

# Manual for Travelling Counsellor System

## Contents

1. Introduction .....	1
a) About the Manual .....	1
b) Background .....	2
2. System Overview .....	2
a) Objective .....	2
b) Function .....	2
3. User Instructions.....	5
a) Installing the app .....	5
b) Registration and Login .....	5
1. Rule of registration.....	5
2. Procedure of login .....	6
3. Filling Travel preference form .....	7
c) Travel Planning.....	8
4. Administrator Guide .....	14

## 1. Introduction

### a) About the Manual

This manual is written as a guidance for the user and administrator to use this Travel Counsellor System. In this manual, the reader can understand all the system functions, their access of permissions and methods to operate the system. The administrator can learn from this

manual to maintain the system.

### **b) Background**

This system is developed by Group 15 for the course COMP208 group software project.

## **2. System Overview**

### **a) Objective**

This system is aimed to help inexperienced travelers to plan a travel.

The user inputs the starting point and travel duration, the system will provide an ultimate solution based on the preference recorded from the preference questionnaire and the actual situation of tourist attractions and cities will be taken into the account.

### **b) Function**

1. Login and registration
2. Select travel date, starting point and destination
3. Select travel preferences
4. Recommend travel routes

### c) Permissions Required for the System

```
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
<uses-permission android:name="android.permission.READ_PROFILE" />
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

Figure 1 User permission

This system requires permission from the device as follows:

#### 1. GET\_ACCOUNTS

This permission will be asked after installation and popped out at the first build.

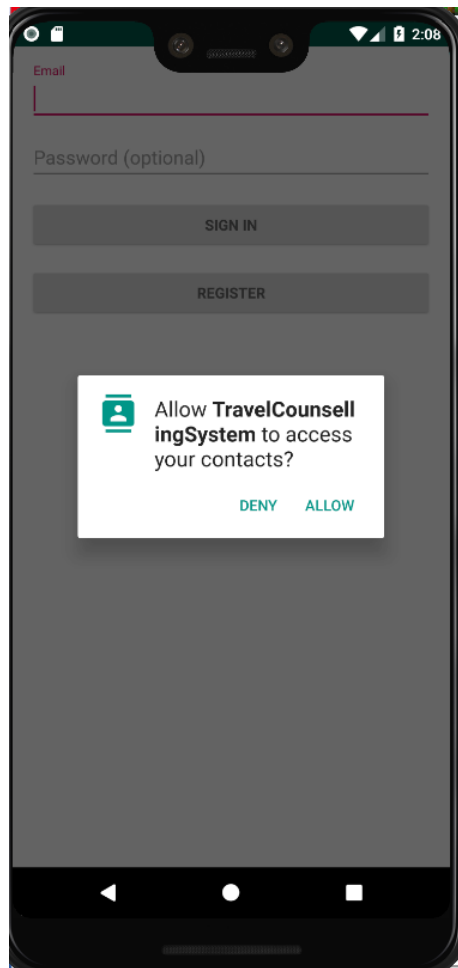


Figure 2 On-the-Fly Permission

## 2. READ\_PROFILE

This permission will read user profile of the device.

## 3. READ\_CONTACTS

This permission will access to the contacts stored in the device.

## 4. INTERNET

This permission will grant the software the access to internet, which is essential to access to the database for login and registration and route recommendation powered by Google Map API

## 5. ACCESS\_FINE LOCATION

This permission will access to the exact location of the device.

#### 6. READ\_EXTERNAL\_STORAGE

This permission will access to the external storage of the device, for potentially storing extra information for the system.

#### 7. WRITE\_EXTERNAL\_STORAGE

This permission enables the system to record changes for the system in the external storage.

### **3. User Instructions**

#### **a) Installing the app**

The APK of the system is provided and the software is available at Google Play and can be downloaded free.

#### **b) Registration and Login**

The user interface of the registration and login are presented as follows.

##### **i. Rule of registration**

1. If either the username or password cannot be null.
2. The password must be last least eight characters long.

3. The password must contain at least one lower case letter.
4. The password must contain at least one upper case letter.
5. The password must have a special character one of the following: ! “ £ \$ % ^ & \* ( )

## ii. Procedure of login

User should first sign up an account using email account and follow the rules to create the account to proceed to the questionnaire interface. Registered user can sign up directly by inputting email and password to access to the main page.

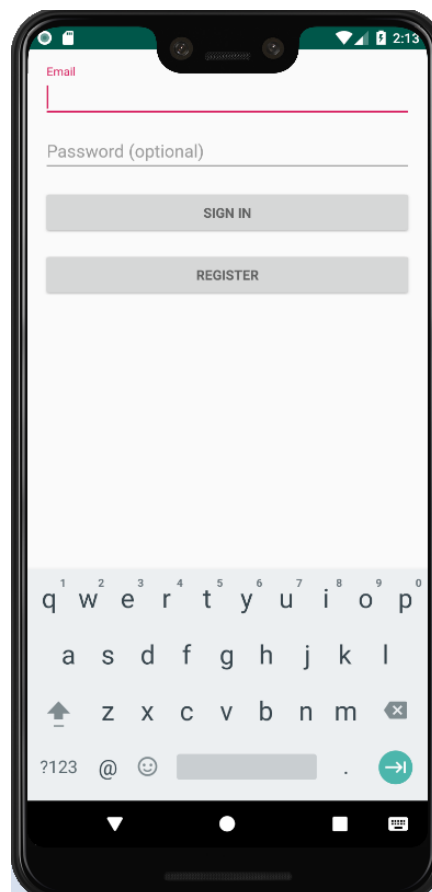
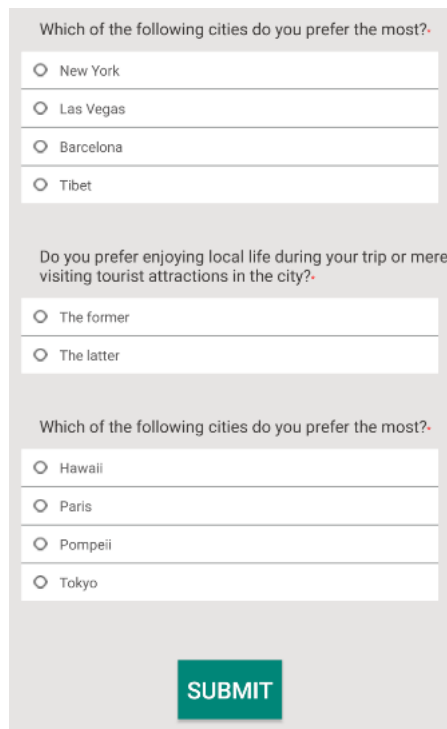


Figure 3 Login Interface

### iii. Filling Travel preference form

The Travel preference form contains five questions, One Yes/No question, three multiple-choice questions and one question that allows users to choose all the choices.



Which of the following cities do you prefer the most?\*

- ☐ New York
- ☐ Las Vegas
- ☐ Barcelona
- ☐ Tibet

Do you prefer enjoying local life during your trip or merely visiting tourist attractions in the city?\*

- ☐ The former
- ☐ The latter

Which of the following cities do you prefer the most?\*

- ☐ Hawaii
- ☐ Paris
- ☐ Pompeii
- ☐ Tokyo

**SUBMIT**

Figure 3 Registration Questionnaire

## c) Travel Planning



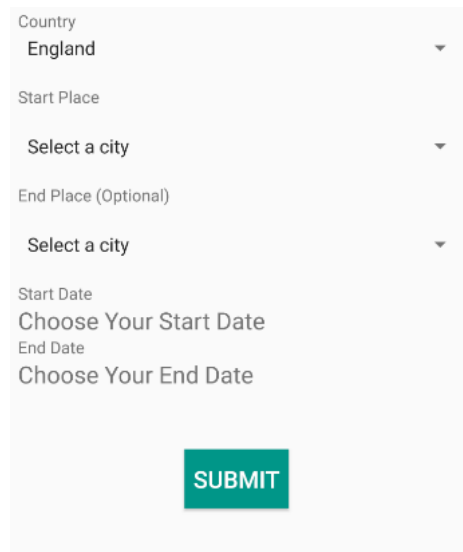
Figure 4 Main Page

The figure shown above is the main page of this system. The user should click “Start Your Journey” button to proceed to the travel-planning interface.

The travel-planning interface requires user to select country, start place and optionally end place of travel via drop-down menus.

After filling out the form, the system will proceed to next interface.



A travel planning form with a light gray background. It contains several dropdown menus and date selection fields. The first dropdown is labeled 'Country' and has 'England' selected. The second dropdown is labeled 'Start Place' and has 'Select a city' selected. The third dropdown is labeled 'End Place (Optional)' and has 'Select a city' selected. Below these are two date selection fields: 'Start Date' with the text 'Choose Your Start Date' and 'End Date' with the text 'Choose Your End Date'. At the bottom right is a red rectangular button with the word 'SUBMIT' in white capital letters.

Country  
England

Start Place  
Select a city

End Place (Optional)  
Select a city

Start Date  
Choose Your Start Date

End Date  
Choose Your End Date

SUBMIT

Figure 5 First Selection for Travel Planning

#### d) City Selection

User can select their desired destination via this interface to add in the route recommendation. After the choice, the user can click the red button to submit to access to the route recommendation.

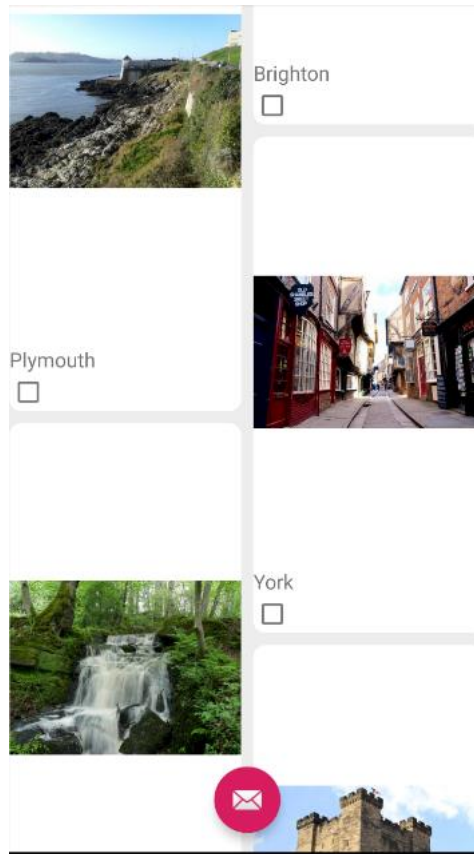


Figure 7 Second Selection for Travel Planning

### e) **Route recommendation**

The route is automatically provided by the map API, user can click button to visit feedback section and return to the main page via two different buttons.

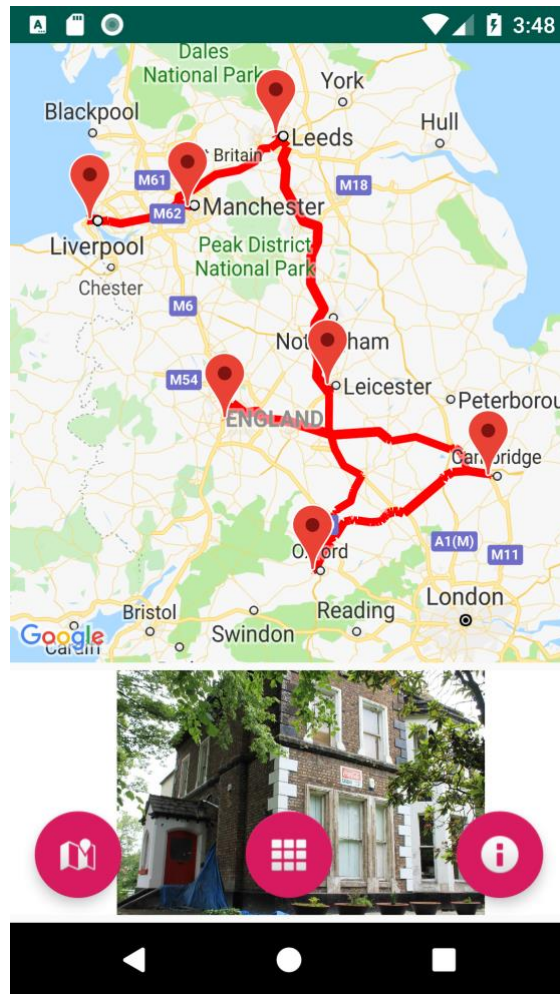


Figure 8 Route Recommendation interface

As the figure shown above, the recommended route will be displayed as red line and each stop will be represented by red markers. The map could be slide to left, right, top, or down, as well as zoom in or zoom out like operating google map. The bottom half of the page shows tourist attractions of the cities. User can swipe left or right to view different tourist attractions.

In addition, user can save the recommended route as screenshot by clicking the bottom-left button. By clicking the bottom-middle

button, the user could return to the home page to make another travel plan. By clicking the bottom-right button, the user may be required to fill in a feedback form which could help developer for further updating.

#### **f) Feedback**

User can also access to feedback questionnaire via navigation bar at the top left corner of the screen.

After the feedback questionnaire was submitted, the system will return to the interface which the user clicks the link to feedback section.

# Feedback

## User Preference

Is This your first time using this system or similar travel planning apps\*

☐ Yes

☐ No

What is your age?\*

☐ 0-16

☐ 16-28

☐ 28-40

☐ 40-60

☐ 60+

How do you know this app?\*

☐ From Google Play Store

## **4. Administrator Guide**

### **a) Database Maintenance Guide**

DB Browser for SQLite software is used for creating cityi and information database. For cityi database, table travelcity is saved in database and has six columns which are cityname, day, scenery, historical, entertaining and location.

Cityname column saves name of city which will be considered in this application and is also primary key of travelcity table. Day column includes recommending date for traveling specified city which only contains integer value. Scenery column contains string value which is either “Y” or “N”. “Y” represents that city is famous for scenery attractions and vice versa. Historical column includes string value “Y” and “N” which represents whether city has famous historical attractions. Entertaining column includes string value “Y” and “N” in order to represent that whether specified city owns famous entertainment attractions. The location column represents latitude and longitude of specified city (figure one represents travelcity table)


There are twenty tables in the information database which each city has relevant table in the database. Table of each city owns four columns which are attractions, about, figure and rate. For attractions column, it represents five famous attractions in the city. For about column, it includes information of each attraction. The figure column includes figure of attraction and rate column includes rate for specified attractions which only real number is allowed to be inserted into column.

▼ 表 (20)	
▶ Bath	CREATE TABLE "Bath" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Birmingham	CREATE TABLE "Birmingham" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Brighton	CREATE TABLE "Brighton" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Bristol	CREATE TABLE "Bristol" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Cambridge	CREATE TABLE "Cambridge" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Canterbury	CREATE TABLE "Canterbury" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Leeds	CREATE TABLE "Leeds" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Leicester	CREATE TABLE "Leicester" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Liverpool	CREATE TABLE "Liverpool" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ London	CREATE TABLE "London" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Lsle of wight	CREATE TABLE "Lsle of wight" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Manchester	CREATE TABLE "Manchester" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Newcastle	CREATE TABLE "Newcastle" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Norwich	CREATE TABLE "Norwich" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Nottingham	CREATE TABLE "Nottingham" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Oxford	CREATE TABLE "Oxford" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Plymouth	CREATE TABLE "Plymouth" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Sheffield	CREATE TABLE "Sheffield" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ Southampton	CREATE TABLE "Southampton" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)
▶ York	CREATE TABLE "York" ( "attractions" TEXT NOT NULL UNIQUE, "about" TEXT NOT NULL, "figure" BLOB NOT NULL, "rate" REAL NOT NULL)

Figure 10 Database Directory

	cityname	day	scenery	historical	entertaining	location
	过滤	过滤	过滤	过滤	过滤	过滤
1	London	5	Y	Y	Y	51.5287718,-0....
2	Cambridge	1	Y	Y	N	52.1989266,0.0...
3	Oxford	1	Y	Y	N	51.7505018,-1....
4	Manchester	2	N	N	Y	53.4723272,-2....
5	York	2	Y	Y	N	53.9586419,-1....
6	Bath	1	Y	Y	N	51.3801748,-2....
7	Birmingham	2	N	N	Y	52.477564,-2.0...
8	Liverpool	2	Y	N	N	53.4123001,-3....
9	Brighton	2	Y	N	Y	50.8374669,-0....
10	Bristol	1	Y	N	N	51.468575,-2.6...
11	Leeds	1	N	N	Y	53.8060835,-1....
12	Sheffield	1	N	N	N	53.3958394,-1....
13	Newcastle	1	Y	Y	N	55.0024505,-1....
14	Nottingham	1	N	N	N	52.9541053,-1....
15	Southampton	1	N	N	N	50.913908,-1.4...
16	Norwich	1	Y	N	N	52.6402265,1.2...
17	Leicester	1	N	N	Y	52.6361606,-1....
18	Canterbury	1	Y	Y	N	51.2785924,1.0...
19	Isle of wight	2	Y	N	Y	50.6712436,-1....
20	Plymouth	3	Y	N	Y	50.3885326,-4....

Figure 11 Snapshot of the city table in database

表:  Bath 

	attractions	about	figure	rate
	过滤	过滤	过滤	过滤
1	bath abbey	Begun in 1499 ...	BLOB	5.0
2	city farm	Bath City Farm i...	BLOB	4.5
3	soft cheese	At Park Farm we...	BLOB	5.0
4	botanical garden	Located on the ...	BLOB	4.5
5	solisbury hill	Believed to be t...	BLOB	4.5

Figure 12 Snapshot of the table Bath in database

For operating database, sqlite database can still use SQL command.

In order to select column from database, the actual code to perform is:



```

Cursor cursor = database.rawQuery( sql: "select figure from Bath", selectionArgs: null);
while (cursor.moveToNext()){
    System.out.println(cursor.getString(cursor.getColumnIndex( columnName: "attractions")))
}

```

Figure 13 Code Example of Using SQL Command to Maintain Database

Moreover, the table of information database has figure column, which owns figure of attractions. The method of selecting picture is different with selecting data of other data types.

```

private ArrayList<Drawable> getDrawable(Cursor cursor){
    ArrayList<Drawable> drawables = new ArrayList<>();
    if(cursor != null && cursor.getCount() != 0) {
        while(cursor.moveToNext()) {
            byte[] img = cursor.getBlob(cursor.getColumnIndex( columnName: "figure"));
            Bitmap bitmap = BitmapFactory.decodeByteArray(img, offset: 0, img.length, opts: null);
            BitmapDrawable bitmapDrawable = new BitmapDrawable(bitmap);
            Drawable drawable = bitmapDrawable;
            drawables.add(drawable);
        }
    }
    return drawables;
}

```

Figure 14 Code Example of Using SQL Command to Database maintenance

```

ImageView ivv=new ImageView( context: this);
ArrayList<Drawable> drawables = getDrawable(cursor);
ivv.setImageDrawable(drawables.get(0));
setContentView(ivv);

```

Figure 15 Code Example of Storing figure to the Database

The figure is saved as blob type of data in the database. In order to extract from database and represent it, it is essential to use cursor.getBlob method to get byte array. Then transform byte array img[] to drawable data type. Create imageview class to initialize instance, use setImageDrawable and setContentView(ivv) method to



Figure 16 Example of Stored Figure from Database

## **b) Testing guide**

During the testing process, virtual Android devices are available in Android studio. Most of the testing and debugging process were processed on Android virtual devices and the built APK should be tested on physical Android devices. Boundary conditions of the input field should be taken in the account and tested . Automated test were provided for RegistrationHelper class to validate account and password and automated test cases will be available in the further version of software in the course of optimization and development.