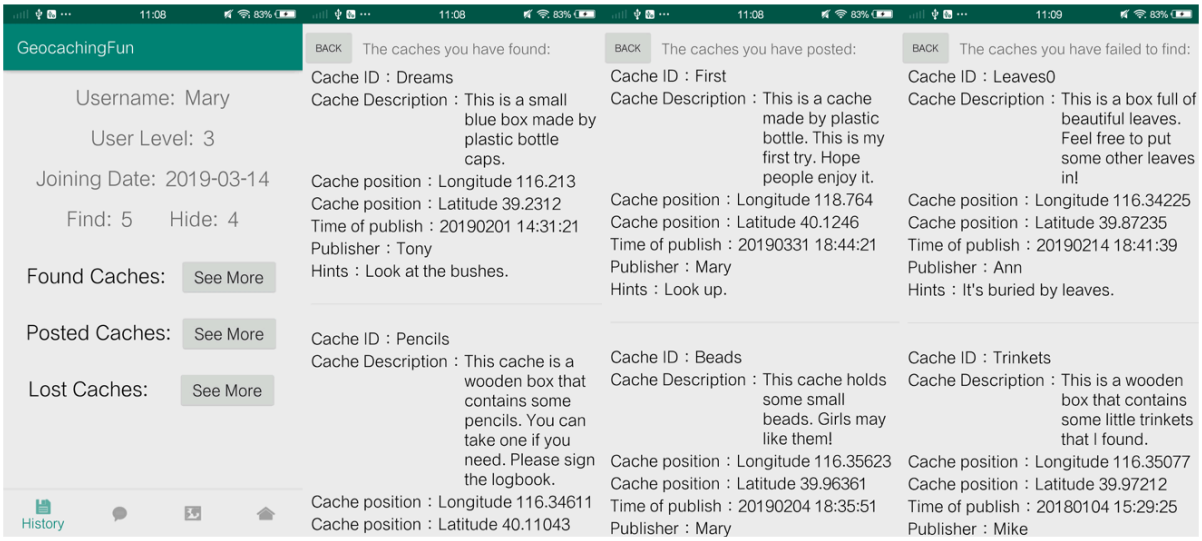

Figure 2 Functional page "History"

As shown in Figure 2, the page "History" shows some basic information of the user, such as the username, the user level, the joining date, and the number of the hidden and tried-to-find geocaches. Users can press the buttons "See More" to see detailed information of the geocaches that the user has found, posted and failed to find.

Figure 3 shows the main features on the page "Share" which allows users to communicate and share with each other.

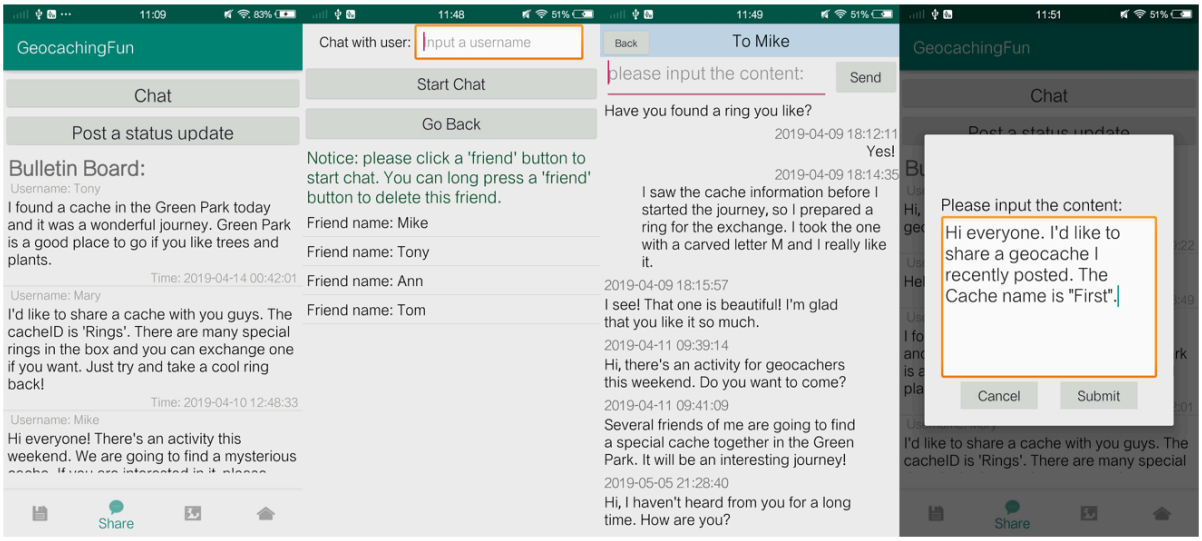

Figure 3 Functional page "Share"

The page "Share" allows users to chat with other users, view status updates on a virtual bulletin board and post status updates on the bulletin board that opens to everyone. As Figure 3 shows, the chatting page contains the history conversations between the two users. Before getting into

the chatting page, several friends' usernames are listed as buttons, users can click a button and start a chat with a specific friend. If users want to start a conversation with a user not included in the friend list, they can simply input a correct username and then start the chat, and the system will automatically add the user as a friend so that users don't need to input the username again for the next chat.

Figure 4 shows the feature of searching and viewing geocaches.

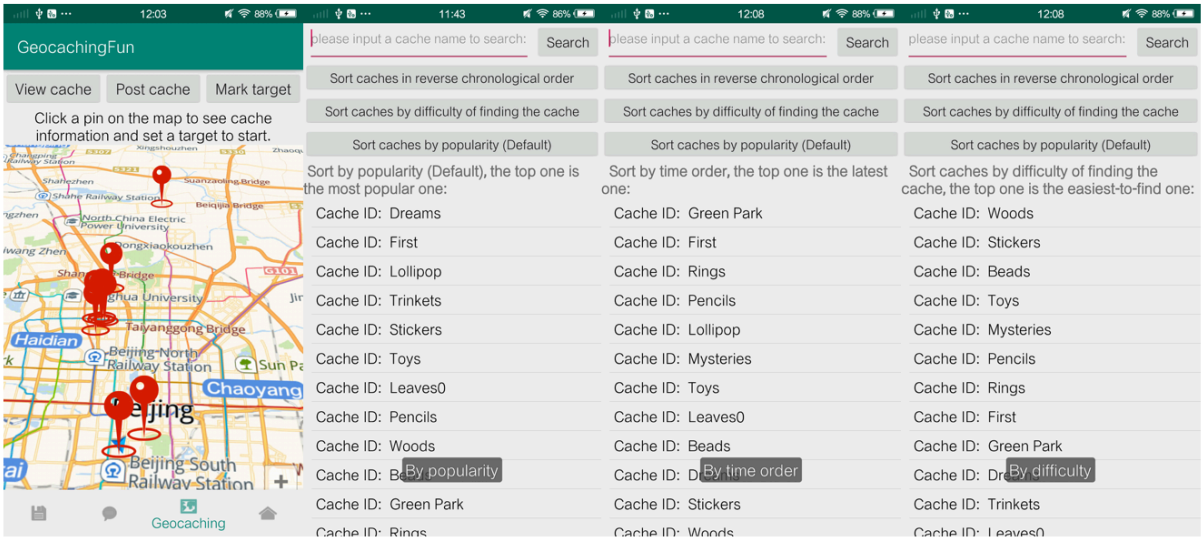

Figure 4 Searching and viewing geocaches

Figure 4 shows three ways for the users to view geocaches. First, they can click on one of the red pins on the map to view the geocache information. Second, they can click the button "View cache" on the "Geocaching" page and then input a cache ID and search for a specific geocache. Third, they can click the button "View cache" on the "Geocaching" page and then click on one cache item in the cache list to view the detailed information. The cache list can be arranged by three different order: time, popularity and the difficulty of finding the geocache.

Figure 5 shows the process of finding a geocache through the help of the app. Once users choose a way to view the geocache, the detailed information is presented. If users are interested in the geocache, they can set it as a target to find. Once a target is set, other pins on the map will disappear and only the positions of the target and the user can be showed on the map. User can utilize the map to help navigate to the target. Once journey is finished, users can mark the geocache as found or lost.

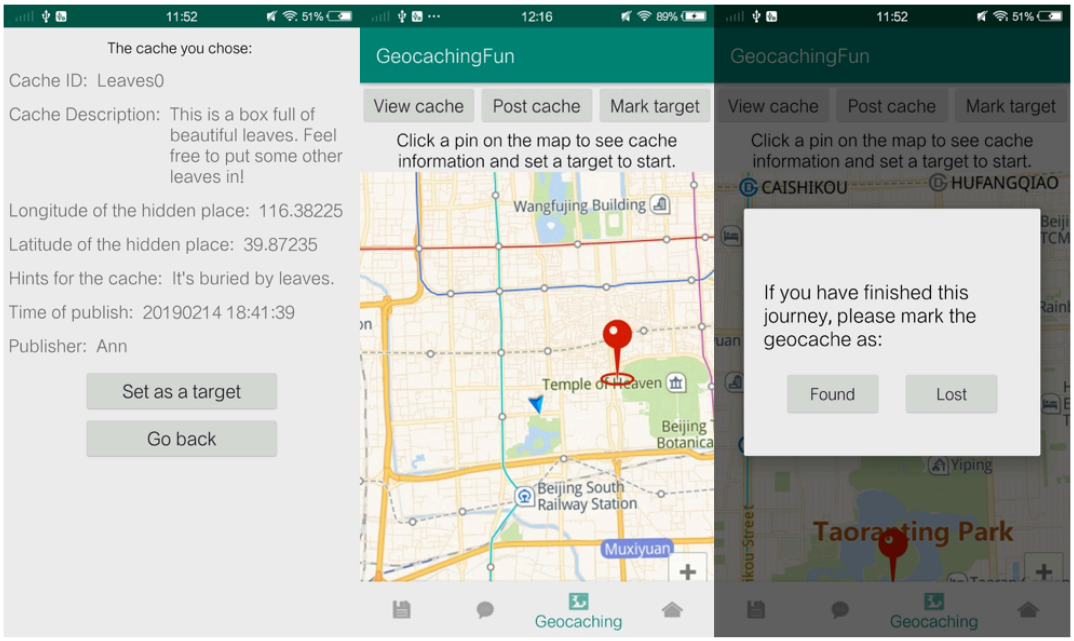

Figure 5 Finding a geocache

Figure 6 shows one of the main feature: posting a geocache.

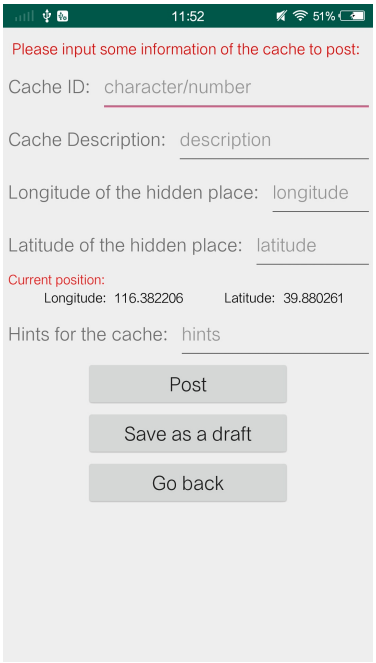

Figure 6 Posting a cache

As shown in Figure 6, the page “Geocaching” allows users to post a geocache by offering basic information of the geocache, such as cache ID, longitude and latitude position of the cache, description of the cache and hints for others users to find the cache. If the users haven’t decided whether to post or not, they can save a draft and then edit it later.

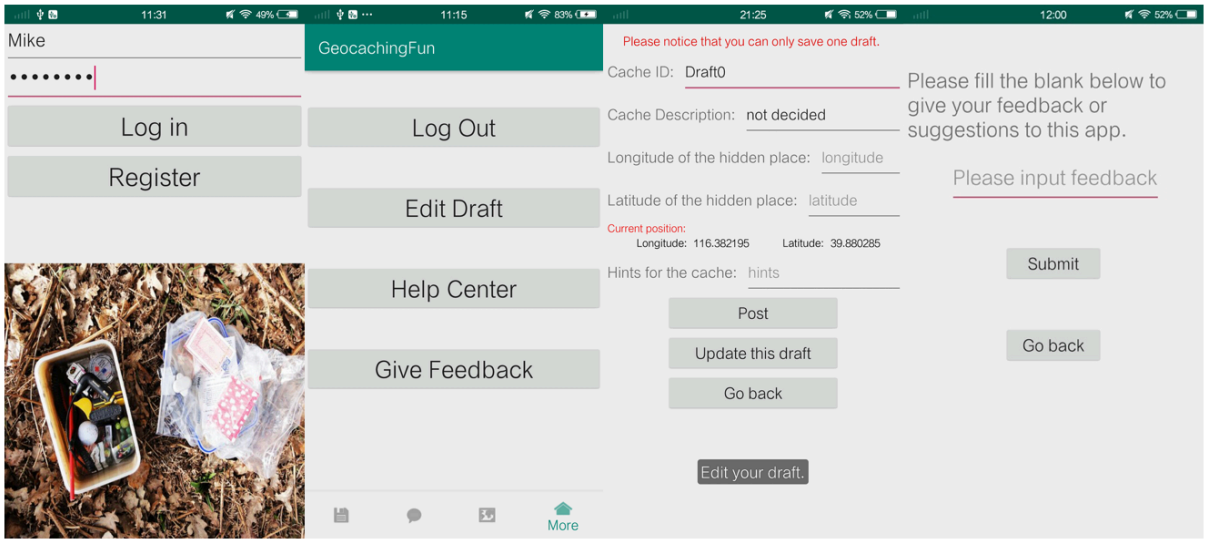

Figure 7 Login page and functional page "More"

As Figure 7 shows, the login page allows users to register or login, followed by the page "History" if the user could successfully login. The functional page "More" allows users to log out, edit the draft of a geocache to be posted, view the help centre, and give feedback to the app. As shown in Figure 8, the help centre contains the introduction of geocaching, suggestions on how to participate in the game, and the accepted rules to follow.

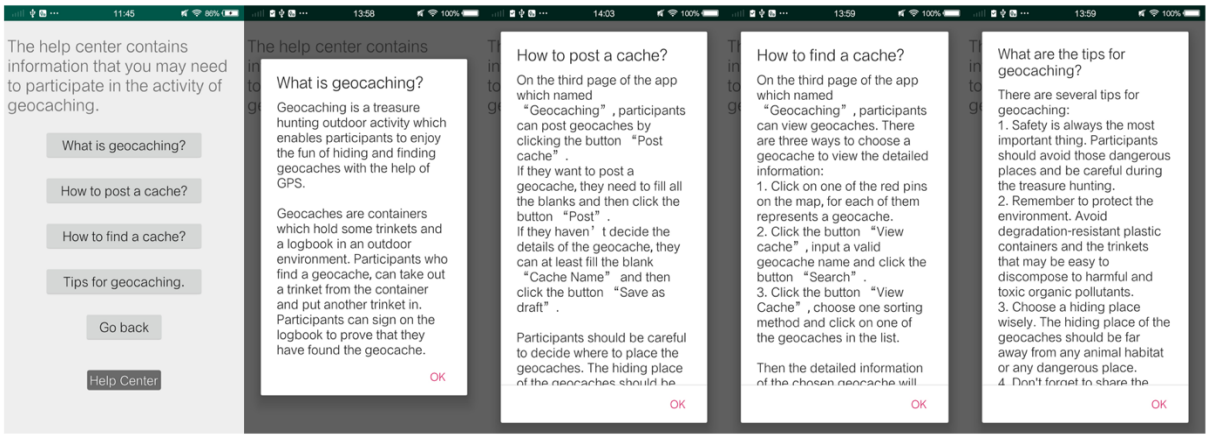

Figure 8 Help centre

3.2.2 Web service consumer

Web service consumers are also known as clients and requesters in a client and server application delivered over HTTP. In this project, the web service consumers are embedded in the app. Once the users click a button or jump to a page, the web service consumer need to make a request to the server to do some operations on the data in the database, such as fetching information from the database and writing data to the database.

In this project, RESTful web services (Selonen et al, 2012) are used, which has no limited standards and the interface has no formal way to describe. Therefore, the service producer and service consumer need to have a mutual understanding of the data being exchanged and process it meaningfully. For example, in this project, the web service consumer need to know which identified URL to use so that the server can listen to the request and then deal with it. Also, the web service consumer needs to know what format the response message from the server has in order to resolve it correctly.

3.3 Server

A web server receives an HTTP request and responds to the request. In this project, the server handles the requests and is deployed on Tomcat, which is a servlet container that executes the program. In this project, Spring Boot, Spring Data JPA, and Hibernate tools were utilized in the development of the server.

3.3.1 Spring Boot

Spring Boot is a module of Spring (Pivotal Software, Inc., 2019) which provides RAD (Rapid Application Development) feature to Spring framework. Spring Boot allows developers to get rid of the templated configuration and simplifies the utilization of the Spring framework to create a stand-alone application. There are other advantages of Spring Boot, for example, Tomcat is embedded in Spring Boot, so WAR files are not necessary to be deployed. Maven configuration is simplified and Spring configuration can be automatically done. Also, it provides the production-ready features, such as metrics, health checks and externalized configuration (Buceamaneatonis et al, 2017).

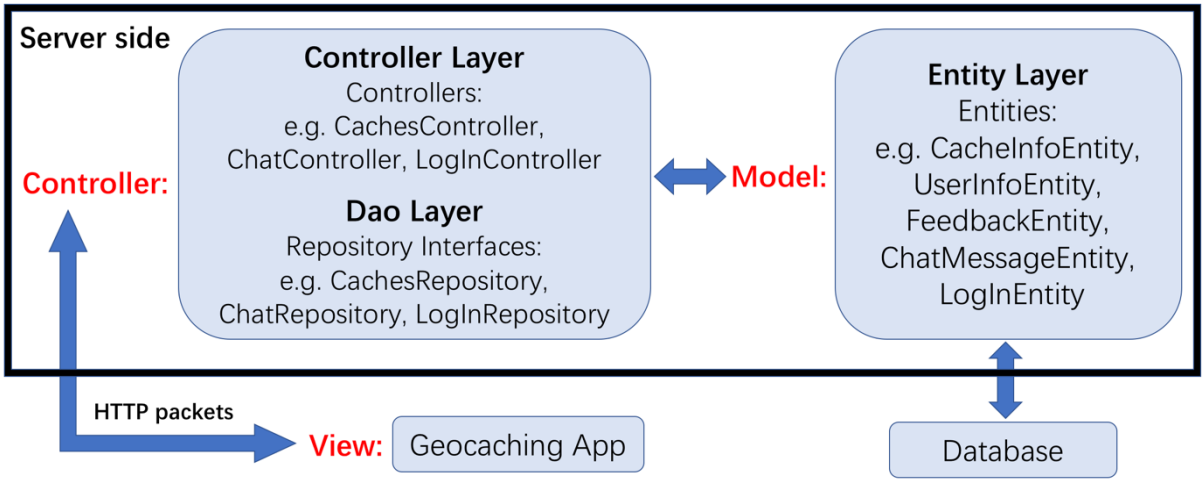

Figure 9 Server architecture

In this project, the architecture of the server can be illustrated by Figure 9. The controller layer deals with the requests from the app. In Spring Boot, annotations are used to realize the address mapping and dependency injection. For example, the @RestController annotation indicates that the result needs to be brought back to the caller. The @RequestMapping annotation ensures that any HTTP request is mapped to the corresponding method by providing routing information. The Dao layer is to realize data persistence. There are many interfaces in the Dao layer worked as a medium for the controllers to manipulate the data represented by the entity layer. By Spring-data-jpa, each interface extends the JpaRepository interface. The business logic can be realized by the framework without instantiation, which makes it more convenient to operate on the data in the database. Entity layer possesses entity classes corresponding to the tables in the database, which turning the data manipulation of POJOs (Plain Old Java Object) into direct data manipulation in the database.

3.3.2 Spring Data JPA and hibernate

This project created a spring-boot application which used JPA (Java Persistence API) to connect to the database. Spring Data JPA is an encapsulated application framework based on ORM(object-relational mapping) and JPA standards, and it makes JPA more easily realized based on Repository. The aim of JPA is to provide a POJO persistence formal standard for ORM (object-relational mapping) (Farshi, 2011). JPA supports two types of metadata, XML and JDK 5.0 annotations. The framework realizes the object persistence in the database according to metadata's description of the mapping relationship between objects and tables. Developers are freed from the tedious JDBC and SQL codes and they can use object-oriented query language to manipulate the entities. In this project, the annotations and the object-oriented query language are used in the controllers and interfaces to manipulate the data.

Hibernate (Anil, 2006) is an open source ORM framework that encapsulated JDBC in lightweight and allows developers to manipulate the database in object-oriented mode. Hibernate has its own query language named HQL to deal with the objects. In this project, Hibernate was utilized to manipulate the database and do operations to the data.

3.4 Database

3.4.1 ER diagram of the database

There are nine tables in the database as shown in Figure 10.

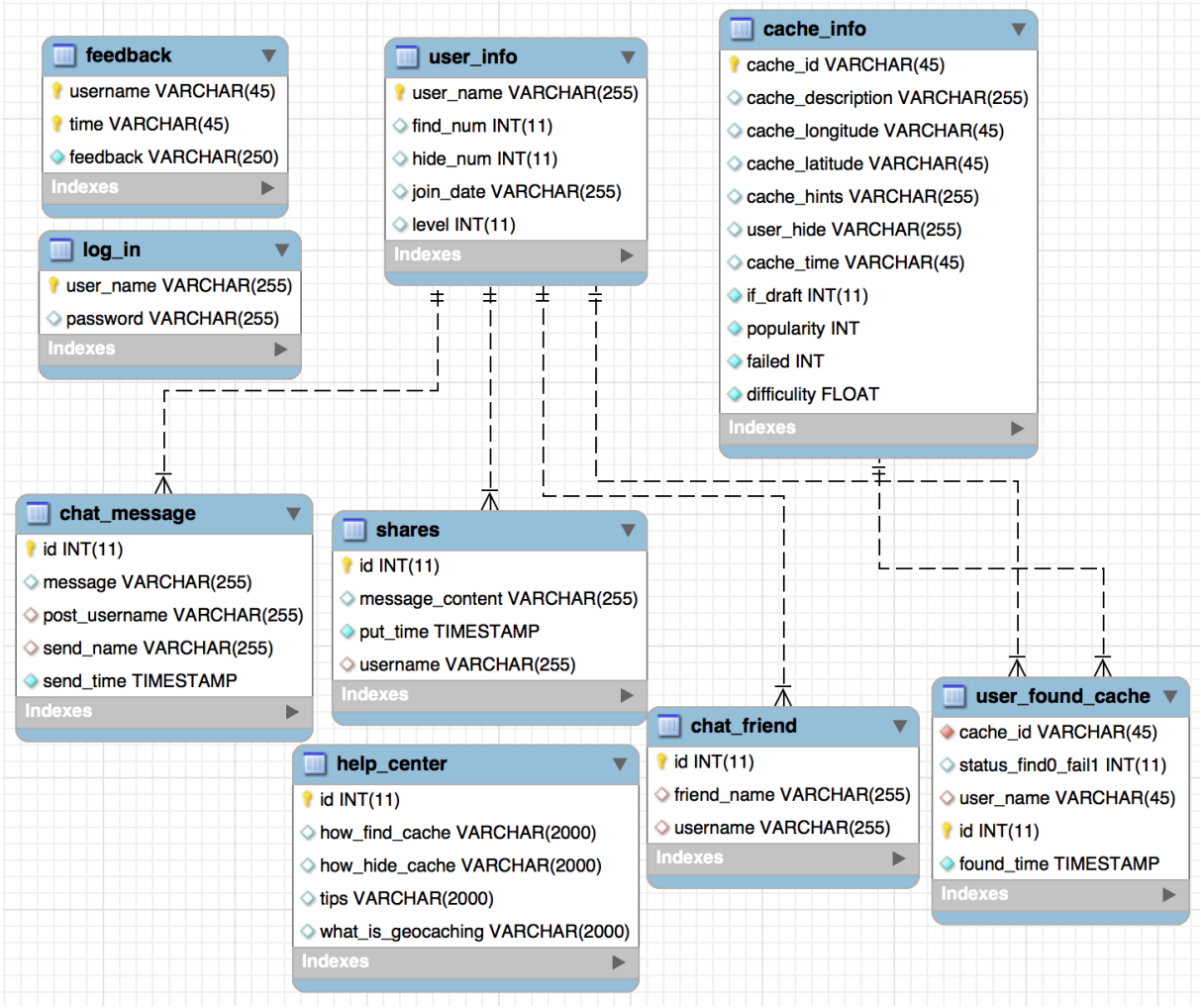

Figure 10 ER diagram of the database

Table “user_info” stores some basic information of the users and table “cache_info” stores some basic information of geocaches, whose primary keys are separately the username and the cache id. Table “log_in” contains usernames and the corresponding passwords, which could be merged into the table “user_info”, however, in this project, they are separated due to the convenience of developing the login function. Table ‘feedback’ stores users’ feedbacks to the app and table “help_center” stores useful guidance for geocaching. “user_found_cache” is a junction table that keeps users’ history records of found caches. Table “shares” stores users’ statues updates, table “chat_message” stores the chatting messages, and the table “chat_friend” stores users’ relationships, whose primary keys are self-generated ids.

3.4.2 Implementation of the database

The database was connected to the server and the server manipulates the database through Spring Data JPA and hibernate. The server has nine entity classes corresponding to the nine

tables in the database. Through hibernate, the annotations @Entity and @Table in an entity class can link the entity class to a table in the database. The annotation @Id indicates the primary key of the table, and the annotation @Column indicates an attribute. There are other annotations that can represent the relationships between the tables, such as @ManyToMany, @OneToMany, and @ManyToOne. The server can do operations to the database through Spring-data-jpa and hibernate tools.

3.5 Amap API

Gaode open platform (Alibaba, 2002) provides multiple maps for developers to choose, such as the 2D maps, the 3D maps, and the satellite maps. Developers can build a map easily with the help of the Amap API and SDK in web, web service, Android app and iOS app development. Amap API also provides online and offline modes, data support and interaction between multiple maps, which satisfies developer's need on building a map.

In this project, Android 3D Amap SDK was downloaded and configured in Android Studio. The application was registered on Gaode open platform and a key for development was applied. The support documents on the Gaode open platform were useful guidance for development. In the geocaching app, an Amap is displayed and the position of the user and the positions of the geocaches are pinned on the map. The arrow on the map shows the direction that the user is heading, which can help the user navigate to the target geocache.

3.6 Software and hardware used for implementation

3.6.1 Software used for implementation

Android Studio is an integrated development tool kit based on IntelliJ IDEA for Android app development released by Google. In this project, Android Studio 3.2 was used to develop the geocaching app. IntelliJ IDEA is an integrated environment for java programming language development released by JetBrains. IntelliJ IDEA 2018.3.3 was used in this project to develop the server. MySQL Workbench provides visual designing, model building and database management for database administrators and program developers. MySQL Workbench 6.3 was the development tool used for ER diagram modelling and database building. Postman is a powerful Chrome plug-in developed by Google to test and debug web pages. Postman 6.7.4 was used in this project to simulate HTTP requests, such as GET and POST. It is convenient in Postman to set parameters and the body of a request, as well as viewing the cookies, headers and the body of the response.

3.6.2 Hardware used for implementation

The Android smart phone of model DAXIAN LS9300 Android 4.1.1 was borrowed from QMUL for testing. Since the operating speed of DAXIAN LS9300 was too slow, another Android smart phone of model Vivo X6 PlusA Android 5.0.1 was used for testing in this project, which had a GPS receiver embedded in the phone.

Chapter 4: Results and Discussion

4.1 Testing

4.1.1 Black-box testing

Black-box testing focuses on the functionalities of the system. Anyone can carry out the black-box testing by running the features of the system one by one, and there's no need for the tester to get familiar with the business logic of the system. The purpose of the black-box testing is to find errors and bugs in the system through running the functions, and then try to fix them to make sure that the system could run well and could be more stable and robust. Table 3 shows some examples of the test cases that have tested the login and register function.

Table 3: Test cases examples of logging in and register

Features to test	Case No.	Input Operations	Output Display	Outcome
Login	1	Empty username or empty password	Showing a warning dialogue to ask the user not to give empty content	Login failed
	2	Non-existed username and a password	Showing a warning dialogue that tells the user the username inputted does not exist	Login failed
	3	Correct username and the wrong password	Showing a warning dialogue that tells the user the password inputted is not correct	Login failed
	4	Correct username and correct password	Showing a dialogue that tells logging in succeeded	Login succeeded
Register	1	Non-existed username and a password	Showing a dialogue that tells register succeeded	Register succeeded
	2	Existed username and a password	Showing a warning dialogue to tell the user to change a username and try again	Register succeeded

Table 4: Test cases examples of posting a geocache and chatting with others

Features to test	Case No.	Input Operations	Output Display	Outcome
Posting a geocache	1	If there are some blanks not filled and the button "Post" is clicked	Showing a warning dialogue to ask the user to fill all the blanks	Posting failed
	2	If the cache ID has been existed in the system and the button "Post" is clicked	Showing a warning dialogue to ask the user to change a cache ID and try again	Posting failed
	3	If all the blanks are filled, the cache ID inputted does not exist in the system, and the button "Post" is clicked	Showing a dialogue that tells the user the geocache has been posted	Posting succeeded
	4	If the "cache ID" is left blank and the button "Save as a draft" is clicked	Showing a dialogue that tells logging in succeeded	Saving failed
	5	If the "cache ID" is filled and the button "Save as a draft" is clicked, and the user has a draft in the system	Showing a warning dialogue to tell the user that each user could only save one draft of a geocache	Saving failed
	6	If the "cache ID" is filled and the button "Save as a draft" is clicked, and the user does not have a draft in the system	Showing a dialogue to tell the user that the draft has been saved and could be edited later	Saving succeeded
Chatting	1	If the inputted username to send messages to does not exist and the button "Start Chat" is clicked	Showing a warning dialogue to tell the user to input a correct username and try again	Chatting failed
	2	If the inputted username is correct and the button "Start Chat" is clicked	Showing the chatting page and the history messages between the two users	Chatting succeeded

Table 4 shows some examples of the test cases that tested the function of posting a geocache and chatting with other users. Notice that each user can only save one draft in the system, so the second trying of saving a draft would be denied. Each draft should at least have the "cache ID" blank filled and the other blanks could be empty in a draft. To post a geocache, all the blanks of the geocache information should be filled, otherwise the operation would be failed. To chat with another user, the inputted username to send messages to should exist in the system.

The black-box testing helps find the errors and flaws of the system by testing the functionalities with all the possibilities. The tester who came up with all the possibilities of the operations of the system could have no programming skills or no background knowledge of the internal logic of the system. Through black-box testing, the errors could be fixed and the functionalities of the system could all run well.

4.1.2 White-box testing

White-box testing is usually implemented by developers who know the business logic of the system and it tests on the internal logic of the system. While black-box testing focuses on the input and output of the functionalities, white-box testing focuses more on how the functionality is realized and what business logic is behind this. Flow charts can help find the flaws of the logic. Figure 11 shows the interfaces of the function "edit a draft". Figure 12 shows a flow chart of the function "edit a draft".

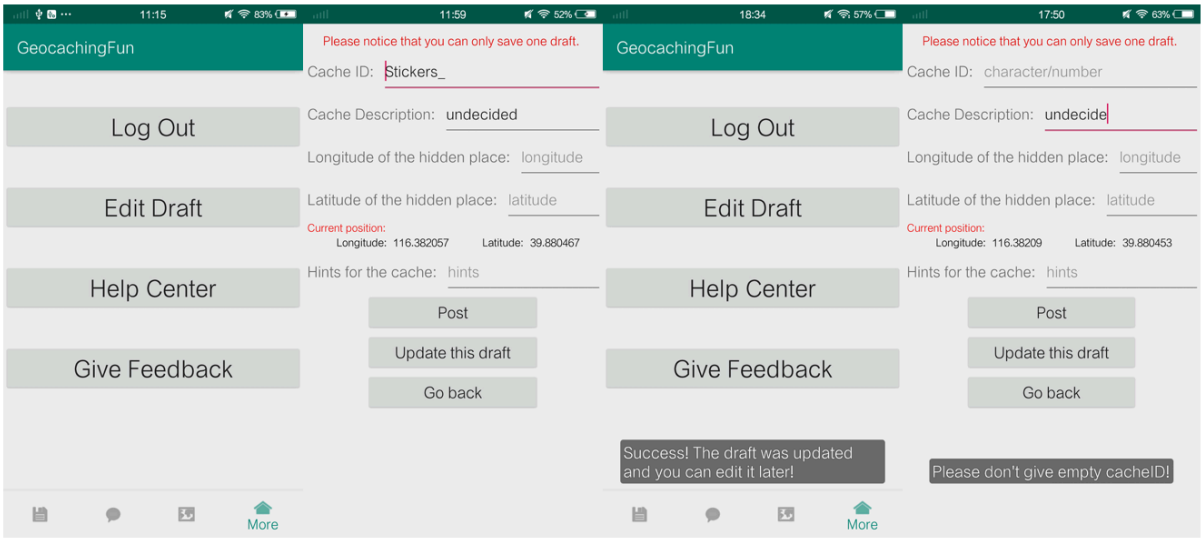

Figure 11 Interfaces of the function "edit a draft"

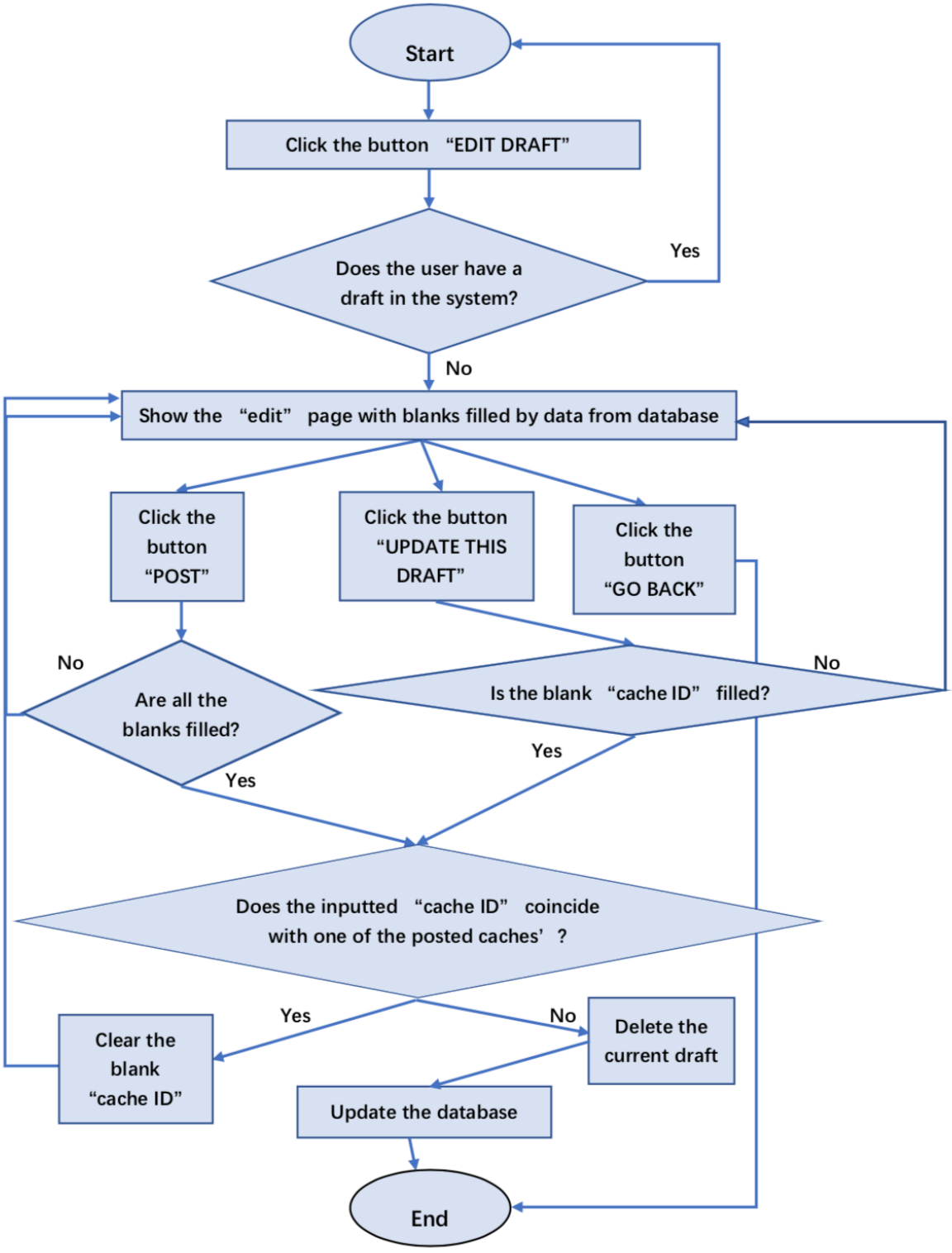

Figure 12 Flow chart of the function "edit a draft"

The flow chart helps display the business logic the function "edit a draft" and it shows how the developer has realized this. If any segment is wrong, the developer would know where to look up and how it supposed to be fixed. In this function, there was a specific problem. If the

developer forgot to delete the current draft of the user when the user successfully updated the draft or posted the draft, the invalid draft would be stored in the database forever and the user could never save a new draft, for each user could only save one draft. The flow chart helped discover the error.

4.2 Analysis of developed system

4.2.1 Pros and cons of the system

The system realized the common user requirements of a geocaching app, such as allowing users to view cache information, set a target geocache and mark the target geocache as found or lost. The map in the app has a pointer showing where the user is heading, which could help the user figure out the direction to go. Additionally, the system realized some features that other apps' free versions didn't realize, such as showing users' history records of the posted, found and lost geocaches, allowing users to chat with other users and post status updates, providing instructions on how to participate in this activity and offering platforms for users to give feedback to the system.

In addition to the functionalities, the app has clear texts and buttons to lead the users to participate in this activity. The response to each operation allows users to feel involved and interacted. For example, when the user inputs an existed username and the correct password, the system will show a small popup dialog to tell the user that he/she logged in successfully. While, if the user tries to login with a wrong username or password, the system will show the mistake and tell the user how to fix it, which is quite user-friendly, as shown in Figure 13.

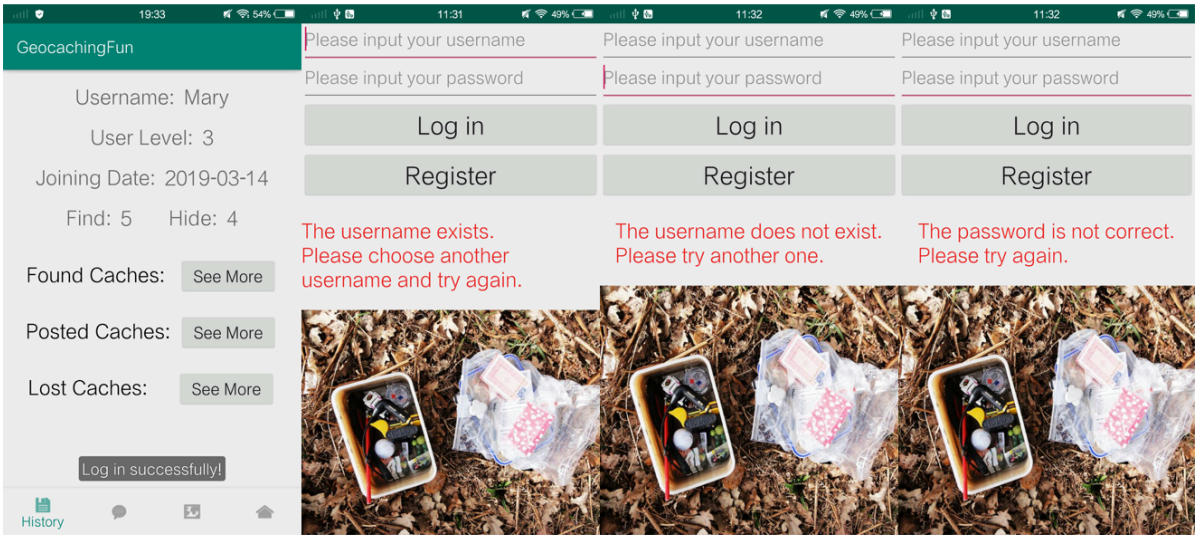

Figure 13 Interaction with users

However, the app still needs some improvements, since there are some features that other apps possess but this app does not. Some apps can show the attributes of the environment near the geocache, such as whether pets are allowed in the park where the target geocache was located, whether there's parking lot or not, and whether the road is friendly to take a baby carriage. These attributes can be sweet instructions which give full consideration to people's needs. Additionally, other apps provide offline features that allow users to participate in the activity without an Internet, for example, showing an offline map to assist users to find the geocache, and saving the user operations to the local database temporarily and then updating the remote database on the server once the Internet is available.

4.2.2 Special features of the app

The app could work like a social media platform by offering two functionalities to the users: chatting with other users and posting status updates. Users could chat with one user at a time and the chatting page would show the history messages in chronological order between the two person. The system would always jump to the latest messages for users to catch up easily if there are too many history messages. After the first conversation between the two users, the system would automatically add the users to each other's friend list so that they could click buttons to talk to their friends the next time instead of inputting the username again and again, which is convenient for users to start a chat. Additionally, Users are also allowed to delete a friend.

The app also allows users to post status updates, which could give the users a sense of being connected to the others. If the user would like to share their experience or feelings with others, they could post a status update onto the virtual bulletin board which is open to every user. The bulletin board shows users' status updates in time order with the latest status update on the top. The app never forces users to share, and it only offers a platform for the users who would like to share and enjoy themselves sharing with others. While, there are still some improvements to make, for example, allowing multiple users to chat at the same time, allowing users to choose to only share their status updates to the users on the friend list, and allowing users to choose to close the bulletin board and the chatting channel if they don't enjoy social activities.

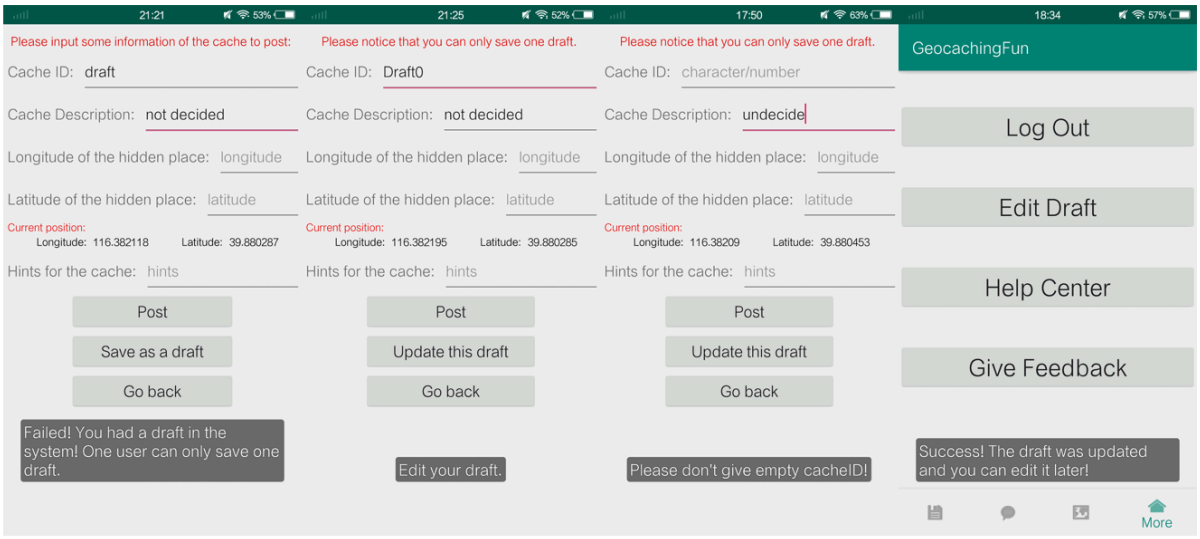

Figure 14 Saving a draft of a geocache to be posted

The app also have some other special features like allowing users to edit a draft of the geocache to be posted, in case they haven't decide where to post it or what hints to give, as shown in Figure 14. While users can only save one draft, if they already had a draft in the system, they could not save a second one. Additionally, the app provides a help center for novices to get introduction and instructions, which helps users to start the activity more easily. Users are also allowed to give feedback to the app and the system to help the developer to make improvements.

Chapter 5: Conclusion and Further Work

5.1 Conclusion

5.1.1 Purpose of the project

The purpose of this project was to development a smart phone app that could help people participate in the activity of geocaching more conveniently. There are some requirements of the geocaching app. First, the app should provide a platform for users to post geocaches. Every user should be able to view geocache information and set one target to find through the platform. Second, the app should present a map to users to assist them navigate to the target, and the map should show the user's positioning and the location of the target geocache. Third, the app should keep a history record of the posted, found and failed-to-find geocaches, and allow users to view their history records. Fourth, the app should work as a social media platform that allows users to share experience and communicate with others. Last but not least, the app should teach users how to participate in this activity and allow users to give feedback to the app. To realize the functionalities of this app, a database and a server were supposed to be designed and implemented.

5.1.2 Achieved outcomes

There were three outcomes in this project: a smart phone geocaching app, a database and a server. The Android geocaching app was developed in Android Studio 3.2. The opening page required users to log in or register. The main page included a bottom navigation bar with four items: "History", "Share", "Geocaching", and "More", and each item represented a functional page.

"History" page allowed users to view user information and history records of the posted, found and lost geocaches.

"Share" page allowed users to chat with other users by offering the username. Once a conversion is made between the two users, the system would automatically add each other to the friend list and keep a record of the exchanged messages. "Share" page also provided a bulletin board for users to post status updates to share their experience and feelings and view other's status updates.

"Geocaching" page presented a map to users with pins, where each pin represented a geocache. Users could click a pin and view the detailed information of the geocache. This page also

allowed users to search for a specific geocache and view the geocaches in three orders. In this project, the geocaches could be arranged by time, by popularity, or by the difficulty of finding the geocaches. If users were interested in a geocache, they could set the geocache as a target to find, and then all the other pins would automatically disappear from the map. Users could read the map to navigate to the target. Once they finished the journey, they could mark the target geocache as found or lost, which would be automatically recorded by the app and users were allowed to view this record later on "History" page. Also, users were allowed to post a geocache in this page by offering some information. If they submitted the geocache information, the new geocache would be pinned on the map. If they saved the geocache information as a draft, they could edit it later on "More" page.

"More" page allowed users to log out, edit the draft of a geocache to be posted, get into the help center, and give feedback to the app. The help center could give users an introduction of geocaching, teach them how to participate in this activity, and offer some tips for geocaching to users.

The database was developed in MySQL Workbench 6.3. There were nine tables in this database. Table "user_info" managed user information, which included the username, the joining date, the user level, and the number of caches that the user hid or tried to find. Table "cache_info" managed cache information, which included cache id, description, position, hints, the name of the user who hid it, the time of release, draft status, the popularity, the difficulty and the times that users failed to find. Table "log_in" stored the username and the password, which helped check users' identification. Table "feedback" stored users' suggestions to this app and table "help_center" included some introduction to the activity of geocaching and some instructions on how to participate in geocaching. Table "user_found_cache" kept the history records of the caches that users had tried to find. Table "chat_friend" stored the usernames of the users' friends that they had chatted with before, and table "chat_message" stored the messages that two users had exchanged. Table "shares" included the status updates that users had posted to the bulletin board.

The server was developed in Intellij IDEA 2018.3.3 and the model Spring Boot was applied in the development. The RESTful web services were chosen to use. There were three layers in the server: the controller layer, the dao layer, and the entity layer. The controller layer included some controller classes and each of them was intended to deal with some specific HTTP requests, which were defined in the controller class. The dao layer worked as a medium, which had several repository interfaces. Each interface was related to one entity class, and each entity

class represented a table in the database. Therefore, the controllers could manipulate the data in the database through the dao layer by the entity classes. Once the controller finished the operations to be done in the database, the controller would send an HTTP response back to the app so that the app could show the response to the users.

The whole system was connected by web services. The app worked as the web service consumer, which was known as the client of the system, and it could send HTTP requests to the server. The server was designed to deal with the requests and to give responses. The server was responsible for manipulating the database according to the requests, such as fetching data from the database, inserting data to the database, and comparing data in the database.

5.1.3 Problems and changes in the project

There was a problem in the development of the app. The Baidu Map API (Baidu, 2010) did not provide an English version of the maps, however, the other features of the developed app were presented in English. In order to make the app in unified language, the Amap API was applied to use as a solution. The Amap API can provide maps in English language and it does the functionalities that the Baidu Map API does, such as showing the 3D maps, positioning and presenting the offline maps. Therefore, this was not a compromised solution, but rather a better solution.

One change in the project was the type of web services utilized in developing the server. As mentioned in the “Project Specification Form”, SOAP-based web services would be used. However, the RESTful web services were finally applied in the development of the server. There are several reasons to make the change. First, REST is more suitable for this project since the server does not need to maintain a state of information from one request to another. In this project, different users can make requests to the server at the same time and the requests may not related to each other. Second, SOAP has rigid standards since a WSDL file is needed and the messages are required in XML format, while, REST does not have strict standards and many format can be used. In this project, JSON format was used, which is a clear and simple format. Third, RESTful web services consume less bandwidth than SOAP-based web services because the latter has heavy-weight contents in the SOAP messages (Muehlen et al, 2005).

5.2 Future work

To provide more convenience for users, future work includes offering offline features in the app. For example, the app could allow users to fetch user information, cache information, and

“help centre” information from the database on the server once the user logged in when the Internet is available. An offline map should be presented to help users navigate to the target geocache without the Internet. Any operation from the user happened offline should be recorded by the local database of the app in the smart phone. Once the app had an Internet connection, the local records would be updated to the main database on the server, which could cause no contradiction because each user would have a distinct local database and the local operations would not affect other users. The system allows more than one user to find the same geocache at the same time, and the app only records which geocache the user has found, i.e. the container of the trinkets, instead of which trinket the user has exchanged, therefore no contradiction needs to be solved in this case.

Also, the app could generate and present the path lines of the participants on the map during a treasure-hunting journey, which could be captured by users in screenshots to share in other social media platforms. Since GPS could not be so accurate in showing exactly where the geocache is located, this feature would not take away other user’s joy of finding the same geocache if one of the users posted the path lines to the geocache.

References

Alibaba. (2002) *Amap API.* [online] Available from: https://lbs.amap.com [Accessed 17 March 2019].

Alibaba. (2002) *Amap.* [online] Available from: https://lbs.amap.com/getting-started/map [Accessed 14 April 2019].

Alibaba. (2002) *Gaode Open Platform.* [online] Available from: https://lbs.amap.com/api/android-sdk/guide/create-map/show-map [Accessed 6 April 2019].

Anil H. (2006). Agile java development with spring, hibernate and eclipse. *Sams.*

Baidu. (2010) *Android Maps SDK, Baidu Maps Open Platform.* [online] Available from: http://lbsyun.baidu.com/index.php?title=androidsdk [Accessed 19 April 2019].

Belqasmi, F., Singh, J., Melhem, S. Y. B., & Glitho, R. H. (2012). Soap-based vs. restful web services: a case study for multimedia conferencing. *IEEE Internet Computing*, 16(4), 54-63.

Buceamaneatonis, R., & Buceamaneatonis, R. (2017). How to design a web survey using spring boot with mysql: a romanian network case study. *Social Science Electronic Publishing*, 17(2), 63.

Farshi, S. J. (2011). Portlet development with spring portlet MVC and WebSphere portal V7. *Conference of the Center for Advanced Studies on Collaborative Research.*

Groundspeak, Inc. (2000). *Geocaching.* [online] Available from: https://www.geocaching.com/play [Accessed 10 January 2019].

Gutierrez, F. (2017). *Spring Boot. Spring Boot Messaging.*

JavaTpoint. (2011) *Spring Boot JPA.* [online] Available from: https://www.javatpoint.com/spring-boot-jpa [Accessed 2 February 2019].

Muehlen, M. Z., Nickerson, J. V., & Swenson, K. D. (2005). Developing web services choreography standards - the case of rest vs. soap. *Social Science Electronic Publishing*, 40(1), 9-29.

NOAA. (2017) *GPS Overview.* [online] Available from: https://www.gps.gov/systems/gps/ [Accessed 5 March 2019].

O'Hara, K. (2008). Understanding geocaching practices and motivations. *Sigchi Conference on Human Factors in Computing Systems.*

Pivotal Software, Inc. (2019) *Spring Boot.* [online] Available from: https://spring.io/projects/spring-boot [Accessed 31 March 2019].

Pivotal Software, Inc. (2019) *Spring Guides.* [online] Available from: https://spring.io/guides [Accessed 10 April 2019].

Selonen, P., Belimpasakis, P., You, Y., Pylvänäinen, T., & Uusitalo, S. (2012). Mixed reality web service platform. *Multimedia Systems*, 18(3), 215-230.

Sheehy, J., & Vinoski, S. (2010). Developing restful web services with webmachine. *IEEE Internet Computing*, 14(2), 89-92.

Sullivan, L. (2016) *Geocaching is easy with these Android apps.* [online] Available from: https://www.androidapps.com/geocaching-is-easy-with-these-android-apps/ [Accessed 31 March 2019].

The Geocaching Junkie. (2017) *Cachly vs Geocaching App: A comparison Review.* [online] Available from: https://thegeocachingjunkie.com/2017/02/05/cachly-vs-geocaching-app-a-comparison-review/ [Accessed 2 March 2019].

Vartiainen, T., & Tuunanen, T. (2013). Co-creation of Value for IT-Enabled Services: A Case of Geocaching. *Hawaii International Conference on System Sciences.*

Vinoski, S. (2008). Restful web services development checklist. *IEEE Internet Computing,* 12(6), 96-95.

Wikipedia. (2019) *Geocaching.* [online] Available from: https://en.wikipedia.org/wiki/Geocaching#History [Accessed 4 April 2019].

Zed Said Studio LLC. (2015) *Cachly Help.* [online] Available from: http://www.cach.ly/help/index.php/Main_Page#Cachly_Help_v5.0 [Accessed 2 March 2019].

Acknowledgement

My deepest gratitude goes first and foremost to my supervisor of the this project. She offered many useful and clear suggestions during the whole project. She organized the remote meetings and the face-to-face meetings properly so that every meeting made full use of time. During the meetings, I could report my progress, discuss the problems and changes with my supervisor, and get encouraged to work harder. I appreciated her attitude of making everything precise and clear, as well as her ability of critical thinking while giving advices. I learned a lot from my supervisor, such as how to organize a literature review, what precise words to use in the report or slides, and where to find useful information online. My supervisor took the responsibility of her job to instruct me on doing this project and I appreciated her work and time. Again, I'd like to provide my greatest gratitude to my supervisor of this project.

Second, I would like to express my gratitude to BUPT and QMUL, as well as all the teachers who have taught me before. Without the background knowledge and skills, I might not finish this project successfully. I appreciated every teacher's hard work and their rigorous teaching attitudes. Additionally, I'd like to thank QMUL for lending me an Android smart phone as the testing equipment in this project.

Last, my thanks would go to my beloved family for their constant care and support for years. Also, I appreciated my own hard work and the time I spent on this project.

Appendix

北京邮电大学 本科毕业设计(论文)任务书

Project Specification Form

Part 1 – Supervisor

论文题目 Project Title	A Geocaching smart phone app		
题目分类 Scope	Software Development	Implementation	Software
主要内容 Project description	Geocaching is an outdoor activity which allows people to both hide and look for objects called ”geocaches”, using a Global Positioning System (GPS) receiver. The aim of this project is to develop a smart phone app that allows users to participate in a geocaching activity, in a specific geographical area. The smart phone app would download the location of nearby ”geocache” locations from the internet and use the GPS receiver on the phone to to help locate said ”geocaches”. The developed smart phone app should also include extra features such as hints to help find the ”geocaches”, and store a record of the ones found by the user.		
关键词 Keywords	web services, database, smart phone app, GPS, geocaching		
主要任务 Main tasks	**1** Requirements analysis, and learning/familiarisation with GPS, geocaching, and the sites that provide ”geocache” locations. Carry out a literature search of existing similar applications.		
	2 Learn how to program a smart phone app for a chosen SDK, and a database to store users’ history about ”geocacaches”.		
	3 Develop a server that provides the information to the smart phone app, which it can get from existing web services.		
	4 Develop a smart phone app from gathered requirements and test with appropriate data.		
主要成果 Measurable outcomes	**1** A smart phone app that will assist with participating in a geocaching activity.		
	2 A server that will deliver the location of ”geocaches” to the app and help manage users’ geocaching history.		
	3 A web service consumer that pulls ”geocache” information from existing web services.		

北京邮电大学 本科毕业设计（论文）任务书

Project Specification Form

Part 2 - Student

学院 School	International School	**专业 Programme**	**Internet of Things Engineering**		
姓 Family name	Wang	**名 First Name**	Ran		
BUPT 学号 BUPT number	2015213404	**QM 学号 QM number**	151006766	**班级 Class**	2015215119
论文题目 Project Title	A Geocaching smart phone app				
论文概述 Project outline **Write about 500-800 words** **Please refer to Project Student Handbook section 3.2**	Geocaching is a treasure hunting activity which enables participants to enjoy the fun of hiding trinkets and finding unknown trinkets from other participants with the help of Global Positioning System (GPS) embedded in a mobile phone. Participants need to hide or find containers called “geocaches” which hold trinkets and logbooks in an outdoor environment. The trinkets are trade items, which means participates should take out one item from the container and put another item back. Participants can sign on the logbook to prove that they have found the geocache. A Geocaching app should assist users to participate in the Geocaching activity by offering the following basic functions: 1. Allow users to search for a geocache as a target and provide users with a map to help users navigate to the position of the geocache. 2. Allow users to post a geocache with description and location of it, as well as hints to help other users find the geocache more easily. Allow users to save a draft of post if they haven’t decided yet. 3. Allow users to view history records of posts and findings, and mark whether a target geocache is found or not. 4. Allow users to register and log in/out. 5. Allow users to share and communicate with each other to express their thrilled feelings after finding a difficult-to-find geocache or to report a damage or missing of the geocache. To implement that, the app should allow users to send messages to a specific user by offering the username or to share their experience and feelings to everyone on a virtual bulletin board. 6. Provide introductions of geocaching and instructions on how to participant in the activity for novices. The Geocaching smart phone app can be regarded by users as a guide to participate in geocaching, a platform to search and post geocaches, a database to store their history records of posts and findings, a tool to help them locate the target geocaches, as well as a social media platform to share experience and information. The Geocaching app should download the locations of the geocaches from the Internet and the GPS receiver on the mobile phone should help locate the geocaches. When users try to get the locations of the geocaches stored in the database, the web service consumer will send request whose format is XML to the server by HTTP protocol, which also follows the SOAP (Simple Object Access Protocol). After the server receives the request, it will invoke the web				

	service that matches the parameters and return values of the request and the WSDL (Web Service Description Language) address to extract the locations of geocaches. WSDL is used to describe the availability of the service. The database can also be accessed through the web service. The web service will unpack the SOAP message and process the converted information as required and generate a response. The response will be sent back to the web service consumer in XML format through HTTP and then presented to the user on the interface of the app. In this project, Android Software Development Kit (SDK) is chosen to develop the Geocaching app, and Android Studio Version 3.1 which is an Integrated Development Environment (IDE) for developing on the Android platform will be used. MySQL Workbench Version 6.3 which is a visual database design tool will be used for developing the database. Background material: https://www.geocaching.com/play https://developers.google.com/maps/documentation/android-sdk/map https://developer.android.com/things/sdk/drivers/location https://www.mapbox.com/help/android-navigation-sdk/ https://developer.android.com/guide/topics/location/strategies#java https://www.tutorialspoint.com/webservices/web_services_examples.htm https://www.c-sharpcorner.com/uploadfile/puranindia/creating-the-web-service-consumer/ Wimmer M., Eberhardt D., Ehrnlechner P., Kemper A. (2004) Reliable and Adaptable Security Engineering for Database-Web Services. In: Koch N., Fraternali P., Wirsing M. (eds) Web Engineering. ICWE 2004. Lecture Notes in Computer Science, vol 3140. Springer, Berlin, Heidelberg Garney, W. R., Young, A., Mcleroy, K. R., Wendel, M. L., & Schudiske, E. (2015). A qualitative examination of exergame motivations in geocaching. Games for Health Journal, 5(1).
道德规范 **Ethics**	Please confirm that you have discussed ethical issues with your Supervisor using the ethics checklist on QMPlus. YES

	Summary of ethical issues: N/A
中期目标 **Mid-term target.** **It must be tangible outcomes, E.g. software, hardware or simulation.** **It will be assessed at the mid-term oral.**	Finish gathering and analysing user and system requirements, as well as making an Entity Relationship (ER) diagram of the system. Finish building a basic structure of the database and inserting test data to the database. Finish building a basic structure of the app and forming the basic interfaces.

Work Plan (Gantt Chart)

Fill in the sub-tasks and insert a letter X in the cells to show the extent of each task

	Nov	Dec	Jan	Feb	Mar	Apr	May
Task 1 break down							
Gather and analyse user and system requirements.	X						
Familiarisation with GPS, geocaching, and sites that provide geocaches.	X						
Carry out a literature research of existing similar applications.	X	X					
Task 2 break down							
Learn how to program a smart phone app using Android Studio.		X	X				
Make an Entity Relationship diagram of the system.		X	X				
Build a basic structure of the app and form the user interfaces.			X	X			
Build a basic structure of the database and insert testing data.			X	X			
Task 3 break down							
Develop a server that provides the information to the smart phone app, e.g. presenting the records of the geocaches found by the user.			X	X	X		
Develop a Web service consumer that extracts information from existing Web services, e.g. extracting the location of the geocaches.			X	X	X		
Task 4 break down							
Fulfil the remaining features of the app and complete the system, e.g. showing the help center, saving a draft of the user' post, giving hints, etc.				X	X		
Test the individual features of the system and debug, e.g. testing the connection of the app and the server, testing the view of history records, etc.					X	X	
Test the system by some test techniques, e.g. Black-box and White-box.						X	

北京邮电大学 本科毕业设计（论文）初期进度报告

Project Early-term Progress Report

学院 School	International School	**专业 Programme**	**Internet of Things Engineering**		
姓 Family name	Wang	**名 First Name**	Ran		
BUPT 学号 BUPT number	2015213404	**QM 学号 QM number**	151006766	**班级 Class**	2015215119
论文题目 Project Title	A Geocaching smart phone app				

已完成工作 Finished work:

- What material was read or researched?
  The online websites I have carefully read are listed as follows:
  https://www.geocaching.com/play
  https://www.androidapps.com/geocaching-is-easy-with-these-android-apps/
  https://www.lifewire.com/best-geocaching-apps-4171594
  https://thegeocachingjunkie.com/2016/04/21/geocaching-app-review-cachly/
  https://developers.google.com/maps/documentation/android-sdk/map
  https://www.tutorialspoint.com/webservices/web_services_examples.htm

- What work was done?
  I have gathered user and system requirements of the geocaching app. Through visiting the official website of Geocaching [1], I got more familiar with this activity, e.g. the procedures of hiding a geocache, the tips of finding a geocache and how GPS is used in geocaching.

  I have carried out a literature research of existing similar applications of geocaching, in which I listed the features of different geocaching apps in a table and analyzed the fundamental requirements as well as the additional features and advantages of several geocaching apps. After that, I figured out the important features and the creative features of my app. I have also drawn basic prototypes of the interfaces of my app and discussed with my supervisor in the first face-to-face meeting.

  I have downloaded and installed Android Studio Version 3.1 and MySQL Workbench Version 6.3. I tried to start some simple programs in Android Studio by following the instruction documents I downloaded on the official website [2]. Besides, I have generated an ER diagram of the database as shown in Fig.1. The table ‘Log_in_out’ stores all users’ name and password which is used during logging in. The table ‘UserInfo’ stores user’s information and the table ‘CacheInfo’ stores cache’s information. The table ‘User_hide_Cache’ and the table ‘User_find_Cache’ take the responsibility for connecting ‘UserInfo’ and ‘CacheInfo’. For example, when user A wants history records for found caches, the system will search the table ‘User_find_Cache’ first with the username of user A, then several found cache IDs are achieved, and then the system will search the table ‘CacheInfo’ to find out detailed information of the caches and present to user A. The table ‘Share’ stores the message a user has published, and the table ‘HelpCenter’ stores the instructions for geocaching which has no relationship with other tables.

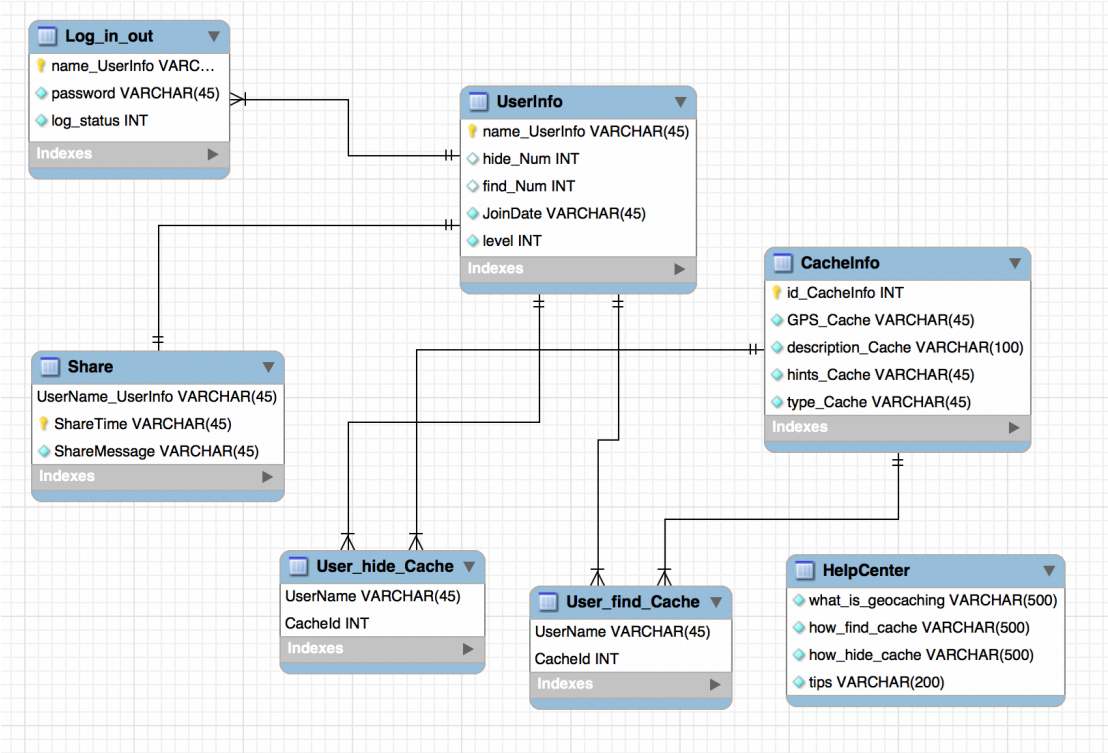

Fig.1 ER diagram of the database

- What problems were faced?
  It was hard to ensure that users could publish whatever messages they want without raising ethic issues.

- What solutions were found?
  The solution to the above problem is to make prescribed rules of the messages that a user can publish. Instead of letting users to type the message, the system will give several options of the contents, e.g. encouraging others to go for your resent hidden geocache, reporting a damage of a geocache, and sharing the thrilled feelings of finding your first geocache.

Reference:
[1] Groundspeak, Inc., (2000). *Geocaching*. [online] Available from: https://www.geocaching.com/play [Accessed 10/1/2019].
[2] Google, (2013). *Android Developers.* [online] Available from: https://developer.android.com/docs/ [Accessed 10/1/2019].

是否符合进度？On schedule as per GANTT chart?

YES

下一步 Next steps:

- What are the next immediate steps?

Learn more about programming in Android Studio and build a basic structure of the app, as well as forming the user interfaces. Build a basic structure of the database according to the ER diagram and insert testing data.

北京邮电大学 本科毕业设计（论文）中期进度报告

Project Mid-term Progress Report

学院 School	International School	**专业 Programme**	**Internet of Things Engineering**		
姓 Family name	Wang	**名 First Name**	Ran		
BUPT 学号 BUPT number	2015213404	**QM 学号 QM number**	151006766	**班级 Class**	2015215119
论文题目 Project Title	A Geocaching smart phone app				

是否完成任务书中所定的中期目标？Targets met (as set in the Specification)?

YES

已完成工作 Finished work:

- Targets set at project specification (i.e. 'Mid-term targets'):
  Finish gathering and analysing user and system requirements, as well as making an Entity Relationship (ER) diagram of the system. Finish building a basic structure of the database and inserting test data in the database. Finish building a basic structure of the app and forming the basic interfaces.

- Finished Work:
  I have familiarised myself with the activity of geocaching and existing geocaching apps. During the study, I analyzed user and system requirements and carried out a literature research of existing similar applications. After I confirmed the features of my app, I generated an ER diagram of the system and built a basic structure of the database in MySQL Workbench, however, I still need to learn how to connect the database to the server.

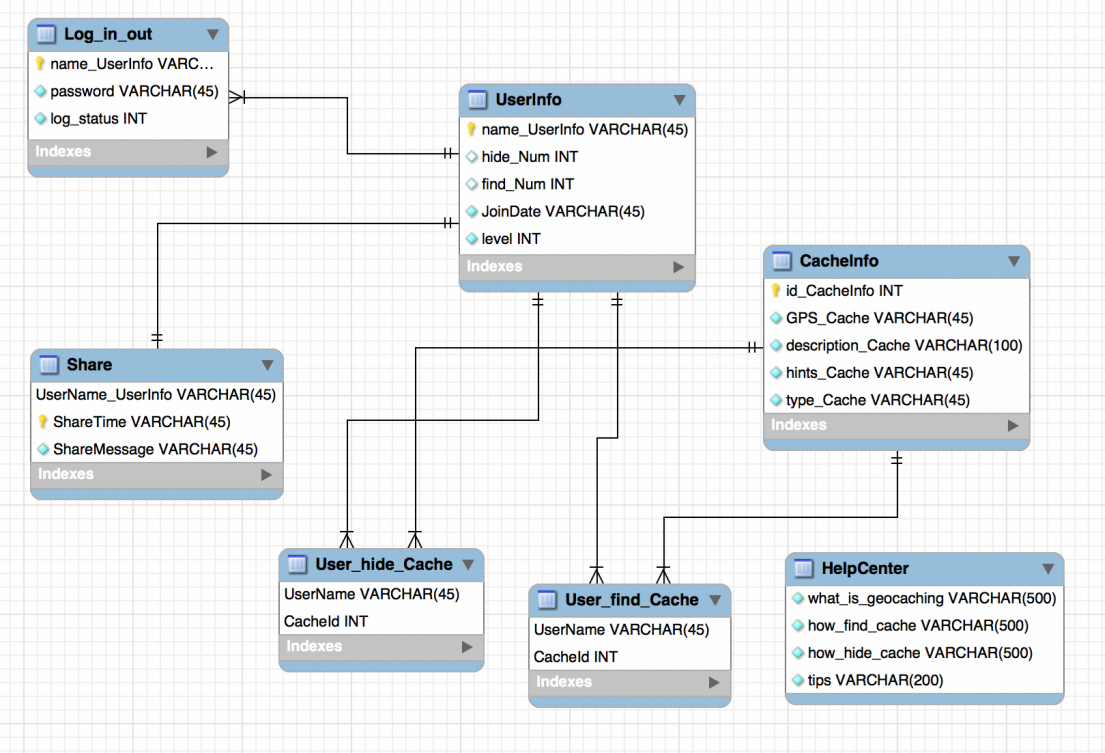

Fig.1 ER diagram of the database

I designed the graphical user interfaces of the app according to the user requirements and the components that could be used in the layout. I have formed a basic structure of the app with four items in the bottom navigation bar, and each indicates a functional page of the app. Currently, only the basic structure of the pages are formed and fake data are presented, which means the web services are not available and the database is not connected. On the page 'History', the user can see the level, joining date of the system, the founded geocaches and the posted geocaches. On the page 'Share', the use can either read and post status updates with predefined format or exchange non-real time messages with a specific user. On the page 'Geocaching', a Baidu Map is shown and users are supposed to post or find the information of a geocache. On the page 'More', users can log out, edit drafts of posting, get instructions and give feedback. However, I have spent days debugging and trying to display the Baidu Map on page 'Geocaching' and there're still bugs in the program. Therefore, no map is shown on the page 'Geocaching' in the following figure.

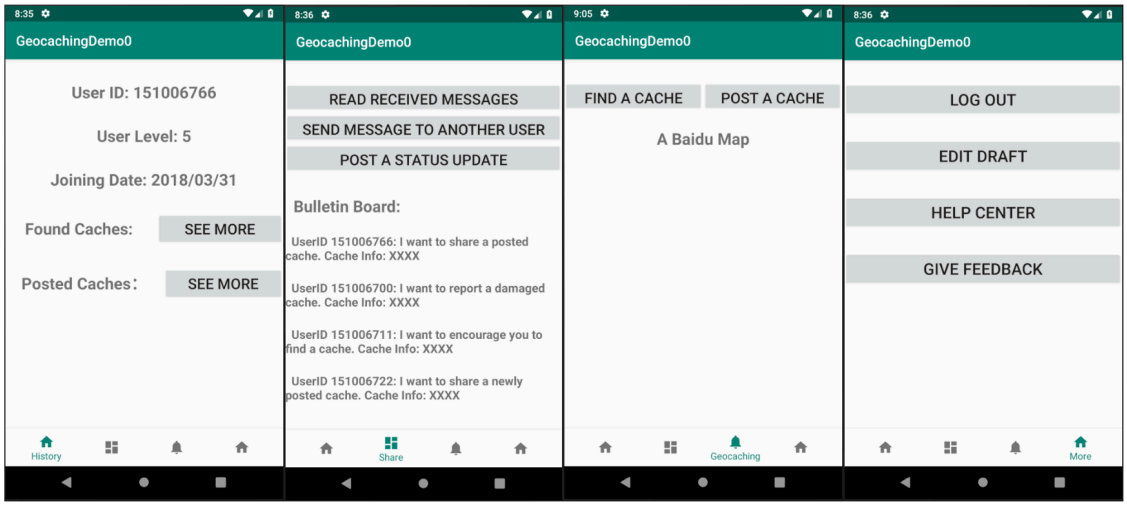

Fig.2 Screenshots of the Geocaching app

I applied for a key to use Baidu Maps in Android development[1], and downloaded Android SDK of Baidu Map and configured it in Android Studio and my application. I have learnt how server and web service consumer work in Android app systems, while I am still learning how to implement them in my app. I downloaded IntelliJ IDEA Version 2018.3.3 for server development and Postman Version 6.7.3 for server testing. I also downloaded and configured Tomcat as a platform where to hold the server. The architecture of the system is shown in Fig.3.

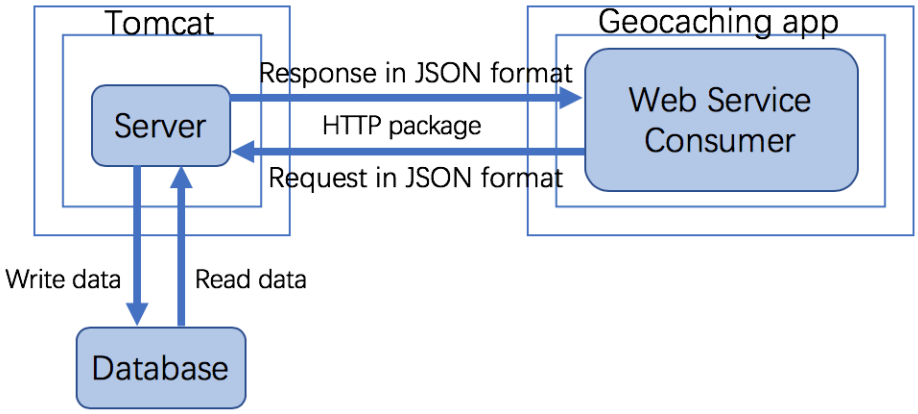

Fig.3 System architecture

Reference:

[1] Baidu, (2012). Baidu Map open platform. [online] Available from: http://lbsyun.baidu.com/index.php?title=androidsdk

尚需完成的任务 Work to do:

- Work to do:
  1. Complete the graphical user interfaces by considering all the user and system requirements.
  2. Develop a server that receives requests from web service consumer and invokes web services to fetch the data from the database or store the data to the database. For example, when users are logging in, the server need to invoke the web service that compare the inputted user ID and password to the existing IDs and passwords in the database to see whether they are matched.
  3. Develop a web service consumer that sends requests to the server and receives responses. For example, when users are logging in, the web service consumer may send a request passing the inputted ID and password to the server, and then receive a true or false that indicates whether the user has logged in successfully.
  4. Connect the database to the server and create services to get access to the data. Insert testing data to ensure the features are satisfied.
  5. Fulfil the remaining features of the app and complete the system by considering all the user and system requirements.
  6. Test the system by test techniques, e.g. functional testing, black-box and white-box testing.

- Can finish on time or not?
  True. I can finish before the project deadline.

存在问题 Problems:

1. At first, I intended to use Google Maps. However Google Maps need Google services to be supported and these cannot be accessed in China without a VPN(Virtual Private Network). While VPN is not easily to get in China today.
2. At first, I considered the feature of logging in as optional for users, which means users could still use the app without logging in. However without user ID, the 'History' page will have limited useful information to present, and it does not make sense when users choose to post status updates and exchange messages with others.

拟采取的办法 Solutions:

1. I decided to use Baidu Maps instead of Google Maps, and I downloaded the Android SDK of Baidu Maps and applied for a key on the website of Baidu Map Open Platform to do Android development.
2. Instead of designing another pair of graphical user interfaces for users who don't log in, I decided to design a welcome page that ask users to log in or register, which means users who don't log in cannot use the app. Once users log in successfully, a page with bottom navigation bar will be presented.

论文结构 Structure of the final report:

Table of Contents:
Abstract
Chapter 1: Introduction
    1.1 Motivation and aims
    1.2 Features of the system
        1.2.1 User requirements
        1.2.2 Achieved Functionalities
    1.3 Structure of the report
Chapter 2: Background
    2.1 Geocaching
        2.1.1 History of geocaching

北京邮电大学 本科毕业设计（论文）教师指导记录表

Project Supervision Log

学院 School	International School	**专业 Programme**	**Internet of Things Engineering**		
姓 Family name	Wang	**名 First Name**	Ran		
BUPT 学号 BUPT number	2015213404	**QM 学号 QM number**	151006766	**班级 Class**	2015215119
论文题目 Project Title	A Geocaching smart phone app				

Please record supervision log using the format below:

Date: dd-mm-yyyy
Supervision type: face-to-face meeting/online meeting/email/other (please specify)
Summary:

Date: 30-10-2018
Supervision type: email
Summary: presented supervision arrangement and gave instructions to start the project.

Date: 5-11-2018
Supervision type: online meeting
Summary: discussed some questions about the project, e.g. what creative features to add to the app and what web service consumer is.

Date: 11-11-2018
Supervision type: email
Summary: made project equipment request.

Date: 13-11-2018
Supervision type: face-to-face meeting
Summary: discussed the project specification and the detailed design of the app, e.g. what ethics are involved in the project, what features the app should have and how to implement to the new feature in the app.

Date: 23-11-2018
Supervision type: email
Summary: gave feedback to the draft specification.

Date: 4-12-2018
Supervision type: face-to-face meeting
Summary: checked progress of the project according to the GANTT chart, discussed the literature research of the project and the structure of the system, and searched web services that could be used in the project online.

Date: 14-12-2018
Supervision type: email
Summary: reminded us to stick to the GANTT chart in the project specification and keep a record, as well as making a request of project supervision log.

Date: 24-12-2018

Supervision type: online meeting
Summary: discussed the feedback of the Project Supervision Log and a problem of the equipment borrowed from QMUL, checked recent progress of the project, e.g. the ER diagram of the database, the literature review, and the submission of the Chinese name of the project on QMplus.

Date: 8-1-2019
Supervision type: email
Summary: gave instructions and suggestions to the early-term progress checking report.

Date: 17-1-2019
Supervision type: email
Summary: informed the arrangement of the following remote meetings and face-to-face meetings.

Date: 12-2-2019
Supervision type: remote meeting
Summary: gave feedback to the Table of Content of the final report and discussed some questions related to it.

Date: 23-2-2019
Supervision type: email
Summary: gave instructions and suggestions on the mid-term progress checking report and slides for mid-term viva.

Date: 2-3-2019
Supervision type: face-to-face meeting
Summary: checked the progress of this project and gave instructions and suggestions on the slides for mid-term viva.

Date: 21-3-2019
Supervision type: remote meeting
Summary: checked the progress of this project and discussed some questions related to the changes in developing the app of this project.

Date: 1-4-2019
Supervision type: remote meeting
Summary: checked the progress of this project and discussed some questions about the project.

Date: 1-4-2019
Supervision type: email
Summary: gave some useful materials about the web services.

Date: 13-4-2019
Supervision type: face-to-face meeting
Summary: checked the progress of this project, reviewed the draft final report and the slides for mock viva, and gave suggestions on how to improve them.

Date: 15-4-2019
Supervision type: mock viva
Summary: students gave presentations one by one and the supervisor asked questions about each project. Finally, the supervisor gave some instructions and suggestions on how to improve the projects and how to make better presentations.

User Guide

Table of Contents

Chapter 1: Introduction

“GeocachingFun” is an app that can assist people to participate in the activity of geocaching. Geocaching is a treasure hunting outdoor activity which enables participants to enjoy the fun of hiding and finding geocaches with the help of GPS. Geocaches are containers which hold some trinkets and a logbook in an outdoor environment. Participants who find a geocache, can take out a trinket from the container and put another trinket in. Participants can sign on the logbook to prove that they have found the geocache.

The app “GeocachingFun” contains one login page and four main functional pages: “History”, “Share”, “Geocaching”, and “More”. Participants are allowed to register and login on the login page. On the page “History”, participants can view their history records of posted, found and failed-to-find geocaches. On the page “Share”, participants can chat with other users and post and view status updates. On the page “Geocaching”, participants are allowed to view geocaches’ information, search or post a geocache, set a target and mark the target. On the page “More”, participants can log out, edit the draft of a geocache to be posted, give feedback to the app and view the help center.

This user guide intended to teach users how to use this app. In chapter 1, a brief introduction of the app is given. In chapter 2, the requirements of the installation environment is stated. In chapter 3, the specific operations of how to use the app to participate in the activity of geocaching are introduced, which included the screenshots.

Chapter 2: Installation Environment

The app “GeocachingFun” is an Android smart phone app which needs the GPS service to realize some of the functionalities, e.g. showing the position of the user on a map. Therefore, users should download the APK(Android Package) of the app to a smart phone which has an Android operating system and is embedded with the GPS service. The Android operating system of the smart phone should be of a version above Android 4.0. Additionally, Android 5.0.2 and the above versions are recommended.

Chapter 3: Instructions on how to use the app

3.1 Register and login

In order to use the app, users need to register or login first. If users have never registered in the system, they should first register. As shown in Figure 1, users need to input a username and a password and then click the button “Register” in order to register in the system. If users have already registered, then they need to input the username and the corresponding password and then click the button “Log in”.

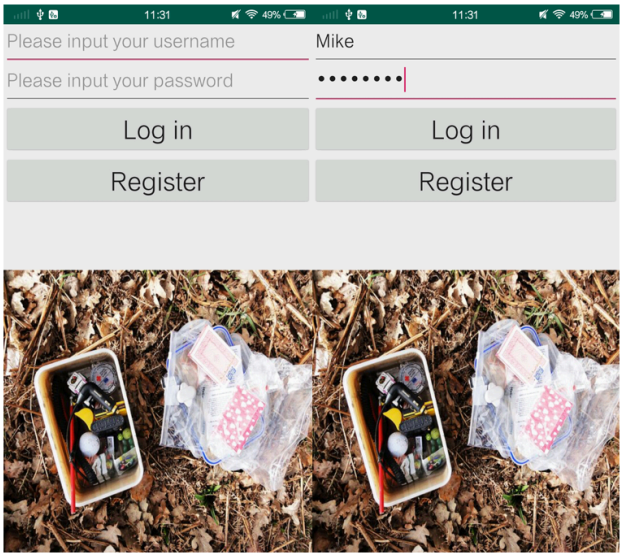

Figure 1 Register and login page

However, if users are trying to register and the inputted username already exists in the system, users may see a warning tag as shown in Figure 2. In this way, users need to re-input another username and the password and try again.

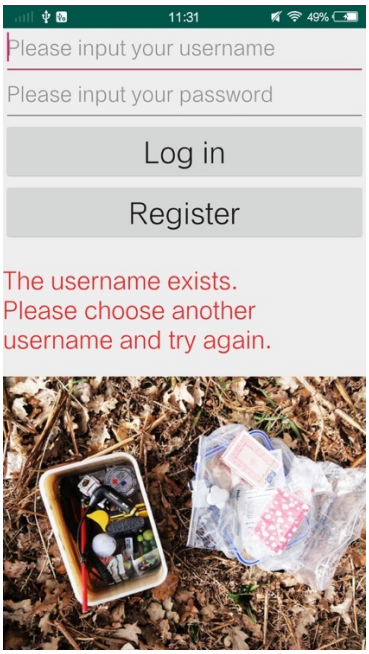

Figure 2 Register failure

While if users are trying to log in and the inputted username does not exist in the system or the inputted password is wrong, users may see warning tags as shown in Figure 3. In this way, users need to re-input the username and the password and try again.

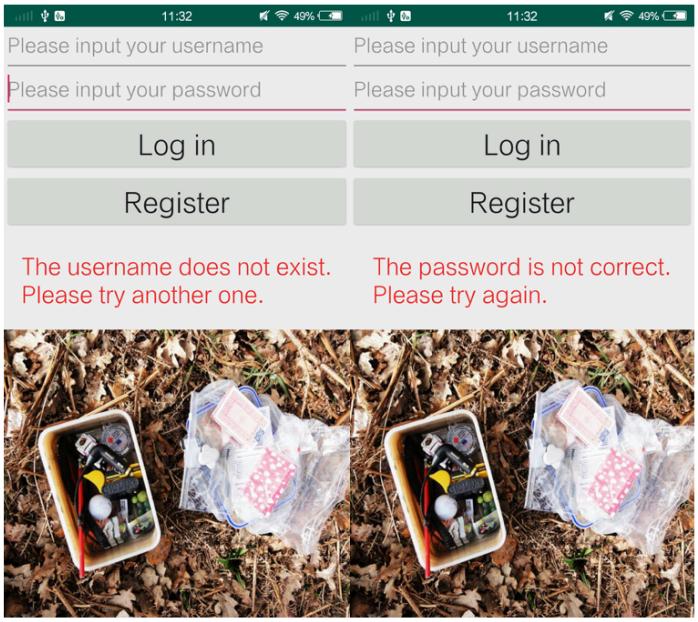

Figure 3 Login failure

If users register or log in successfully, the app will jump to the main page “History”.

3.2 View user information and the history records

In the page “History”, users can view their information like username, user level, joining date, and the number of the geocaches users have posted and tried to find. The user level is depended on the user’s total point. If users post a geocache, they will get 2 points, and if users try to find a geocache, they will get 1 point. The user level is 1 if the user’s total point is 0. The user level is 2 if the user’s total point is between 1 and 5. The user level is 3 if the user’s total point is between 6 and 20. The user level is 4 if the user’s total point is between 21 and 50. The user level is 5 if the user’s total point is between 51 and 100. If the user’s total point is beyond 100, the user level is 666, which means that the user has reached to a very high level.

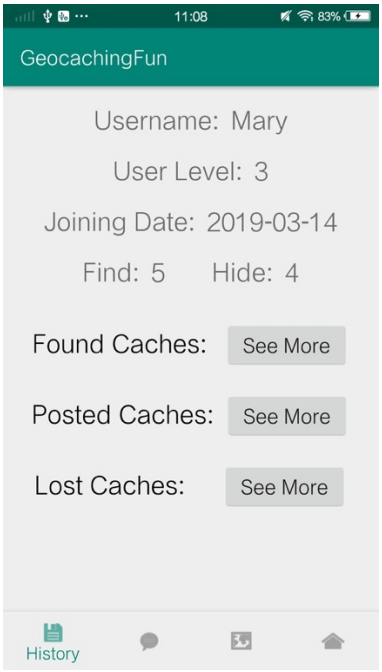

Figure 4 History page

Users can view the detailed information of the found, posted, and lost geocaches by clicking the corresponding button “See More” as shown in Figure 5.

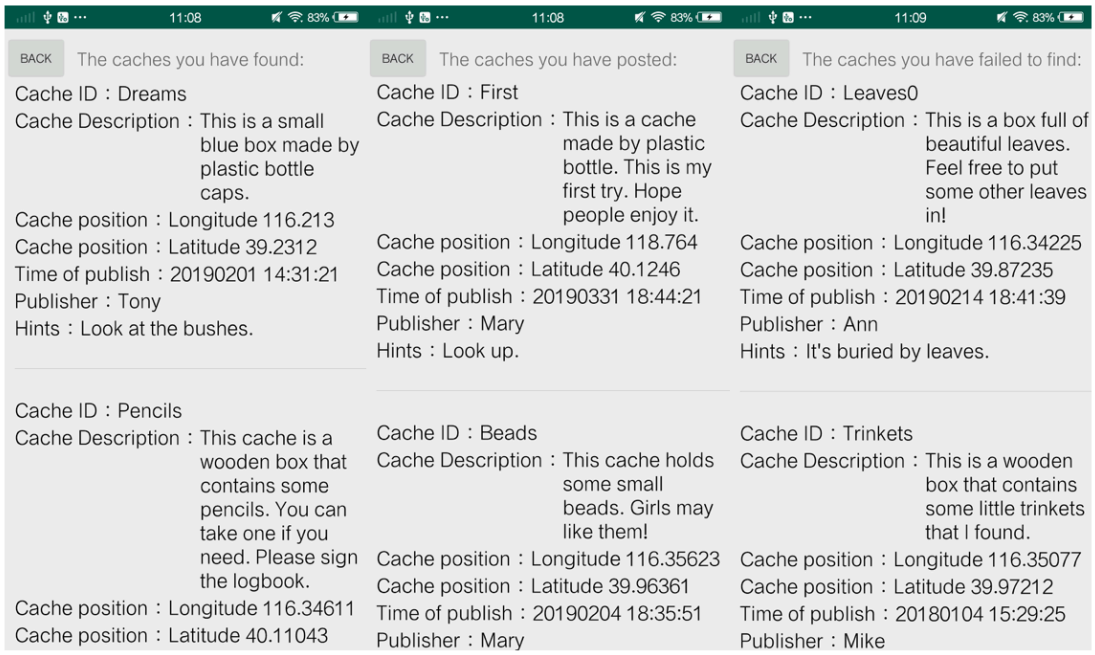

Figure 5 History records of the found, posted and lost geocaches

3.3 Chat with other users

Users can chat with others in the page “Share” by clicking the button “Chat”, as shown in Figure 6.

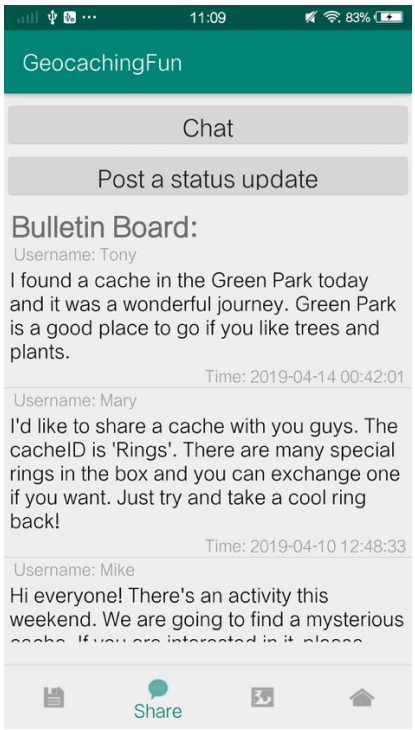

Figure 6 Share page

Users can either input a correct username or click one of the friend buttons to start a chat as shown in Figure 7. The system automatically adds the two users in a conversation to each other's friend list, therefore users do not need to input the same username again to start a chat next time.

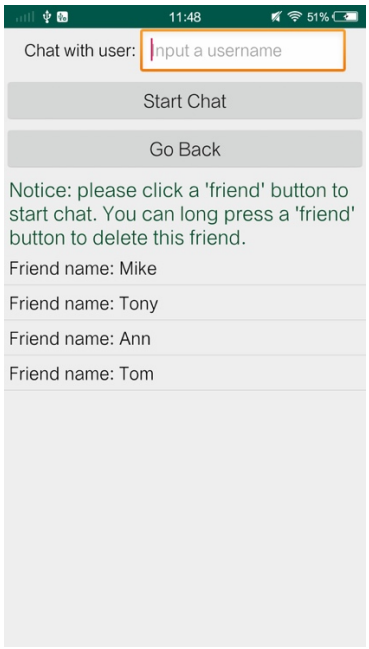

Figure 7 Chat with other users

If the inputted username of the user to talk to does not exist, the app will show a warning tag as shown in Figure 8. In this way, users need to input the correct username and try again.

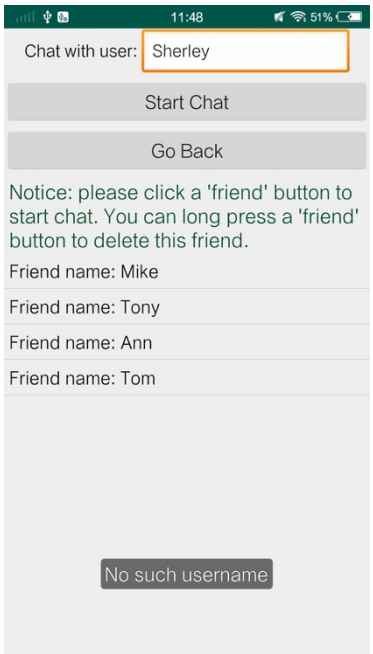

Figure 8 Wrong username

If users start a chat successfully, the app will show the chatting page as shown in Figure 9. Users can input the content of a message and then click the button “Send” to send the message to the friend.

Figure 9 Chatting page

The history messages are arranged by time order and the chatting page always slides to the latest messages. Once a message is sent, the app will automatically refresh the history messages on the chatting page.

3.4 View and post status updates

Users can view the status updates posted by other users on the bulletin board in the page “Share”, as shown in Figure 6. The status updates on the bulletin board are arranged in time order with the latest one on the top.

If users click the button “Post a status update”, they can input the content that they would to share with everyone and then click the button “Submit” as shown in Figure 10. Or, if they decide not to post this content, they can click the button “Cancel”.

Figure 10 Post a status update

3.5 Search and view geocaches and set a target

Users can view geocaches and search for a geocache if they click the button “View cache” in page “Geocaching” as shown in Figure 11. Each red pin on the map represents a geocache.

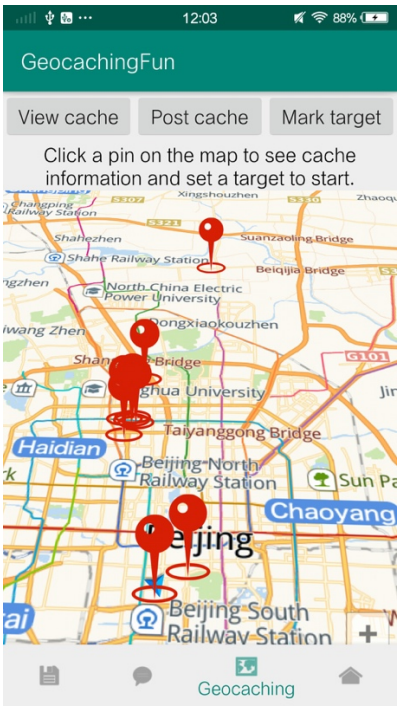

Figure 11 Geocaching page

Users can view the list of geocaches as shown in Figure 12.

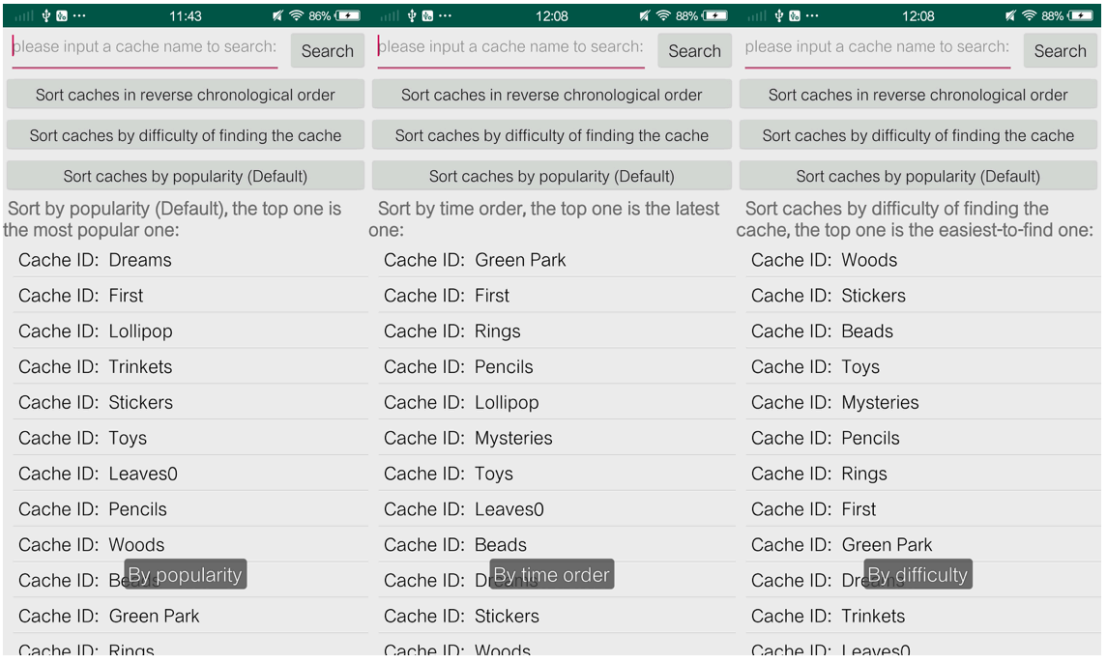

Figure 12 Search and view geocaches

The geocaches in the list are arranged by popularity by default. Users can click the button “sort caches in reverse chronological order” to rearrange the geocaches by time order, or click the button “sort caches by difficulty of finding the cache” to rearrange the geocaches by difficulty. The value of the difficulty of a geocache is the quotient of the total lost times that all users in the system have lost the geocache divided by the total times that all users in the system have set the geocache as a target. The value of the popularity of a geocache is the times that all users in the system have set the geocache as a target.

Users can also input a cache ID and click the button “Search” to search for a geocache. If the inputted cache ID is wrong, the app will show a tag like this: “Invalid cache name please try again”. If the inputted cache ID is empty, the app will show a tag like this: “Please don’t give empty ID”.

If Users input a valid cache ID and click the button “Search”, or if they click on one of the geocaches in the list, or if they click on one of the red pins on the map in Figure 11, they can view the detailed information of the geocache as shown in Figure 13.

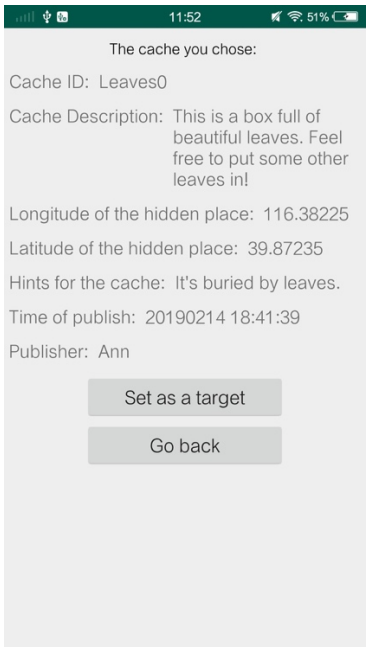

Figure 13 View geocache information and set a target

Users can set the geocache as a target to find by clicking the button “Set as a target”. Or, they could get back to the previous page by clicking the button “Go back” if they are not interested in this geocache. If a target is set, the app will jump to the page “Geocaching” which contains a map showing the position of the user by a small blue triangle and the position of the target geocache by a red pin as shown in Figure 14. Users can start a journey and use the map to navigate to the target.

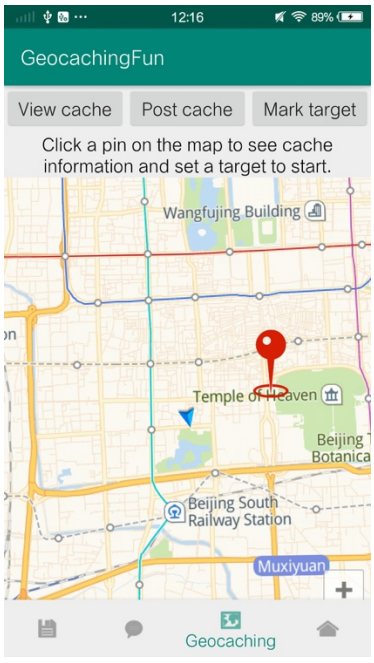

Figure 14 Positions of the target and the user

3.6 Post a geocache

Users can click the button “Post cache” in Figure 11 to post a geocache. As shown in Figure 15, users need to fill all the blanks and then click the button “Post” in order to post a geocache, otherwise, the system will show a tag like this: “You should fill all the blanks to post a geocache”. If the inputted cache ID already exists in the database, the system will show a tag like this: “Failed! The cacheID already existed! Please change the cacheID!”. Then users need to change the cache ID and try again until the system shows the tag: “Success! Your cache was posted. Thank you!”

Users need to at least fill the blank “CacheID” if they want to save a draft, otherwise, the system will show the tag: “Please don't give empty cacheID!”. Users should notice that each user can only save one draft in the system. If users click the button “Save as a draft” and the system shows the tag: “Failed! You had a draft in the system! One user can only save one draft.”, then users can either go back or fill all the blanks to post the geocache. If the draft is saved successfully, the system will show a tag: “Success! The draft was saved and you can edit it later!”.

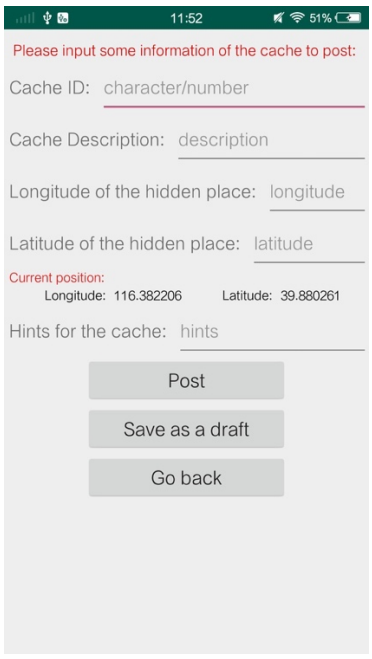

Figure 15 Post a geocache

3.7 Mark the target

Users can mark the target as found or lost by clicking the corresponding button in Figure 16

when the journey of finding the target is finished. If users haven't set a target before, the system will show the tag: "You haven't set any target!".

If users have found the cache before and click the button "Found", the system will show the tag: "You have found this cache before, so the database won't record it again.". If users have marked the geocache as lost before and click the button "Found", the system will show the tag: "You tried to find this cache and failed before. The database has been updated now.".

If users have found the cache before and click the button "Lost", the system will show the tag: "You have found this cache before, so the database won't record it as lost.". If users have marked the geocache as lost before and click the button "Lost", the system will show the tag: "You tried to find this cache and failed again. The database is not updated.".

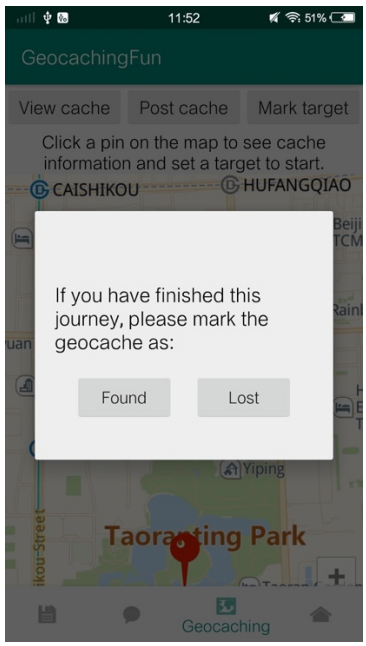

Figure 16 Mark the target

3.8 Edit the draft of a geocache to be posted

Users can click the button "Edit Draft" in Figure 17 to edit the draft of a geocache to be posted. While if users do not have a draft in the system, the system will show the following tag when the button "Edit Draft" is clicked: "Failed! You don't have a draft!"

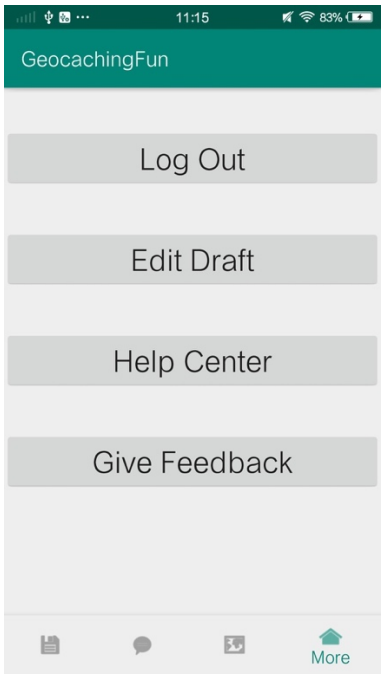

Figure 17 More page

If the button “Edit Draft” in Figure 17 is clicked, the page in Figure 18 will be shown, in which users can post the geocache, update the draft or go back.

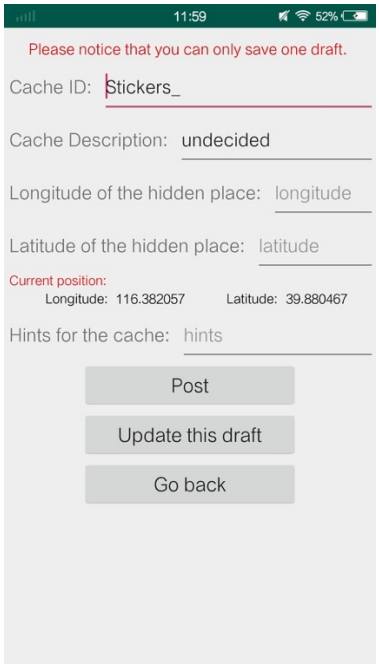

Figure 18 Edit the draft

3.9 View the help centre

The help center is a guide on how to participate in the activity of geocaching with the help of

the app “GeocachingFun”, therefore, users are recommended to view this part first, if they are novices. Users can view the help center if they click the button “Help Center” in Figure 17. There are four parts in the help center as shown in Figure 19: “what is geocaching”, “how to post a cache”, “how to find a cache” and “tips for geocaching”. If one of these four buttons is clicked, the system will show the corresponding description. Users can click the button “Go back” to get back to the previous page.

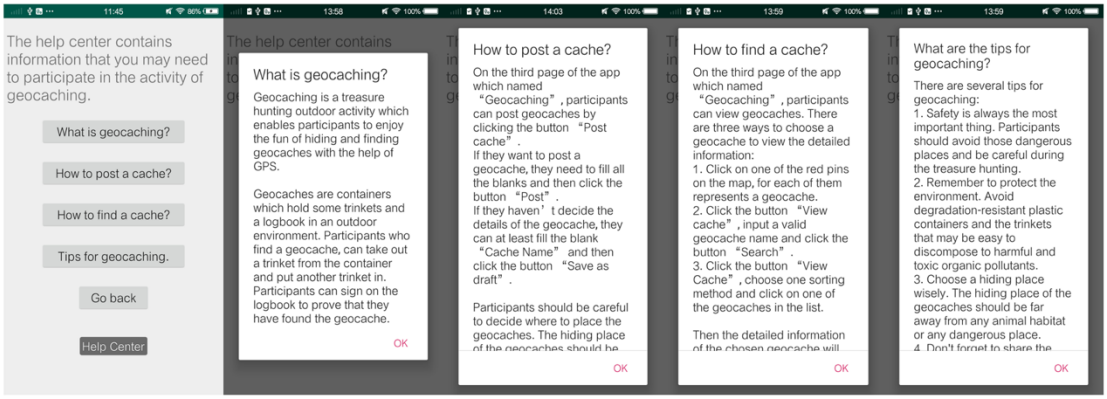

Figure 19 Help centre

3.10 Give feedbacks to the app

Users can click the button “Give Feedback” in Figure 17 to give a feedback to the system. As shown in Figure 18, users can input the feedback and then click the button “Submit” to submit the feedback. Users can also click the button “Go back” to get back to the previous page, in this case, the system will show the tag: “You can always give feedback when you are ready. Thank you!”. If the inputted content of the feedback is empty, the system will show the tag: “Please don't give empty feedback!”. If users successfully submit a feedback, the system will show the tag: “Success! Your feedback will be taken seriously. Thank you!”.

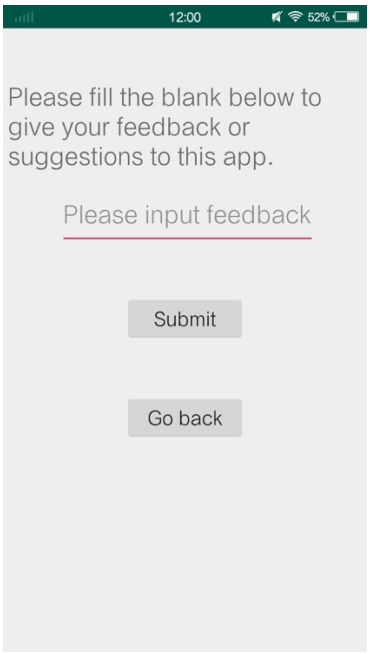

Figure 20 Give feedback to the app

3.11 Log out

Users can click the button “Log out” in Figure 17 to log out of the system. Then the system will show the login/register page again in case users want to log in again.

Risk Assessment

There could be risks preventing the successful completion of this project and the use of the system. There are mainly two types of risks, either something goes wrong with the system, or something happens to the developer. The system contains three parts: an Android app, a server and a database. Table 5 describes some risks that could affect the completion of the project, and the ratings and corresponding preventative actions. The ratings referred to the student handbook 3.6.13 risk assessment. “Likelihood Rating” in table 5 refers to level of likelihood and “Impact Rating” refers to ratings of risk and urgency of required action.

Table 5: Risk assessment

Description of Risk	Description of impact	Likelihood Rating	Impact Rating	Preventative Actions
Internet connection may fail	The app cannot work	Moderate	Moderate Risk	Develop some offline features, such as providing an offline map
The database may crash	The system cannot work, cause data inconsistency between the app and the database	Rare	Low Risk	Make a copy of the database and rerun it, warn the users that data may lost during the crash
The server may crash	The system cannot work and the HTTP requests cannot be answered	Rare	Low Risk	Make a copy of the server and rerun it, give feedback to the users
The smart phone may get stuck	The app cannot run successfully	Rare	Low Risk	Do more testing and rerun the app
The developer’s computer may be broken	The development has to stop temporarily	Rare	No Risk	Make copies of the programs

In conclusion, there are little risks in developing and running the system as shown in Table 5. Some preventative actions can be taken to avoid these risks, such as taking backups, running more tests and working ahead of time.

Environmental Impact Assessment

There can be some impacts to the environment during the development of the system and during people participating the activity of geocaching.

First, there can be a large amount of electricity consumption during the development of the system. In this project, a computer was used to develop the app, the server and the database, and an Android smart phone was used to run the testing. Therefore, electricity was used to charge the computer and the smart phone. One way to reduce the consumption is to reduce the time of testing, which requires the developer to have a clear logic and to avoid bugs in the development. In this project, the developer tried to avoid meaningless testing and tried to reduce the time of debugging by following a clear logic.

Second, people can cause energy consumption when participating the activity of geocaching. For example, people may cause electricity consumption when charging their smart phones since GPS is an energy-consuming service. Additionally, people may cause oil consumption when driving to a specific park or some other places where hid the geocaches. However, the activity of geocaching should not be avoided because of the energy consumption, while some suggestions can be given to the participants. For example, participants are encouraged to find a geocache nearby, or to ride bicycles to get to the places where hid the geocaches.

Third, geocaching participants may raise waste disposal and recycling issues. In the activity of geocaching, participants are allowed to hide geocaches in the nature environment, which are containers that include a logbook and some trinkets. If participants use degradation-resistant plastic boxes as containers, they can do great harms to the environment. Or, if the participants choose some trinkets that are easy to discompose to harmful and toxic organic pollutants, they can do great damage to the plants and the animals in the nature. Therefore, participants are encouraged to avoid plastic containers and other degradation-resistant containers. When choosing the material of the containers, wooden boxes are always the best. Participants are also warned not to put dangerous or harmful trinkets into the containers, and they should improve their skills of choosing the places to hide the geocaches, for the places should be far away from any animal habitat. Additionally, the developer or the manager of the system should make regular inspections and monitoring on the materials of the containers and the places where put the geocaches.