

**PROJECT REPORT FOR PRT452 SOFTWARE ENGINEERING: PROCESS AND TOOLS ON RANDOM PATROLLING**

|  |  |  |
| --- | --- | --- |
| **Name** | **Role** | **Contact Details** |
| Mr. Siva Vallabhaneni | Product Owner and Client | siva.vallabhaneni@cdu.edu.au |
| Dr. Pater Shaw | Subject Faculty | Peter.shaw@cdu.edu.au |
| Aayush Sapkota | Team Member | [96.estd@gmail.com](mailto:96.estd@gmail.com) |
| Labin Sapkota | Team Member | [loveinspk@gmail.com](mailto:loveinspk@gmail.com) |
| Saugat Pathak | Team Member | 306457@students.cdu.edu.au |
| Yunesh Timalsina | Team Member | Yunesh.timalsina@students.cdu.edu.au |

**Acknowledgement**

Team would like to express special gratitude towards our Client Mr. Siva Vallabhaneni who showed a confidence on our abilities and gave us opportunity to work on his idea. His constant feedback on our work helped us a lot to improve the final outcome.

We are grateful towards our coordinator Dr. Peter Shaw, who enlightened our path towards knowledge and information. Dr. Peter was continuing source of support and knowledge and his guidance helped the entire team to complete the project on time.

Contents

[History of Game theory 4](#_Toc495696101)

[1) Project Information 4](#_Toc495696102)

[1) Project Overview 4](#_Toc495696103)

[2) Purpose of the project 5](#_Toc495696104)

[3) Objective and Goals 5](#_Toc495696105)

[2) Requirement 5](#_Toc495696106)

[2.1 Functional Requirement 5](#_Toc495696107)

[2.2 Non- Functional Requirement 6](#_Toc495696108)

[3) UI Design 7](#_Toc495696109)

[4) Database Design 15](#_Toc495696110)

[5) UML Design 15](#_Toc495696111)

[a) Use Case Diagram 15](#_Toc495696112)

[b) Sequence Diagram 16](#_Toc495696113)

[c) Activity Diagram 18](#_Toc495696114)

[d) Class Diagram 19](#_Toc495696115)

[6) Testing 20](#_Toc495696116)

[7) Design Pattern 22](#_Toc495696117)

[8) Pair Programming 22](#_Toc495696118)

[9) Story Cards 22](#_Toc495696119)

[10) Agile Process 38](#_Toc495696120)

[11) Security 39](#_Toc495696121)

[Handling User Data 42](#_Toc495696122)

[Handling User Credential 42](#_Toc495696123)

[12) Code 43](#_Toc495696124)

[13) Reference 96](#_Toc495696125)

# History of Game theory

Game theory is the study of mathematics model and is mainly used in psychology, political science, economics, computer science and biology. Game theory began with the idea regarding the existence of mixed strategy equilibria in two people zero sum games and its proof by John von Neumann. Game theory is the study of human conflict and cooperation within a competitive situation. In some respects, game theory is the science of strategy, or at least the optimal decision-making of independent and competing actors in a strategic setting. The key pioneers of game theory were mathematicians John von Neumann and John Nash, as well as economist Oskar Morgenstern. [x]

Game theory creates a language and formal structure of analysis for making logical decisions in competitive environments. The term “game” can be misleading. Even though game theory applies to recreational games, the concept of “game” simply means any interactive situation in which independent actors share more-or-less formal rules and consequences. [x]

The formal application of game theory requires knowledge of the following details: the identity of independent actors, their preferences, what they know, which strategic acts they are allowed to make, and how each decision influences the outcome of the game. Depending on the model, various other requirements or assumptions may be necessary. Finally, each independent actor is assumed to be rational. [x]

# Project Information

## Project Overview

Random Patrolling is an android application. With this application the user is able to locate his/her next station to visit. This application helps the user which station to visit next, total number of station to visit within the given time, number of times he/she have to visit the station in the allocated time. This application is based on travelling sales person algorithm which generates the routes based upon the shortest distance between the stations. The application regenerates the route if there is alert in between.

## Purpose of the project

This project is designed for those who want to visit the different places with certain time limitation and do their work. With the help of this project one can find the shortest distance between their stations. The main purpose of this project is that the user can find the shortest routes between the stations and can find the route to the station through the Google map inbuilt in the application. The guidelines like the number of station he/she have to visit in his working time, number of visits in each station, minimum and maximum time one has to spent in each station is provided in the application. The admin or the organization is allowed to add all the above information and the user can view it through his login credential. If some obstacles occur in between his route like some alarm occurs in some station and required the immediate present of the user then according to his location the path reroutes automatically based upon his visit history on the vary day.

## Objective and Goals

The objective of completing this project is to learn about various software development process and the challenge and obstacles one can face during the development of application. After the successful completion of the project one can learn following things.

* To analyse and examine the process involved software system engineering.
* To build the computer solution for real life based problems.
* To conduct a series of test on software at various levels.
* To evaluate testing methodologies critically through research reports.
* To develop the personal skills and find the way to mingle with the group members of software engineering teams.

# Requirement

## 2.1 Functional Requirement

Function requirements are requirements which show the functionality of the application. Functionality and service which are provided by application in order to meet user requirement is called functional requirements.

* Application should allow user to login using their credential.
* User should be able to add places and view the places they have added.
* After the number of station finalized user must be able to see the places they have to visit in the Google map with the direction.
* After visiting the certain station, they can push up the message that they have visited it.
* They can push the alert alarm if they find anything suspicious and thinks it need to be taken an extra care.

## 2.2 Non- Functional Requirement

The requirement that improves the user flexibility to use the application is known as non-functional requirement.

* Our application runs on android version on or above API 19 (Android 4.4 KitKat).
* The login secures the user data and that can only be viewed by them.
* The application opens in about 3 to 4 seconds.
* The user interface is user friendly and can be easily used by the user without any difficulties.

# UI Design

The Various windows of our app are listed below.

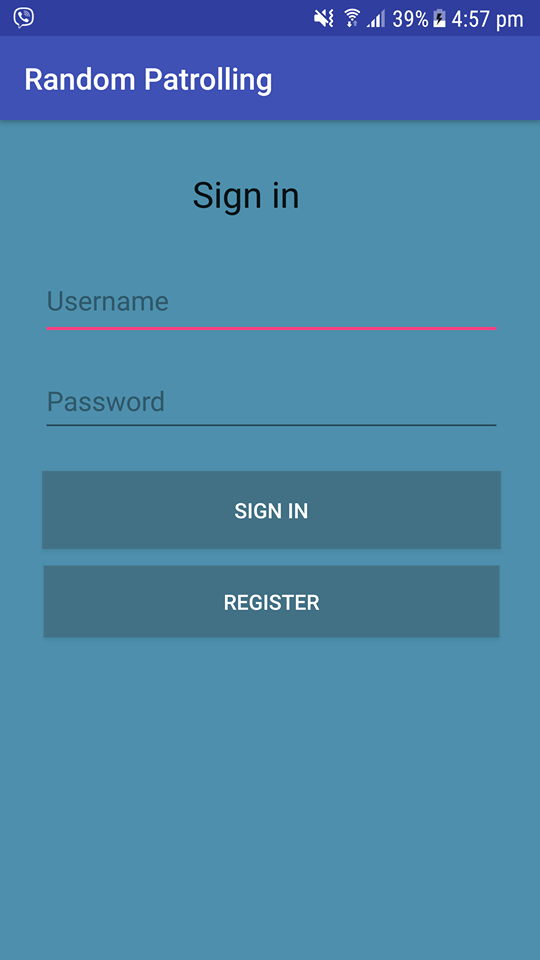


Figure: Login Page

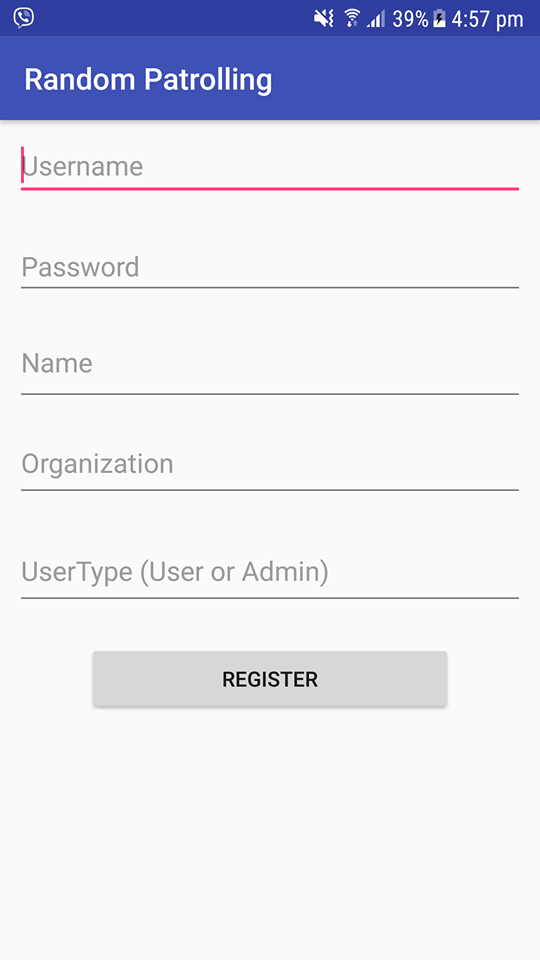


Figure: Registration Page

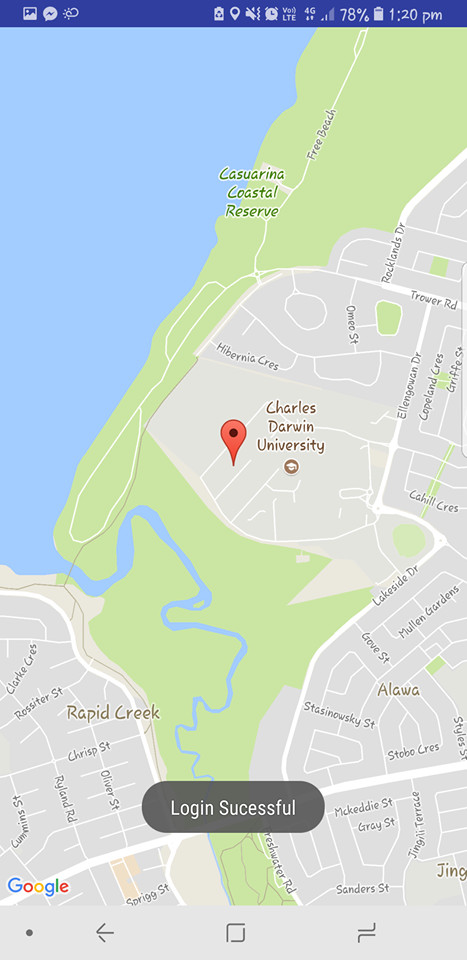


Figure: Successful Login Page

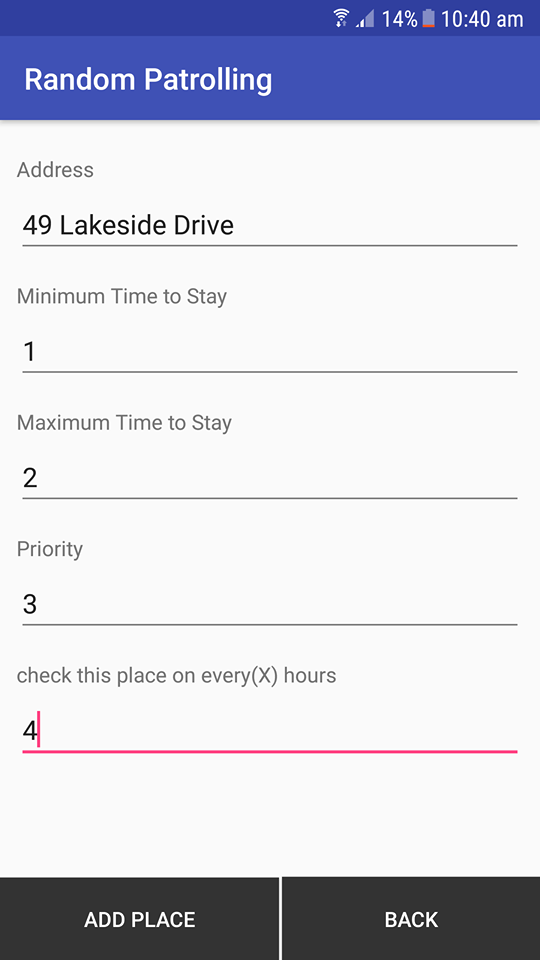


Figure: Add Place Page



Figure: Page for after add Place

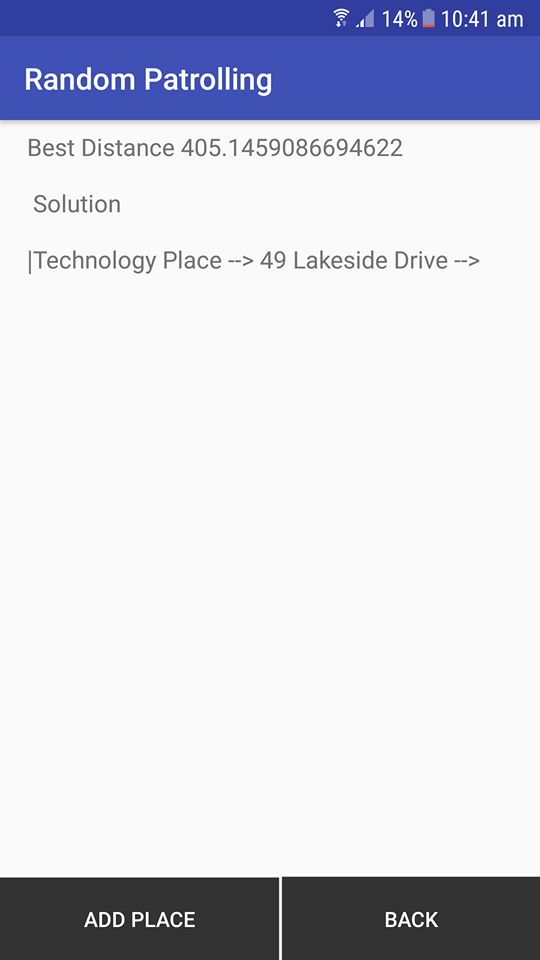


Figure: Calculate Distance Page

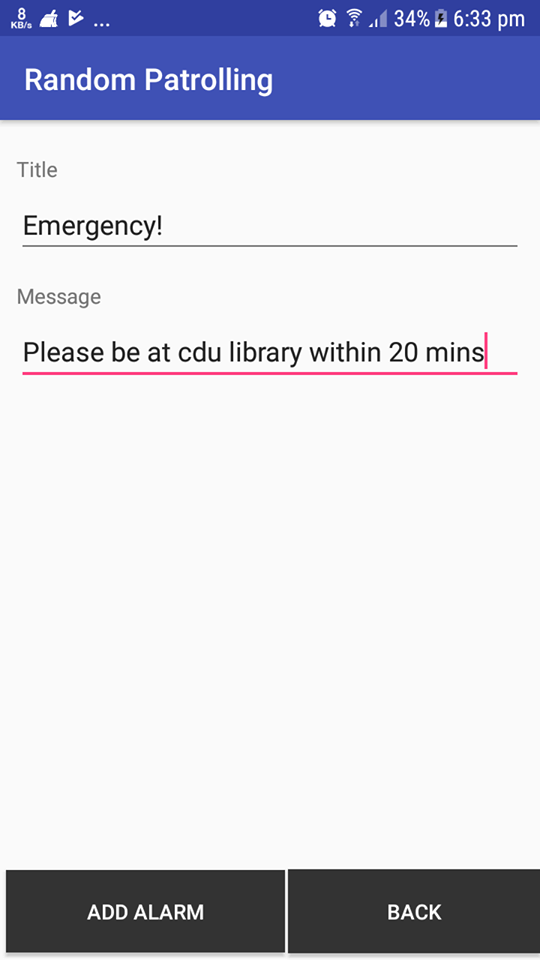


Figure: Add Alarm Page

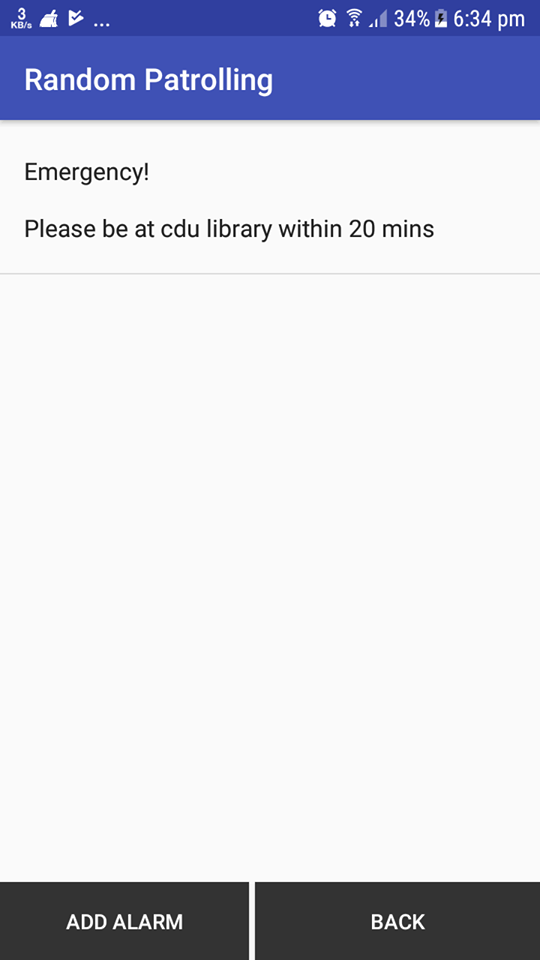


Figure: After adding Alarm Page

# Database Design

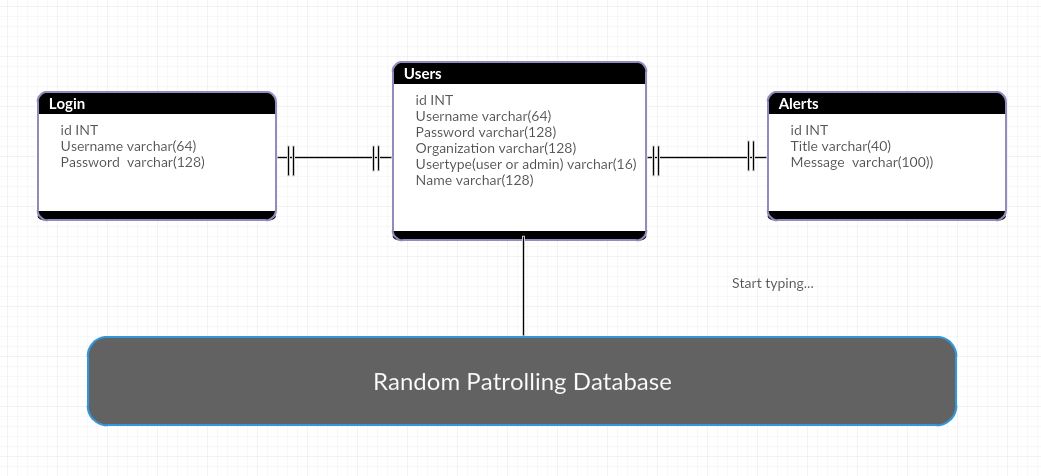


Fig: Database Design

# UML Design

## Use Case Diagram

Use Case Diagrams is also referred as behaviour diagrams and it describes a set of actions that a system or user of that system can perform alone or with the collaboration of another external user. In our application a user can login into the app with the login credential. After the user logins he can perform a set of actions like adding places that are to be visited, view the places that are added and the shortest route to visit those places. The routes will be shown on the Google map.

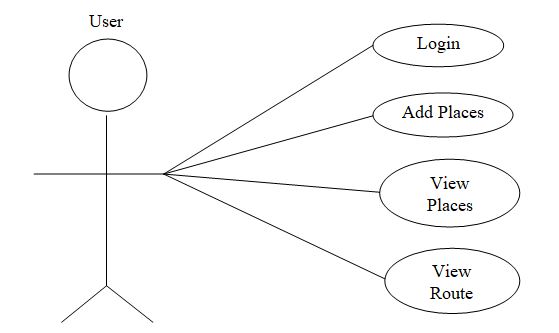


Fig: Use Case Diagram

## Sequence Diagram

Sequence diagrams are also known as event diagrams or event scenarios. This diagram shows that how an object operates with another object and in which sequence. The flow of the application or project in the continuous flow is shown through sequence diagram. In our project at first the user enters into the app by logging into it. The username and password is checked into the database. If not matched then the username and password not matched message is displayed in pop up. If matched then the user will add the places he/she have to visit. These places are saved into the database. Then the user can view the places added and routes to visit those places.

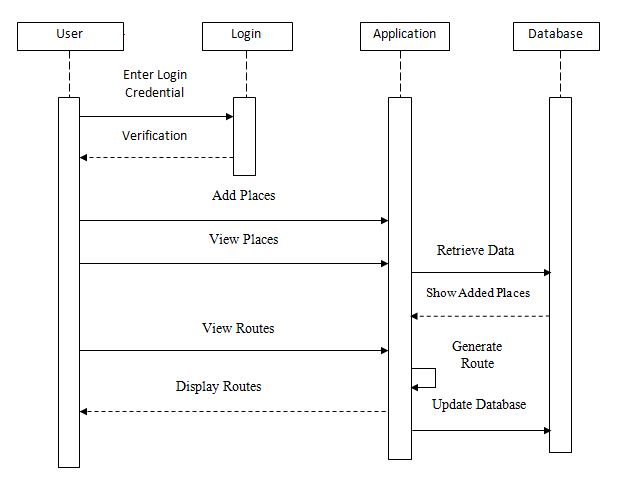


Fig: Sequence Diagram

## c) Activity Diagram

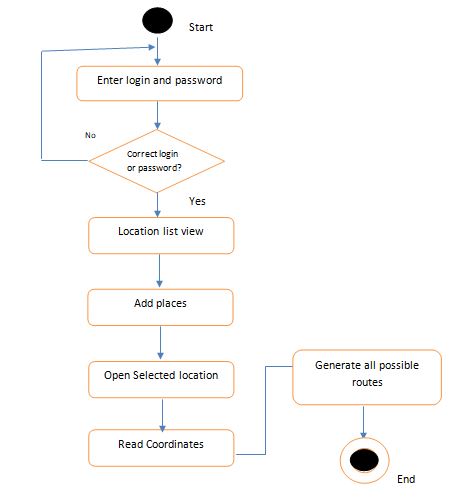


Fig: Activity Diagram

Fig: Activity Diagram

## d) Class Diagram

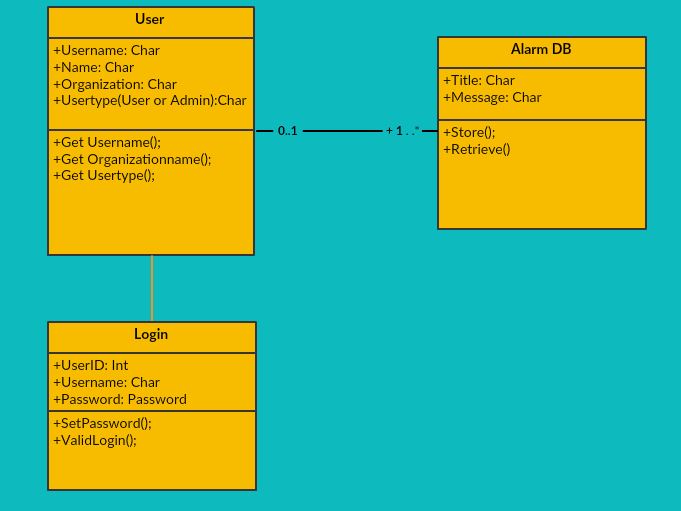


Fig: Class Diagram

# Testing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Case Id | Test Scenario | Test Steps | Test Data | Expected Result | Actual Result | Pass/Fail |
| T01 | Check User Login with valid data | 1. Open App  2. Enter username  3. Enter Password  4. Click login | User Id: yunesh  Password: \*\*\*\*\*\* | Login successful | Login Failed | Fail |
| T02 | Check User Login with valid data | 1. Open App  2. Enter username  3. Enter Password  4. Click login | User Id: yunesh  Password: \*\*\*\*\*\* | Login successful | Login successful | Pass |
| T03 | Check User Login with invalid data | 1. Open App  2. Enter username  3. Enter Password  4. Click login | User id: chill  Password:  \*\*\*\*\*\* | Login unsuccessful | Login successful | Fail |
| T04 | Check User Login with invalid data | 1. Open App  2. Enter username  3. Enter Password  4. Click login | User id: chill  Password:  \*\*\*\*\*\* | Login unsuccessful | Login unsuccessful | Pass |
| T05 | Add Places with valid location | 1. Open App  2. Login  3. Click Add Places  4. Long tap on map | Charles Darwin University | Place Added | App crashed | Fail |
| T06 | Add Places with valid location | 1. Open App  2. Login  3. Click Add Places  4. Long tap on map | Charles Darwin University | Place Added | Place Added | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T07 | Add Places with invalid location | 1. Open App  2. Login  3. Click Add Places  4. Long tap on map | Clicked in Sea | Do Nothing | Place Added | Fail |
| T08 | Add Places with invalid location | 1. Open App  2. Login  3. Click Add Places  4. Long tap on map | Clicked in Sea | Do Nothing | Do Nothing | Pass |

# Design Pattern

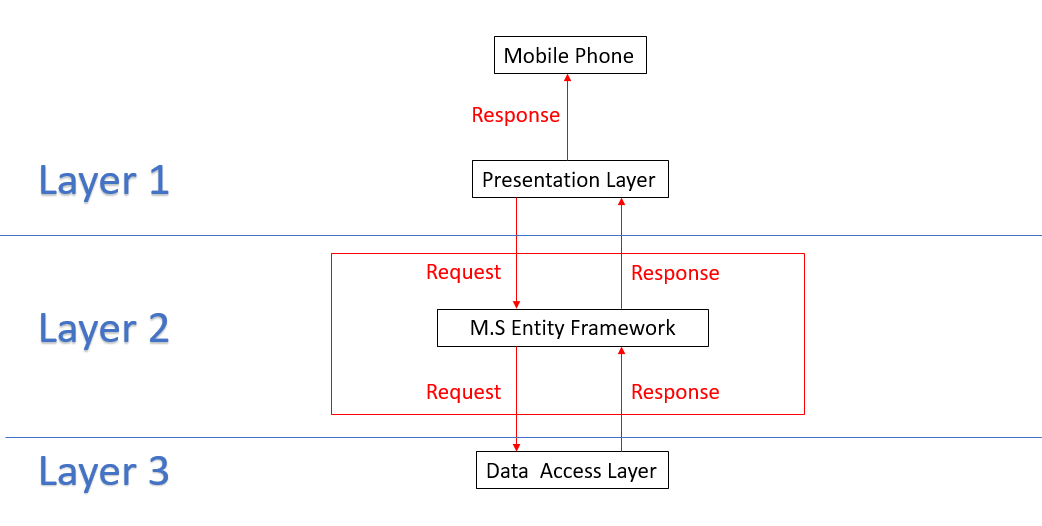


Figure: Design Pattern

# Pair Programming

In this case, we decide to try pair programming during the development stage. We choose two programmers to work together, one is skilful and another is less.

However, at the beginning stage of programming, they argue for a long time instead of programming, but this situation disappeared after they start to program. One of them observed and gave ideas when another one was programming. Although, sometimes they were not in the same track, but after a short discussion, everything went well again.

The position of programmer and observer always change in order to keep the efficiency and one of them can take a break. But generally speaking, it is really exhausting after a day of development.

To sum up, pair programming seems inefficient and it really difficult to judge it is true or not, but the software does run well and we did not see any bug for now and also, both of the programmers did learn new knowledge in this case.

# Story Cards

We use different agile objects which are shown through Story cards. In every story

cards we finish through our priority marks.

**Story-1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Install Android Studio in all members laptops** | | | | |
| **Estimation** | | 2.5 hours | **Priority** | High |
| **Story Description** | | | | |
| As a | Project group member | | | |
| We want | To develop good android project. | | | |
| So that | We can start to develop the Android project. | | | |
| **Acceptance Criteria** | | | | |

**Story 2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name – Setup the Git repository** | | | | |
| **Estimation** | | One day | **Priority** | High |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To setup all team member contribution for project. | | | |
| So that | We can access easily same file. | | | |
| **Acceptance Criteria** | | | | |

**Story -3:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Divide tasks to all team member** | | | | |
| **Estimation** | |  | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To provide equal tasks to each member | | | |
| So that | They can equally contribute to implement good project. | | | |
| **Acceptance Criteria** | | | | |

**Story 4:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name- Make a paper Prototype** | | | | |
| **Estimation** | | 2 days | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To create an overall initial design. | | | |
| So that | All member will know about overall design of Application. | | | |
| **Acceptance Criteria** | | | | |

**Story 5:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name-** Creating Interface | | | | |
| **Estimation** | |  | **Priority** | High |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To create an interface consisting the list of destination | | | |
| So that | People can view and start the application | | | |
| **Acceptance Criteria** | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Implement Map into the Project** | | | | |
| **Estimation** | |  | **Priority** | High |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To setup the real map in Project | | | |
| So that | People can easily find location. | | | |
| **Acceptance Criteria** | | | | |

**Story-6**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Start to write documentation Abstract** | | | | |
| **Estimation** | |  | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To specify the overall information of the project | | | |
| So that | Get all information related to the project | | | |
| **Acceptance Criteria** | | | | |

**Story 7**

**Story card 8:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Write documentation parts project description** | | | | |
| **Estimation** | |  | **Priority** | High |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To Describe the information of our project | | | |
| So that | Know about the what the project is. | | | |
| **Acceptance Criteria** | | | | |

**Story 9:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: list of places with details** | | | | |
| **Estimation** | | 1 day | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To list all patrolling location with their details. | | | |
| So that | User can know their patrolling location and details associated with it. | | | |
| **Acceptance Criteria** | | | | |

**Story 10:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Integrate those places and map** | | | | |
| **Estimation** | | 1 day | **Priority** | High |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To integrate the various places with the map | | | |
| So that | The next location of the list will come at first | | | |
| **Acceptance Criteria** | | | | |

**Story 11.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Integrate GPS and Map** | | | | |
| **Estimation** | | 5 h | **Priority** | High |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To Integrate Map with the GPS location | | | |
| So that | User easily finds the location with the GPS | | | |
| **Acceptance Criteria** | | | | |

**Story 12**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Write Project purpose, Objectives and Goal in Documentation** | | | | |
| **Estimation** | | 2 hours | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To describe the purpose and objectives of this application | | | |
| So that | User finds the actual purpose and goals of the project in this part. | | | |
| **Acceptance Criteria** | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Connect list with map** | | | | |
| **Estimation** | | 5h | **Priority** | High |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To connect list of the location with real google map | | | |
| So that | User finds the location how and where to go. | | | |
| **Acceptance Criteria** | | | | |

Story 13

Story 14

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Provide details to the map location in lists** | | | | |
| **Estimation** | | 2 hours | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To provide the description of the map location in the list view | | | |
| So that | It became easy to know overall information of the saved location. | | | |
| **Acceptance Criteria** | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Find the TPS Algorithm** | | | | |
| **Estimation** | | 20h | **Priority** | High |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To find the best algorithm for TPS | | | |
| So that | User can solve the problem about route | | | |
| **Acceptance Criteria** | | | | |

Story 15

**Story 16**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: get time to travel between location by using google map** | | | | |
| **Estimation** | | 5h | **Priority** | High |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To provide the minimum time between the different location | | | |
| So that | User can find the best route with less time | | | |
| **Acceptance Criteria** | | | | |

**Story 17.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Connect TSP algorithm and time** | | | | |
| **Estimation** | | 6 h | **Priority** | High |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To give the connection with TSP algorithm and time | | | |
| So that | User easy to find the best route with minimum time interval. | | | |
| **Acceptance Criteria** | | | | |

**Story 18**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Get the nearest location** | | | | |
| **Estimation** | | 2 h | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To setup the way that user can add the location | | | |
| So that | User can find the nearest location that they saved. | | | |
| **Acceptance Criteria** | | | | |

**Story 19**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Provide routes to the location** | | | | |
| **Estimation** | | 5 h | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To provide route of the different user saved location | | | |
| So that | User can track the way of different location | | | |
| **Acceptance Criteria** | | | | |

**Story 20**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Connect all modules together (every activity sequentially)** | | | | |
| **Estimation** | | 3 hours | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To connect every events and pages to each other | | | |
| So that | The application will give the good output | | | |
| **Acceptance Criteria** | | | | |

**Story 21.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Provide current GPS location** | | | | |
| **Estimation** | | 1h | **Priority** | High |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To mark the current location where user have. | | | |
| So that | User able to know the current location. | | | |
| **Acceptance Criteria** | | | | |

**Story 22**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Feed current GPS location at nth location toTSP** | | | | |
| **Estimation** | | 3 h | **Priority** | High |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To setup the number location to TSP | | | |
| So that | User can add multiple number of location which they want. | | | |
| **Acceptance Criteria** | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Check if all components are working properly** | | | | |
| **Estimation** | | 5 h | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To check all modules and components are working properly or not | | | |
| So that | We are able to find all bugs | | | |
| **Acceptance Criteria** | | | | |

**Story 23**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: UML Design (Use case Diagram)** | | | | |
| **Estimation** | | 1 h | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To create Use case diagram to describe the set of actions of system. | | | |
| So that | It helps to save the lots of time and describes all thing that might any wrong. | | | |
| **Acceptance Criteria** | | | | |

**Story 24:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: UML Design (Use case Diagram)** | | | | |
| **Estimation** | | 1 h | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To create Use case diagram to describe the set of actions of system. | | | |
| So that | It helps to save the lots of time and describes all thing that might any wrong. | | | |
| **Acceptance Criteria** | | | | |

**Story 25:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: UML Design (Sequence Diagram) Documentation** | | | | |
| **Estimation** | | 2 h | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To create Sequence Diagram to describe how objects operate with one another. | | | |
| So that | It helps to interact logic between the objects in the system in the time order that the interactions take place | | | |
| **Acceptance Criteria** | | | | |

Story 26

Story 27

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Add UI Design in Documentation** | | | | |
| **Estimation** | | 1 h | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To Add all UI Design of our project | | | |
| So that | We can view the overall design of the Application | | | |
| **Acceptance Criteria** | | | | |

Story 27

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: UML Design (Activity Diagram) Documentation** | | | | |
| **Estimation** | | 1h | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To create that to represent the flow of activity from one to another activity. | | | |
| So that | We are able to know that the actual state of the system. | | | |
| **Acceptance Criteria** | | | | |

Story 28

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story Name: Write Requirements of the Project in Documentation** | | | | |
| **Estimation** | | 2 h | **Priority** | Medium |
| **Story Description** | | | | |
| As a | Developer | | | |
| I want | To know the main Functional and non- Functional requirements of the application | | | |
| So that | We know about all possible requirements of Application. | | | |
| **Acceptance Criteria** | | | | |

# Agile Process

Agile Software Development is generally new term and it is altogether different from conventional advancement systems. Nimble strategy put more concentrate on techniques for change administration and less on in advance designs based approach. The primary thought behind deft advancement is to feature and quicken the activities on developing conditions, accomplishing the due dates and necessities (Rao et al. 2011).



Fig: Agile Function

It is a set of tools and systems which depend on iterative and incremental approach. In light-footed, the prerequisites and arrangements are developed by careful joint effort between self-sorting out and self-working groups (Rao et al. 2011). Customer is additionally an exceptionally significant piece of the improvement group and normal correspondence is kept up and constant criticism is consumed.

The organizations and experts in IT divisions incline toward nimble advancement approach because of the way that it gives enhanced correspondence between the colleagues, fast arrival of the item and adaptability in the plans. The improvement group fluctuates between 2-20 and extensive organizations like Microsoft and other enormous organizations change their spry group as indicated by the undertaking prerequisites (Begel and Nagappan, 2017)

Agile groups are gathering of multi-talented experts and item clients having noteworthy information of the space. Various short and quick improvement emphases are directed to suit the progressions depicted by the customer. It elevates small scale item guide to set up a more exact task grindstone.

# Security

Our application asks for the username and password in order to login. If the username and password is not matched then the user is denied the further access not matched then the user is denied the further access.

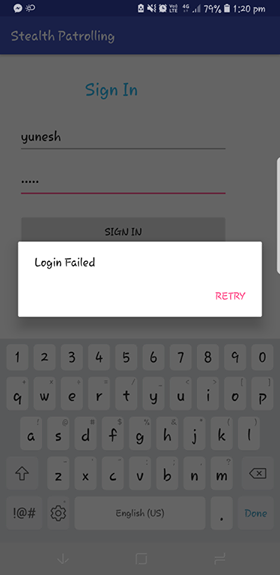


Fig: Unsuccessful Login

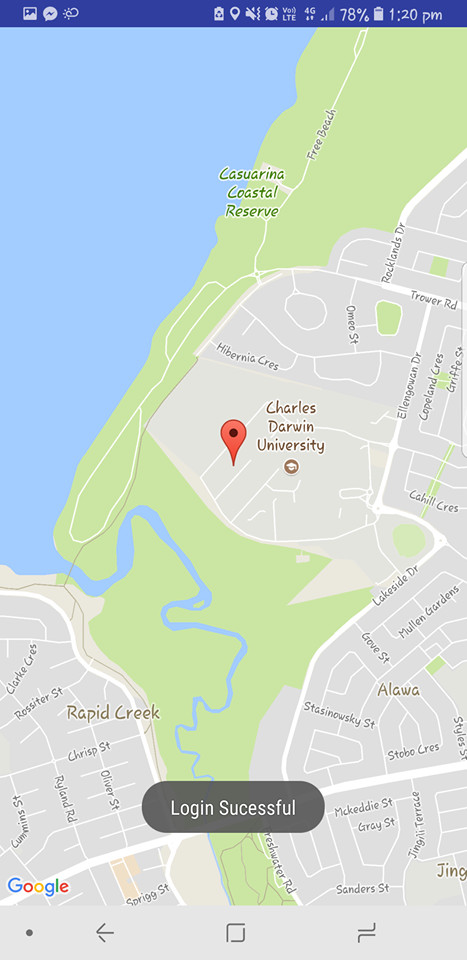


Fig: Successful Login

## Handling User Data

In general, the best approach for user data security is to minimize the use of APIs that access sensitive or personal user data. The user data is stored in our app only until the task is performed. After the task is completed then the user data is automatically removed from the database. This reduces the chances of inadvertently exposing the data, and it also reduces the chance of attackers attempting to exploit our application.

## Handling User Credential

Asking the user to use their authentication credential continuously will prone to the phishing attack. Instead of asking their login credential frequently, in our app we ask the user to login once and store those credential in our cloud based database and use the token to give them access. For this we have used AccountManager class which provokes the database automatically and verify the user. We have used CREATOR before passing credential so that our app does not pass the credential to any external source. Since the credential is used only by our app we have used checkSignature() method to verify that the application accesses the AccountManager.

# Code

1. **MainActivity.java**

**package** aayush.randompatrolling;  
**import** android.content.Intent;  
**import** android.os.Handler;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**public class** MainActivity **extends** AppCompatActivity {  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 **final** Handler handler = **new** Handler();  
 handler.postDelayed(**new** Runnable(){  
 @Override  
 **public void** run() {  
 Intent i = **new** Intent(MainActivity.**this**, Login.**class**);  
 startActivity(i);  
 }  
 }, 3000);  
 }  
}

1. **Login.java**

**package** aayush.randompatrolling;  
**import** android.content.Context;  
**import** android.net.ConnectivityManager;  
**import** android.net.NetworkInfo;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.content.Intent;  
**import** android.support.v7.app.AlertDialog;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.Toast;  
**import** com.android.volley.RequestQueue;  
**import** com.android.volley.Response;  
**import** com.android.volley.toolbox.Volley;  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
**public class** Login **extends** AppCompatActivity {  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_login***);  
 **final** EditText getUsername = (EditText) findViewById(R.id.***user\_username***);  
 **final** EditText getPassword = (EditText) findViewById(R.id.***user\_password***);  
 **final** Button sign\_in = (Button) findViewById(R.id.***sign\_in\_button***);  
 **final** Button login\_Register = (Button) findViewById(R.id.***register\_button***);  
 login\_Register.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 Intent loginIntent = **new** Intent(Login.**this**, register.**class**);  
 Login.**this**.startActivity(loginIntent);  
 }  
 });  
 sign\_in.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (hasNetworkConnection()) {  
 **final** String username = getUsername.getText().toString();  
 **final** String password = getPassword.getText().toString();  
 Response.Listener<String> responseListener = **new** Response.Listener<String>() {  
 @Override  
 **public void** onResponse(String response) {  
 **try** {  
 JSONObject jsonResponse = **new** JSONObject(response);  
 **boolean** sucess = jsonResponse.getBoolean(**"sucess"**);  
 **if** (sucess) {  
 String name = jsonResponse.getString(**"name"**);  
 String organization = jsonResponse.getString(**"organization"**);  
 String user\_type = jsonResponse.getString(**"user\_type"**);  
 Intent intent = **new** Intent(Login.**this**, MapsActivity.**class**);  
 Login.**this**.startActivity(intent);  
 Toast.*makeText*(getApplicationContext(), **"Login Sucessful"**, Toast.***LENGTH\_SHORT***).show();  
 } **else** {  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(Login.**this**);  
 builder.setMessage(**"Login Failed"**).setNegativeButton(**"Retry"**, **null**).create().show();  
 }  
 } **catch** (JSONException e) {  
 e.printStackTrace();  
 }  
 }  
 };  
  
 LoginRequest loginRequest = **new** LoginRequest(username, password, responseListener);  
 RequestQueue req\_queue = Volley.*newRequestQueue*(Login.**this**);  
 req\_queue.add(loginRequest);  
 } **else** {  
 Toast.*makeText*(getApplicationContext(), **"No internet Connection"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 });  
 }  
 **public boolean** hasNetworkConnection() {  
 **boolean** connected = **false**;  
 **try** {  
 ConnectivityManager connectivityManager = (ConnectivityManager) getSystemService(Context.***CONNECTIVITY\_SERVICE***);  
 NetworkInfo netActive = connectivityManager.getActiveNetworkInfo();  
 connected = netActive != **null** && netActive.isAvailable() && netActive.isConnected();  
 **return** connected;  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 **return** connected;  
 }  
}

1. **LoginRequest.java**

**package** aayush.randompatrolling;  
**import** com.android.volley.Request;  
**import** com.android.volley.Response;  
**import** com.android.volley.toolbox.StringRequest;  
**import** java.util.HashMap;  
**import** java.util.Map;  
**public class** LoginRequest **extends** StringRequest {  
 **private static final** String ***LOGIN\_REQUEST\_URL*** = **"https://asapkota.spinetail.cdu.edu.au/stealth\_patrolling/login.php"**;  
 **private** Map<String, String> **params**;  
 **public** LoginRequest(String username, String password, Response.Listener<String> listener){  
 **super**(Request.Method.***POST***, ***LOGIN\_REQUEST\_URL***, listener, **null**);  
 **params** = **new** HashMap<>();  
 **params**.put(**"username"**, username);  
 **params**.put(**"password"**, password);  
 }  
 **public** Map<String, String> getParams() {  
 **return params**;  
 }  
}

1. **MapsActivity.java**

**package** aayush.randompatrolling;  
**import** android.Manifest;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.content.pm.PackageManager;  
**import** android.location.Address;  
**import** android.location.Criteria;  
**import** android.location.Geocoder;  
**import** android.location.Location;  
**import** android.location.LocationListener;  
**import** android.location.LocationManager;  
**import** android.os.Build;  
**import** android.support.v4.app.ActivityCompat;  
**import** android.support.v4.app.FragmentActivity;  
**import** android.os.Bundle;  
**import** android.support.v4.content.ContextCompat;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** com.google.android.gms.maps.CameraUpdate;  
**import** com.google.android.gms.maps.CameraUpdateFactory;  
**import** com.google.android.gms.maps.GoogleMap;  
**import** com.google.android.gms.maps.OnMapReadyCallback;  
**import** com.google.android.gms.maps.SupportMapFragment;  
**import** com.google.android.gms.maps.model.LatLng;  
**import** com.google.android.gms.maps.model.Marker;  
**import** com.google.android.gms.maps.model.MarkerOptions;  
**import** java.io.IOException;  
**import** java.util.ArrayList;  
**import** java.util.List;  
**import** java.util.Locale;  
**import static** android.location.LocationManager.***NETWORK\_PROVIDER***;  
**public class** MapsActivity **extends** FragmentActivity **implements** OnMapReadyCallback {  
 **public static** LocationListener *locationListener*;  
 **private** GoogleMap **mMap**;  
 GoogleMap.OnInfoWindowClickListener **onInfoWindowClickListener**;  
 **private** LocationManager **locationManager**;  
 placeManager **placeObj** = **new** placeManager();  
 ArrayList<SelectedLocation> **LocationList**;  
 LatLng **newLocationAdded**;  
 **private double latitude**;  
 **private double longitude**;  
 **private** String **address**;  
 **private** String **minStay**;  
 **private** String **maxStay**;  
 **private** String **priority**;  
 **private** String **checkBackOn**;  
 **private** String **result**;  
 Location **lastLocation**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_maps***);  
 *// Obtain the SupportMapFragment and get notified when the map is ready to be used.* SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()  
 .findFragmentById(R.id.***map***);  
 mapFragment.getMapAsync(**this**);  
 }  
 @Override  
 **public void** onMapReady(GoogleMap googleMap) {  
 **mMap** = googleMap;  
 *locationListener* = **new** LocationListener() {  
 @Override  
 **public void** onLocationChanged(Location location) {  
 centerMapLocation(location, **"Your location"**);  
 }  
 @Override  
 **public void** onStatusChanged(String s, **int** i, Bundle bundle) {  
 }  
 @Override  
 **public void** onProviderEnabled(String s) {  
 }  
 @Override  
 **public void** onProviderDisabled(String s) {  
 }  
 };  
 **locationManager** = (LocationManager) **this**.getSystemService(Context.***LOCATION\_SERVICE***);  
 **if** (**locationManager** != **null**) {  
  
 **if** (ContextCompat.*checkSelfPermission*(**this**, Manifest.permission.***ACCESS\_FINE\_LOCATION***) == PackageManager.***PERMISSION\_GRANTED***) {  
 **mMap**.setMyLocationEnabled(**true**);  
 **locationManager**.requestLocationUpdates(LocationManager.***NETWORK\_PROVIDER***, 300, 500, *locationListener*);  
 **lastLocation** = **locationManager**.getLastKnownLocation(LocationManager.***NETWORK\_PROVIDER***);  
 centerMapLocation(**lastLocation**, **"Your location"**);  
  
 } **else** {  
 ActivityCompat.*requestPermissions*(**this**, **new** String[]{Manifest.permission.***ACCESS\_FINE\_LOCATION***}, 1);  
 }  
 }  
 Button places = (Button) findViewById(R.id.***places***);  
 Button alarms = (Button) findViewById(R.id.***alarms***);  
 places.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(getApplicationContext(), places.**class**);  
 startActivity(i);  
 }  
 });  
 alarms.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(getApplicationContext(), alarms.**class**);  
 startActivity(i);  
 }  
 });  
 **LocationList** = **placeObj**.getdestinationList();  
 **for** (SelectedLocation s : **LocationList**) {  
 **address** = s.getName();  
 **latitude** = Double.*parseDouble*(s.getLatitude());  
 **longitude** = Double.*parseDouble*(s.getLongitude());  
 **newLocationAdded** = **new** LatLng(**latitude**, **longitude**);  
 **minStay** = s.getMinTimeToStay();  
 **maxStay** = s.getMaxTimeTOStay();  
 **priority** = s.getPriority();  
 **checkBackOn** = s.getCheckBackOn();  
 **mMap**.addMarker(**new** MarkerOptions().position(**newLocationAdded**).title(**address**));  
 }  
 **onInfoWindowClickListener** = **new** GoogleMap.OnInfoWindowClickListener() {  
 @Override  
 **public void** onInfoWindowClick(Marker marker) {  
 windowClick(marker);  
 }  
 };  
 **mMap**.setOnInfoWindowClickListener(**onInfoWindowClickListener**);  
 }  
 **public void** centerMapLocation(Location location, String title) {  
 LatLng userLocation = **new** LatLng(location.getLatitude(), location.getLongitude());  
  
 **mMap**.moveCamera(CameraUpdateFactory.*newLatLngZoom*(userLocation, 15));  
 }  
 **public** Location getUserLocation(){  
 **return lastLocation**;  
 }  
 **public void** windowClick(Marker marker) {  
 String markerAddress = marker.getTitle();  
 Log.*d*(**"markerTitle"**, markerAddress);  
 **for** (SelectedLocation s : **LocationList**) {  
 **if** (markerAddress.equals(s.getName())) {  
 Intent i = **new** Intent(MapsActivity.**this**, Display\_location\_info.**class**);  
 i.putExtra(**"address"**, s.getName());  
 i.putExtra(**"Latitude"**, s.getLatitude());  
 i.putExtra(**"Longitude"**, s.getLongitude());  
 i.putExtra(**"minTime"**, s.getMinTimeToStay());  
 i.putExtra(**"MaxTime"**, s.getMaxTimeTOStay());  
 i.putExtra(**"priority"**, s.getPriority());  
 i.putExtra(**"checkBackOn"**, s.getCheckBackOn());  
 startActivity(i);  
 }  
 }  
 }  
}

1. **addAlarm.java**

**package** aayush.randompatrolling;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.net.ConnectivityManager;  
**import** android.net.NetworkInfo;  
**import** android.os.Bundle;  
**import** android.support.v7.app.AlertDialog;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.Toast;  
**import** com.android.volley.RequestQueue;  
**import** com.android.volley.Response;  
**import** com.android.volley.toolbox.Volley;  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
**public class** addAlarm **extends** AppCompatActivity {  
 Login **login** = **new** Login();  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_add\_alarm***);  
 **final** EditText titlebox = (EditText) findViewById(R.id.***titlebox***);  
 **final** EditText messagebox = (EditText) findViewById(R.id.***messageBox***);  
 **final** Button addAlarm = (Button) findViewById(R.id.***addAlarmButton***);  
 **final** Button back = (Button) findViewById(R.id.***alarmBack***);  
  
 addAlarm.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if** (hasNetworkConnection()) {  
 **final** String title = titlebox.getText().toString();  
 **final** String message = messagebox.getText().toString();  
 **final** Response.Listener<String> responseListener = **new** Response.Listener<String>() {  
 @Override  
 **public void** onResponse(String response) {  
 **try** {  
 JSONObject jsonResponse = **new** JSONObject(response);  
 **boolean** sucess = jsonResponse.getBoolean(**"sucess"**);  
 **if** (sucess) {  
 Intent intent = **new** Intent(aayush.randompatrolling.addAlarm.**this**, alarms.**class**);  
 aayush.randompatrolling.addAlarm.**this**.startActivity(intent);  
 Toast.*makeText*(getApplicationContext(), **"successfully Sent!"**, Toast.***LENGTH\_SHORT***).show();  
 } **else** {  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(aayush.randompatrolling.addAlarm.**this**);  
 builder.setMessage(**"Sending Failed"**).setNegativeButton(**"Retry"**, **null**).create().show();  
 }  
  
 } **catch** (JSONException ex) {  
 ex.printStackTrace();  
 }  
 }  
 };  
 alarmRequest AlarmRequest = **new** alarmRequest(title, message,responseListener);  
 RequestQueue req\_queue = Volley.*newRequestQueue*(aayush.randompatrolling.addAlarm.**this**);  
 req\_queue.add(AlarmRequest);  
 }  
 **else** {  
 Toast.*makeText*(aayush.randompatrolling.addAlarm.**this**, **"No internet Connection"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 });  
 }  
 **public boolean** hasNetworkConnection() {  
 **boolean** connected = **false**;  
 **try** {  
 ConnectivityManager connectivityManager = (ConnectivityManager) getSystemService(Context.***CONNECTIVITY\_SERVICE***);  
 NetworkInfo netActive = connectivityManager.getActiveNetworkInfo();  
 connected = netActive != **null** && netActive.isAvailable() && netActive.isConnected();  
 **return** connected;  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 **return** connected;  
 }  
 }

1. **addPlace.java**

**package** aayush.randompatrolling;  
**import** android.content.Intent;  
**import** android.database.sqlite.SQLiteException;  
**import** android.support.v7.app.AlertDialog;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.TextView;  
**import** java.util.ArrayList;  
**import** java.util.Arrays;  
**import** java.util.Collections;  
**public class** addPlace **extends** AppCompatActivity {  
 **private** ArrayList<SelectedLocation> **selectedLocations** = **new** ArrayList<>();  
 **private** SelectedLocation **selectedLocationObj** = **new** SelectedLocation();  
 **private double latitude**;  
 **private double longitude**;  
 **private** String **placeCheckBackTime**;  
 **private** String **minTimeStay**;  
 **private** String **maxTimeStay**;  
 **private** String **priorityIndex**;  
 **private** String **addressName**;  
 **private double distance** = 0;  
 **private double fitness** = 0;  
 **public** addPlace() {  
 **for** (**int** i = 0; i < placeManager.*numberOfLocations*(); i++) {  
 **selectedLocations**.add(**null**);  
 }  
 }  
 **public** addPlace(ArrayList<SelectedLocation> selectedLocationA) {  
 **this**.**selectedLocations** = selectedLocationA;  
 }  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_add\_place***);  
 **final** EditText address = (EditText) findViewById(R.id.***address***);  
 **final** EditText minTime = (EditText) findViewById(R.id.***minTimeStay***);  
 **final** EditText maxTime = (EditText) findViewById(R.id.***maxTimeStay***);  
 **final** EditText priority = (EditText) findViewById(R.id.***placePriority***);  
 **final** EditText checkBackTime = (EditText) findViewById(R.id.***checkBackDuration***);  
 **final** Button addPlace = (Button) findViewById(R.id.***addPlaceButton***);  
 Intent getPlaces = getIntent();  
 **if** (getPlaces.getStringExtra(**"addressName"**) != **null**) {  
 String addressname = getPlaces.getStringExtra(**"addressName"**);  
 Log.*d*(**"addressName"**, addressname);  
 address.setText(addressname);  
 }  
 **if** (getPlaces.getDoubleExtra(**"latitude"**, 0) != 0) {  
 **this**.**latitude** = getPlaces.getDoubleExtra(**"latitude"**, 0);  
 Log.*d*(**"latitude"**, String.*valueOf*(**latitude**));  
 }  
 **if** (getPlaces.getDoubleExtra(**"longitude"**, 0) != 0) {  
 **this**.**longitude** = getPlaces.getDoubleExtra(**"longitude"**, 0);  
 Log.*d*(**"longitude"**, String.*valueOf*(**longitude**));  
 }  
 addPlace.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **addressName** = address.getText().toString();  
 **minTimeStay** = minTime.getText().toString();  
 **maxTimeStay** = maxTime.getText().toString();  
 **priorityIndex** = priority.getText().toString();  
 **placeCheckBackTime** = checkBackTime.getText().toString();  
 **selectedLocationObj**.addPlaceInformation(**addressName**, String.*valueOf*(**longitude**),  
 String.*valueOf*(**latitude**), **minTimeStay**, **maxTimeStay**, **priorityIndex**, **placeCheckBackTime**);  
 Log.*d*(**"selectLocationObj"**, **selectedLocationObj**.getName());  
 placeManager.*addCity*(**selectedLocationObj**);  
 Intent i = **new** Intent(addPlace.**this**, MapsActivity.**class**);  
 startActivity(i);  
 }  
 });  
 address.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(getApplicationContext(), mapSelect.**class**);  
 startActivity(i);  
 }  
 });  
 }  
**public void** generateIndividual() {  
 **for** (**int** i = 0; i < placeManager.*numberOfLocations*(); i++) {  
 setLocation(i, placeManager.*getCity*(i));  
 }  
 *//randomly reorder the tour* Collections.*shuffle*(**selectedLocations**);  
 }  
 **public** SelectedLocation getLocation(**int** tourPosition) {  
 **return** (SelectedLocation) **selectedLocations**.get(tourPosition);  
 }  
 **public void** setLocation(**int** tourPosition, SelectedLocation location) {  
 **selectedLocations**.set(tourPosition, location);  
 **fitness** = 0;  
 **distance** = 0;  
 }  
 **public boolean** hasLocation(SelectedLocation location) {  
 **return selectedLocations**.contains(location);  
 }  
 *//get the tour fitness* **public double** getFitness() {  
 **if** (**fitness** == 0) {  
 **fitness** = 1 / getTourDistance();  
 }  
 **return fitness**;  
 }  
 **public int** tourSize(){  
 **return selectedLocations**.size();  
 }  
 **public double** getTourDistance() {  
 **if** (**distance** == 0) {  
 **double** tourDistance = 0;  
 *//loop through all cities* **for** (**int** i = 0; i < tourSize(); i++) {  
 *//origin city* SelectedLocation origin = getLocation(i);  
 *//destination city* SelectedLocation destination;  
 *//check we are not on last city  
 //tour final destination set to starting city* **if** ((i + 1) < tourSize()) {  
 destination = getLocation(i+1);  
 } **else** {  
 destination = getLocation(0);  
 }  
 *//get the distance between the two cities* tourDistance += origin.distanceTo(destination);  
 }  
 **distance** = tourDistance;  
 }  
 **return distance**;  
 }  
 @Override  
 **public** String toString() {  
 String geneString = **"|"**;  
 **for** (**int** i = 0; i < **selectedLocations**.size(); i++) {  
 geneString += getLocation(i).getName() + **" --> "**;  
 }  
 **return** geneString;  
 }  
}

1. **alarmRequest.java**

**package** aayush.randompatrolling;  
**import** com.android.volley.Response;  
**import** com.android.volley.toolbox.StringRequest;  
**import** java.util.HashMap;  
**import** java.util.Map;  
**public class** alarmRequest **extends** StringRequest{  
 **private static final** String ***REGISTER\_REQUEST\_URL*** = **"https://asapkota.spinetail.cdu.edu.au/stealth\_patrolling/alarm.php"**;  
 **private** Map<String, String> **params**;  
  
 **public** alarmRequest(String title, String message, Response.Listener<String>  
 listener){  
 **super**(Method.***POST***, ***REGISTER\_REQUEST\_URL***, listener, **null**);  
 **params** = **new** HashMap<>();  
 **params**.put(**"title"**, title);  
 **params**.put(**"message"**,message);  
 }  
 **public** Map<String, String> getParams() {  
 **return params**;  
 }  
}

1. **alarms.java**

**package** aayush.randompatrolling;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.net.ConnectivityManager;  
**import** android.net.NetworkInfo;  
**import** android.os.Bundle;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.view.View;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.Button;  
**import** android.widget.ListView;  
**import** android.widget.Toast;  
**import** com.android.volley.RequestQueue;  
**import** com.android.volley.Response;  
**import** com.android.volley.toolbox.Volley;  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
**import** java.lang.reflect.Array;  
**import** java.util.ArrayList;  
**public class** alarms **extends** AppCompatActivity {  
 Array **title**[];  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_alarms***);  
 **final** ListView display = (ListView) findViewById(R.id.***listView***);  
 Button back = (Button) findViewById(R.id.***back***);  
 Button addAlarm = (Button) findViewById(R.id.***addAlarm***);  
 **if** (hasNetworkConnection()) {  
 Response.Listener<String> responseListener = **new** Response.Listener<String>() {  
 @Override  
 **public void** onResponse(String response) {  
 **try** {  
 JSONObject jsonResponse = **new** JSONObject(response);  
 Boolean sucess = jsonResponse.getBoolean(**"sucess"**);  
 String title;  
 String message;  
 String id;  
 ArrayList<String> alarmsList = **new** ArrayList<>();  
 ArrayAdapter arrayAdapter = **new** ArrayAdapter(alarms.**this**,android.R.layout.***simple\_list\_item\_1***,alarmsList);  
 **if** (sucess) {  
 **for** (**int** i = 0; i < jsonResponse.length() - 1; i++) {  
 String j = String.*valueOf*(i);  
 JSONObject object = (JSONObject) jsonResponse.get(j);  
 title = (String) object.get(**"title"**);  
 message = (String) object.get(**"message"**);  
 id = (String) object.get(**"id"**);  
 alarmsList.add(**"\n"** +title +**"\n\n"**+message+**"\n"**);  
 }  
 display.setAdapter(arrayAdapter);  
 }  
 } **catch** (JSONException e) {  
 e.printStackTrace();  
 }  
 }  
 };  
 alarmsRequest alarmsRequest = **new** alarmsRequest(responseListener);  
 RequestQueue req\_queue = Volley.*newRequestQueue*(alarms.**this**);  
 req\_queue.add(alarmsRequest);  
 } **else** {  
 Toast.*makeText*(getApplicationContext(), **"No internet Connection"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 back.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(getApplicationContext(), MapsActivity.**class**);  
 startActivity(i);  
 }  
 });  
 addAlarm.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(getApplicationContext(), aayush.randompatrolling.addAlarm.**class**);  
 startActivity(i);  
 }  
 });  
 }  
 **public boolean** hasNetworkConnection() {  
 **boolean** connected = **false**;  
 **try** {  
 ConnectivityManager connectivityManager = (ConnectivityManager) getSystemService(Context.***CONNECTIVITY\_SERVICE***);  
 NetworkInfo netActive = connectivityManager.getActiveNetworkInfo();  
 connected = netActive != **null** && netActive.isAvailable() && netActive.isConnected();  
 **return** connected;  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 **return** connected;  
 }  
}

1. **Display\_Location\_Info.java**

**package** aayush.randompatrolling;  
**import** android.content.Intent;  
**import** android.support.v4.view.ScrollingView;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.ScrollView;  
**import** android.widget.TextView;  
**public class** Display\_location\_info **extends** AppCompatActivity {  
 **private** String **address**;  
 **private** String **minStay**;  
 **private** String **maxStay**;  
 **private** String **priority**;  
 **private** String **checkBackOn**;  
 **private** String **result**;  
 **private** String **latitude**;  
 **private** String **longitude**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_display\_location\_info***);  
 Button back = (Button) findViewById(R.id.***back***);  
 Intent i = getIntent();  
 **address** = i.getStringExtra(**"address"**);  
 **latitude** = i.getStringExtra(**"Latitude"**);  
 **longitude** = i.getStringExtra(**"Longitude"**);  
 **minStay** = i.getStringExtra(**"minTime"**);  
 **maxStay** = i.getStringExtra(**"MaxTime"**);  
 **priority** = i.getStringExtra(**"priority"**);  
 **checkBackOn** = i.getStringExtra(**"checkBackOn"**);  
 **result** = **"Place Info\n\n\n Address:"**+**address**+  
 **"\n\n Latitude: "** + **latitude**+ **"\n\n Longitude: "** + **longitude**+ **"\n\n Min Time to Stay: "** + **minStay**+  
 **" mins \n\n Max time to stay: "** + **maxStay**+**" mins\n\n Priority: "** +**priority**+ **"\n\n Visit Again in: "** + **checkBackOn**;  
  
 TextView view = (TextView) findViewById(R.id.***viewData***);  
 view.setText(**result**);  
 back.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(getApplicationContext(), MapsActivity.**class**);  
 startActivity(i);  
 }  
 });  
 }  
}

1. **mapSelect.java**

**package** aayush.randompatrolling;  
**import** android.Manifest;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.content.pm.PackageManager;  
**import** android.location.Address;  
**import** android.location.Geocoder;  
**import** android.location.Location;  
**import** android.location.LocationListener;  
**import** android.location.LocationManager;  
**import** android.support.v4.app.ActivityCompat;  
**import** android.support.v4.app.FragmentActivity;  
**import** android.os.Bundle;  
**import** android.support.v4.content.ContextCompat;  
**import** android.util.Log;  
**import** com.google.android.gms.maps.CameraUpdateFactory;  
**import** com.google.android.gms.maps.GoogleMap;  
**import** com.google.android.gms.maps.OnMapReadyCallback;  
**import** com.google.android.gms.maps.SupportMapFragment;  
**import** com.google.android.gms.maps.model.LatLng;  
**import** com.google.android.gms.maps.model.MarkerOptions;  
**import** java.io.IOException;  
**import** java.util.List;  
**import** java.util.Locale;  
**public class** mapSelect **extends** FragmentActivity **implements** OnMapReadyCallback, GoogleMap.OnMapLongClickListener {  
 **private** GoogleMap **mMap**;  
 **private** String **address** = **""**;  
 **private** LocationListener **locationListener**;  
 **private** LocationManager **locationManager**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_map\_select***);  
 *// Obtain the SupportMapFragment and get notified when the map is ready to be used.* SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()  
 .findFragmentById(R.id.***map***);  
 mapFragment.getMapAsync(**this**);  
 }  
 @Override  
 **public void** onMapReady(GoogleMap googleMap) {  
 **mMap** = googleMap;  
 **mMap**.setOnMapLongClickListener((GoogleMap.OnMapLongClickListener) **this**);  
 **locationListener** = **new** LocationListener() {  
 @Override  
 **public void** onLocationChanged(Location location) {  
 centerMapLocation(location, **"Your location"**);  
 }  
 @Override  
 **public void** onStatusChanged(String s, **int** i, Bundle bundle) {  
 }  
 @Override  
 **public void** onProviderEnabled(String s) {  
 }  
 @Override  
 **public void** onProviderDisabled(String s) {  
 }  
 };  
 **locationManager** = (LocationManager) **this**.getSystemService(Context.***LOCATION\_SERVICE***);  
 **if** (**locationManager** != **null**) {  
 **if** (ContextCompat.*checkSelfPermission*(**this**, Manifest.permission.***ACCESS\_FINE\_LOCATION***) == PackageManager.***PERMISSION\_GRANTED***) {  
 **mMap**.setMyLocationEnabled(**true**);  
 **locationManager**.requestLocationUpdates(LocationManager.***NETWORK\_PROVIDER***, 0, 0, **locationListener**);  
 Location lastLocation = **locationManager**.getLastKnownLocation(LocationManager.***NETWORK\_PROVIDER***);  
 centerMapLocation(lastLocation, **"Your location"**);  
 } **else** {  
 ActivityCompat.*requestPermissions*(**this**, **new** String[]{Manifest.permission.***ACCESS\_FINE\_LOCATION***}, 1);  
 }  
 }  
 }  
 @Override  
 **public void** onMapLongClick(LatLng latLng) {  
 Geocoder geocoder = **new** Geocoder(getApplicationContext(), Locale.*getDefault*());  
 **try** {  
 List<Address> currnetAddressLists = geocoder.getFromLocation(latLng.**latitude**, latLng.**longitude**, 1);  
 **if** (currnetAddressLists != **null** && currnetAddressLists.size() > 0) {  
 **if** (currnetAddressLists.get(0).getThoroughfare() != **null**) {  
 **if** (currnetAddressLists.get(0).getSubThoroughfare() != **null**) {  
 **address** += currnetAddressLists.get(0).getSubThoroughfare() + **" "**;  
 }  
 **address** += currnetAddressLists.get(0).getThoroughfare();  
 Log.*d*(**"address"**, **address**);  
 }  
 **if** (**address**.equals(**""**)) {  
 **address** = String.*valueOf*(latLng.**latitude**) + **" "** + String.*valueOf*(latLng.**longitude**);  
 }  
 Intent i = **new** Intent(getApplicationContext(), addPlace.**class**);  
 i.putExtra(**"addressName"**,**address**);  
 i.putExtra(**"latitude"**, latLng.**latitude**);  
 i.putExtra(**"longitude"**, latLng.**longitude**);  
 startActivity(i);  
 }  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 **private void** centerMapLocation(Location location, String title) {  
 LatLng userLocation = **new** LatLng(location.getLatitude(), location.getLongitude());  
 **mMap**.clear();  
 **mMap**.moveCamera(CameraUpdateFactory.*newLatLngZoom*(userLocation, 15));  
 }  
}

1. **placeManager.java**

**package** aayush.randompatrolling;  
**import** java.util.ArrayList;  
**public class** placeManager {  
 **private static** ArrayList<SelectedLocation> *destinationLocations* = **new** ArrayList<SelectedLocation>();  
 **public static void** addCity(SelectedLocation location){  
 *destinationLocations*.add(location);  
 }  
 **public static** SelectedLocation getCity(**int** index){  
 **return** (SelectedLocation) *destinationLocations*.get(index);  
 }  
 **public static int** numberOfLocations(){  
 **return** *destinationLocations*.size();  
 }  
 **public** ArrayList<SelectedLocation> getdestinationList(){  
 **return** *destinationLocations*;  
 }  
}

1. **places.java**

**package** aayush.randompatrolling;  
**import** android.content.Intent;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.Button;  
**import** android.widget.LinearLayout;  
**import** android.widget.ListView;  
**import** android.widget.TextView;  
**import** com.android.volley.Response;  
**import** java.util.ArrayList;  
**public class** places **extends** AppCompatActivity {  
 tspCalculator **tsp** = **new** tspCalculator();  
 @Override  
 **protected void** onCreate(**final** Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_places***);  
 Button addPlace = (Button) findViewById(R.id.***addPlaceButton***);  
 Button back = (Button) findViewById(R.id.***placesBack***);  
 back.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(getApplicationContext(), MapsActivity.**class**);  
 startActivity(i);  
 }  
 });  
  
 addPlace.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(getApplicationContext(), addPlace.**class**);  
 startActivity(i);  
 }  
 });  
 TextView resultView = (TextView) findViewById(R.id.***result***);  
 resultView.setText(**tsp**.getTspRoute());  
 }  
}

1. **register.java**

**package** aayush.randompatrolling;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.net.ConnectivityManager;  
**import** android.net.NetworkInfo;  
**import** android.support.v7.app.AlertDialog;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.Toast;  
**import** com.android.volley.RequestQueue;  
**import** com.android.volley.Response;  
**import** com.android.volley.toolbox.Volley;  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
**public class** register **extends** AppCompatActivity {  
 Login **login** = **new** Login();  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_register***);  
 **final** EditText getUsername = (EditText) findViewById(R.id.***register\_username***);  
 **final** EditText getPassword = (EditText) findViewById(R.id.***register\_password***);  
 **final** EditText getName = (EditText) findViewById(R.id.***register\_Name***);  
 **final** EditText getOrganization = (EditText) findViewById(R.id.***register\_organization***);  
 **final** EditText getUserType = (EditText) findViewById(R.id.***register\_user\_type***);  
 **final** Button Register = (Button) findViewById(R.id.***register\_button***);  
 Register.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **if**(hasNetworkConnection()){  
 **final** String name = getName.getText().toString();  
 **final** String organization = getOrganization.getText().toString();  
 **final** String username = getUsername.getText().toString();  
 **final** String password = getPassword.getText().toString();  
 **final** String user\_type = getUserType.getText().toString();  
 **final** Response.Listener<String> responseListener = **new** Response.Listener<String>() {  
 @Override  
 **public void** onResponse(String response) {  
 **try** {  
 JSONObject jsonResponse = **new** JSONObject(response);  
 **boolean** sucess = jsonResponse.getBoolean(**"sucess"**);  
 **if** (sucess) {  
 Intent intent = **new** Intent(register.**this**, Login.**class**);  
 register.**this**.startActivity(intent);  
 Toast.*makeText*(getApplicationContext(), **"successfully registered"**, Toast.***LENGTH\_SHORT***).show();  
 } **else** {  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(register.**this**);  
 builder.setMessage(**"Registration Failed"**).setNegativeButton(**"Retry"**, **null**).create().show();  
 }  
 } **catch** (JSONException ex) {  
 ex.printStackTrace();  
 }  
 }  
 };  
 RegisterRequest registerRequest = **new** RegisterRequest(name, organization, username, password, user\_type, responseListener);  
 RequestQueue req\_queue = Volley.*newRequestQueue*(register.**this**);  
 req\_queue.add(registerRequest);  
 }  
 **else**{  
 Toast.*makeText*(register.**this**, **"No internet Connection"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 });  
 }  
 **public boolean** hasNetworkConnection() {  
 **boolean** connected = **false**;  
 **try** {  
 ConnectivityManager connectivityManager = (ConnectivityManager) getSystemService(Context.***CONNECTIVITY\_SERVICE***);  
 NetworkInfo netActive = connectivityManager.getActiveNetworkInfo();  
 connected = netActive != **null** && netActive.isAvailable() && netActive.isConnected();  
 **return** connected;  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 **return** connected;  
 }  
}

1. **RegisterRequest.java**

**package** aayush.randompatrolling;  
**import** com.android.volley.Request;  
**import** com.android.volley.Response;  
**import** com.android.volley.toolbox.StringRequest;  
**import** java.util.HashMap;  
**import** java.util.Map;  
**public class** RegisterRequest **extends** StringRequest {  
 **private static final** String ***REGISTER\_REQUEST\_URL*** = **"https://asapkota.spinetail.cdu.edu.au/stealth\_patrolling/register.php"**;  
**private** Map<String, String> **params**;  
 **public** RegisterRequest(String name, String organization, String username, String password, String user\_type, Response.Listener<String>  
 listener){  
 **super**(Request.Method.***POST***, ***REGISTER\_REQUEST\_URL***, listener, **null**);  
 **params** = **new** HashMap<>();  
 **params**.put(**"name"**, name);  
 **params**.put(**"organization"**,organization);  
 **params**.put(**"username"**, username);  
 **params**.put(**"password"**, password);  
 **params**.put(**"user\_type"**, user\_type);  
 }  
 **public** Map<String, String> getParams() {  
 **return params**;  
 }  
}

1. **SelectedLocation.java :**

**package** aayush.randompatrolling;  
**import** android.Manifest;  
**import** android.content.Context;  
**import** android.content.pm.PackageManager;  
**import** android.location.Location;  
**import** android.location.LocationManager;  
**import** android.renderscript.RenderScript;  
**import** android.support.v4.app.ActivityCompat;  
**import** android.support.v4.content.ContextCompat;  
**import** com.google.android.gms.maps.OnMapReadyCallback;  
**import** com.google.android.gms.maps.model.LatLng;  
**import** java.util.ArrayList;  
**public class** SelectedLocation {  
 **private** ArrayList<String> **place** = **new** ArrayList<>();  
 **private** LocationManager **locationManager**;  
 **private** String **name**;  
 **private** String **longitude**;  
 **private** String **latitude**;  
 **private** String **minTimeToStay**;  
 **private** String **maxTimeTOStay**;  
 **private** String **priority**;  
 **private** String **checkBackOn**;  
 **double distance**;  
 **public void** addPlaceInformation(String name, String longitude, String latitude,  
 String minTimeToStay, String maxTimeTOStay, String priority, String checkBackOn) {  
 **this**.**name** = name;  
 **this**.**latitude** = latitude;  
 **this**.**longitude** = longitude;  
 **this**.**maxTimeTOStay** = maxTimeTOStay;  
 **this**.**minTimeToStay** = minTimeToStay;  
 **this**.**priority** = priority;  
 **this**.**checkBackOn** = checkBackOn;  
 **place**.add(name);  
 **place**.add(longitude);  
 **place**.add(latitude);  
 **place**.add(minTimeToStay);  
 **place**.add(maxTimeTOStay);  
 **place**.add(priority);  
 **place**.add(checkBackOn);  
 }  
 **public** ArrayList<String> getPlace() {  
 **return place**;  
 }  
 **public** String getName() {  
 **return this**.**name**;  
 }  
 **public** String getLongitude() {  
 **return this**.**longitude**;  
 }  
 **public** String getLatitude() {  
 **return this**.**latitude**;  
 }  
 **public** String getMinTimeToStay() {  
 **return this**.**minTimeToStay**;  
 }  
 **public** String getMaxTimeTOStay() {  
 **return this**.**maxTimeTOStay**;  
 }  
 **public** String getPriority() {  
 **return this**.**priority**;  
 }  
 **public** String getCheckBackOn() {  
 **return this**.**checkBackOn**;  
 }  
 **public double** distanceTo(SelectedLocation location) {  
 **try** {  
 **double** tempLatitude = Double.*parseDouble*(location.**latitude**);  
 **double** tempLongitude = Double.*parseDouble*(location.**longitude**);  
 **double** xLatitude = Math.*abs*(Double.*parseDouble*(**longitude**) - tempLatitude);  
 **double** yLongitude = Math.*abs*(Double.*parseDouble*(**latitude**) - tempLongitude);  
 *//pathagoras theorem* **distance** = Math.*sqrt*((xLatitude \* xLatitude) + (yLongitude \* yLongitude));  
 } **catch** (NullPointerException e) {  
 e.printStackTrace();  
 }  
 **return distance**;  
 }  
}

1. **TourMutation.java:**

**package** aayush.randompatrolling;  
**import** java.text.CharacterIterator;  
**public class** TourMutation {  
 **private static final double *mutuationRate*** = 0.015;  
 **public static final int *competitionSize*** = 5;  
 **private static final boolean *enchance*** = **true**;  
 **public static** TourPopulation evolveTourPopuation(TourPopulation tPopulation) {  
 TourPopulation newTourPopulation = **new** TourPopulation(tPopulation.tourPopulationSize(), **false**);  
 **int** enchancementValue = 0;  
 **if** (***enchance***) {  
 newTourPopulation.saveTour(0, tPopulation.getFittestTour());  
 enchancementValue = 1;  
 }  
 *//Crossover tours  
 //loop over new tours size to create individual/baby from current tours* **for** (**int** i = enchancementValue; i < newTourPopulation.tourPopulationSize(); i++) {  
 addPlace mom = *competitionSelection*(tPopulation);  
 addPlace dad = *competitionSelection*(tPopulation);  
 addPlace baby = *crossover*(mom, dad);  
 *//add baby to new tours* newTourPopulation.saveTour(i, baby);  
 }  
 *//Mutate and add genetics* **for** (**int** i = enchancementValue; i < newTourPopulation.tourPopulationSize(); i++) {  
 *mutate*(newTourPopulation.getTour(i));  
 }  
 **return** newTourPopulation;  
 }  
 *//crossover parents mom and dad to create babies* **public static** addPlace crossover(addPlace mom, addPlace dad) {  
 addPlace baby = **new** addPlace();  
  
 **int** startPosition = (**int**) (Math.*random*() \* mom.tourSize());  
 **int** endPosition = (**int**) (Math.*random*() \* dad.tourSize());  
  
 **for** (**int** i = 0; i < baby.tourSize(); i++) {  
 **if** (startPosition < endPosition && i < endPosition && i > startPosition) {  
 baby.setLocation(i, mom.getLocation(1));  
 } **else if** (startPosition > endPosition) {  
 **if** (!(i < startPosition && i > endPosition)) {  
 baby.setLocation(i, mom.getLocation(i));  
 }  
 }  
 }  
 **for** (**int** i = 0; i < dad.tourSize(); i++) {  
 **if**(!baby.hasLocation(dad.getLocation(i))){  
 **for**(**int** j = 0; j < baby.tourSize(); j++){  
 **if**(baby.getLocation(j)==**null**){  
 baby.setLocation(i, dad.getLocation(i));  
 **break**;  
 }  
 }  
 }  
 }  
 **return** baby;  
 }  
 *//Mutate a tour using swap mutation* **private static void** mutate(addPlace tour){  
 **for**(**int** i=0;i<tour.tourSize();i++){  
 **if**(Math.*random*() < ***mutuationRate***){  
 **int** j = (**int**) (tour.tourSize()\*Math.*random*());  
 SelectedLocation location1 = tour.getLocation(i);  
 SelectedLocation location2 = tour.getLocation(j);  
 *//swap* tour.setLocation(j, location1);  
 tour.setLocation(i, location2);  
 }  
 }  
 }  
 **private static** addPlace competitionSelection(TourPopulation tPopulation1){  
 *//create competiting tour population* TourPopulation competition = **new** TourPopulation(***competitionSize***,**false**);  
 *//for each competiton get a random tour and add it* **for**(**int** i=0;i<***competitionSize***; i++){  
 **int** randomIndex = (**int**) (Math.*random*()\*tPopulation1.tourPopulationSize());  
 competition.saveTour(i,tPopulation1.getTour(randomIndex));  
 }  
 addPlace fittest = competition.getFittestTour();  
 **return** fittest;  
 }  
}

1. **TourPopulation.java:**

**package** aayush.randompatrolling;  
**public class** TourPopulation {  
 addPlace[] **tours**;  
 **public** TourPopulation(**int** tourPopulationSize, **boolean** start){  
 **tours** = **new** addPlace[tourPopulationSize];  
 **if**(start){  
 **for**(**int** i=0;i<tourPopulationSize; i++){  
 addPlace newTour = **new** addPlace();  
 newTour.generateIndividual();  
 saveTour(i, newTour);  
 }  
 }  
 }  
 *//saves tour to a index* **public void** saveTour(**int** i, addPlace tour){  
 **tours**[i] = tour;  
 }  
 *//get tour through a index* **public** addPlace getTour(**int** i){  
 **return tours**[i];  
 }  
 *//Get the shortest(in GA Terms fittest) tour overall* **public** addPlace getFittestTour(){  
 addPlace fittestTour = **tours**[0];  
  
 **for**(**int** i=1;i < **tours**.**length**;i++){  
 **if**(fittestTour.getFitness() <= getTour(i).getFitness()){  
 fittestTour = getTour(i);  
 }  
 }  
 **return** fittestTour;  
 }  
 **public int** tourPopulationSize(){  
 **return tours**.**length**;  
 }  
}

1. **tspCalculator.java:**

**package** aayush.randompatrolling;  
**import** android.widget.TextView;  
**import** java.util.ArrayList;  
**public class** tspCalculator {  
 ArrayList<SelectedLocation> **LocationList**;  
 addPlace **addPlace1** = **new** addPlace();  
 placeManager **place** = **new** placeManager();  
 **public** tspCalculator() {  
 }  
 **public** String getTspRoute() {  
**LocationList** = **place**.getdestinationList();  
 *//start Population* TourPopulation tours = **new** TourPopulation(30, **true**);  
 *//Evolve* tours = TourMutation.*evolveTourPopuation*(tours);  
 **for** (**int** i = 0; i < 100; i++) {  
 tours = TourMutation.*evolveTourPopuation*(tours);  
 }  
 String result = **"Best Distance "** + tours.getFittestTour().getTourDistance() + **"\n\n Solution \n\n"** + tours.getFittestTour();  
 **return** result;  
 }  
}

1. **activity\_add\_alarm.xml:**

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="aayush.randompatrolling.addAlarm"**>  
 <**LinearLayout  
 android:layout\_width="0dp"  
 android:layout\_height="443dp"  
 android:orientation="vertical"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 tools:layout\_constraintTop\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 android:layout\_marginTop="8dp"  
 android:layout\_marginStart="11dp"  
 android:layout\_marginEnd="11dp"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintTop\_toTopOf="parent"**>  
 <**TextView  
 android:id="@+id/Title"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:paddingBottom="5dp"  
 android:paddingEnd="15dp"  
 android:paddingTop="5dp"  
 android:text="@string/title"  
 android:textSize="14dp"** />  
 <**EditText  
 android:id="@+id/titlebox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:inputType="textPersonName"** />  
 <**TextView  
 android:id="@+id/Message"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:paddingBottom="5dp"  
 android:paddingEnd="15dp"  
 android:paddingTop="5dp"  
 android:text="@string/message"  
 android:textSize="14dp"** />  
 <**EditText  
 android:id="@+id/messageBox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:inputType="textMultiLine"** />  
 </**LinearLayout**>  
 <**Button  
 android:id="@+id/addAlarmButton"  
 android:layout\_width="186dp"  
 android:layout\_height="55dp"  
 android:layout\_marginBottom="5dp"  
 android:layout\_marginLeft="4dp"  
 android:background="#333"  
 android:text="@string/addAlarm"  
 android:textColor="#fff"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 tools:layout\_constraintBottom\_creator="1"  
 tools:layout\_constraintLeft\_creator="1"  
 android:layout\_marginStart="4dp"** />  
 <**Button  
 android:id="@+id/alarmBack"  
 android:layout\_width="0dp"  
 android:layout\_height="56dp"  
 android:background="#333"  
 android:text="@string/back"  
 android:textColor="#fff"  
 tools:layout\_constraintRight\_creator="1"  
 android:layout\_marginStart="2dp"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintBaseline\_toBaselineOf="@+id/addAlarmButton"  
 tools:layout\_constraintBaseline\_creator="1"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintLeft\_toRightOf="@+id/addAlarmButton"  
 app:layout\_constraintHorizontal\_bias="0.0"** />  
</**android.support.constraint.ConstraintLayout**>

1. **activity\_add\_place.xml:**

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="aayush.randompatrolling.addPlace"**>  
 <**LinearLayout  
 android:layout\_width="0dp"  
 android:layout\_height="443dp"  
 android:orientation="vertical"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 tools:layout\_constraintTop\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 android:layout\_marginTop="8dp"  
 android:layout\_marginStart="11dp"  
 android:layout\_marginEnd="11dp"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintTop\_toTopOf="parent"**>  
 <**TextView  
 android:id="@+id/AddressName"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:paddingBottom="5dp"  
 android:paddingEnd="15dp"  
 android:paddingTop="5dp"  
 android:text="@string/address"  
 android:textSize="14dp"** />  
 <**EditText  
 android:id="@+id/address"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="@string/gotoselectMap"  
 android:inputType="textPersonName"** />  
 <**TextView  
 android:id="@+id/minTimeToStay"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:paddingBottom="5dp"  
 android:paddingEnd="15dp"  
 android:paddingTop="5dp"  
 android:text="@string/minTime"  
 android:textSize="14dp"** />  
 <**EditText  
 android:id="@+id/minTimeStay"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:inputType="textPersonName"** />  
 <**TextView  
 android:id="@+id/maxTimeToStay"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:paddingBottom="5dp"  
 android:paddingEnd="15dp"  
 android:paddingTop="5dp"  
 android:text="@string/maxTime"  
 android:textSize="14dp"** />  
 <**EditText  
 android:id="@+id/maxTimeStay"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:inputType="textPersonName"** />  
 <**TextView  
 android:id="@+id/priorityOfPlace"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:paddingBottom="5dp"  
 android:paddingEnd="15dp"  
 android:paddingTop="5dp"  
 android:text="@string/priority"  
 android:textSize="14dp"** />  
 <**EditText  
 android:id="@+id/placePriority"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="@string/priority"  
 android:inputType="textPersonName"** />  
 <**TextView  
 android:id="@+id/durationToCheckBackOn"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:paddingBottom="5dp"  
 android:paddingEnd="15dp"  
 android:paddingTop="5dp"  
 android:text="@string/checkbackon"  
 android:textSize="14dp"** />  
 <**EditText  
 android:id="@+id/checkBackDuration"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="@string/xhours"  
 android:inputType="textPersonName"** />  
 </**LinearLayout**>  
 <**Button  
 android:id="@+id/addPlaceButton"  
 android:layout\_width="186dp"  
 android:layout\_height="55dp"  
 android:background="#333"  
 android:text="@string/addPlace"  
 android:textColor="#fff"  
 tools:layout\_constraintBottom\_creator="1"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintLeft\_toLeftOf="parent"** />  
 <**Button  
 android:id="@+id/placesBack"  
 android:layout\_width="0dp"  
 android:layout\_height="56dp"  
 android:background="#333"  
 android:text="@string/back"  
 android:textColor="#fff"  
 tools:layout\_constraintRight\_creator="1"  
 android:layout\_marginStart="2dp"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintBaseline\_toBaselineOf="@+id/addPlaceButton"  
 tools:layout\_constraintBaseline\_creator="1"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintLeft\_toRightOf="@+id/addPlaceButton"** />  
</**android.support.constraint.ConstraintLayout**>

1. **activity\_alarms.xml:**

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="aayush.randompatrolling.alarms"  
 tools:layout\_editor\_absoluteY="81dp"  
 tools:layout\_editor\_absoluteX="0dp"**>  
 <**Button  
 android:id="@+id/addAlarm"  
 android:layout\_width="0dp"  
 android:layout\_height="52dp"  
 android:background="#333"  
 android:text="@string/add\_Alarm"  
 android:textColor="#fff"  
 app:layout\_constraintRight\_toLeftOf="@+id/back"  
 tools:layout\_constraintRight\_creator="1"  
 app:layout\_constraintBaseline\_toBaselineOf="@+id/back"  
 tools:layout\_constraintBaseline\_creator="1"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.0"  
 android:layout\_marginEnd="4dp"** />  
 <**Button  
 android:id="@+id/back"  
 android:layout\_width="190dp"  
 android:layout\_height="52dp"  
 android:background="#333"  
 android:text="@string/back"  
 android:textColor="#fff"  
 tools:layout\_constraintTop\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 tools:layout\_constraintBottom\_creator="1"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 android:layout\_marginTop="24dp"  
 app:layout\_constraintTop\_toBottomOf="@+id/listView"  
 app:layout\_constraintVertical\_bias="1.0"  
 app:layout\_constraintRight\_toRightOf="parent"** />  
  
 <**ListView  
 android:id="@+id/listView"  
 android:layout\_width="0dp"  
 android:layout\_height="423dp"  
 tools:layout\_constraintTop\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 app:layout\_constraintRight\_toRightOf="parent"  
 android:layout\_marginTop="4dp"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"** />  
</**android.support.constraint.ConstraintLayout**>

1. **activity\_display\_location\_info.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="aayush.randompatrolling.Display\_location\_info"**>  
 <**ScrollView  
 android:id="@+id/scrollView"  
 android:layout\_width="0dp"  
 android:layout\_height="435dp"  
 tools:layout\_constraintTop\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 tools:layout\_constraintBottom\_creator="1"  
 android:layout\_marginStart="8dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 android:layout\_marginEnd="8dp"  
 app:layout\_constraintRight\_toRightOf="parent"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 android:layout\_marginRight="8dp"  
 android:layout\_marginLeft="8dp"  
 android:layout\_marginTop="9dp"  
 android:layout\_marginBottom="9dp"  
 app:layout\_constraintHorizontal\_bias="0.0"  
 app:layout\_constraintVertical\_bias="0.0"**>  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"** >  
 <**TextView  
 android:id="@+id/viewData"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:paddingStart="30px"  
 android:paddingEnd="30px"  
 android:textSize="16dp"  
 android:text="TextView"** />  
 </**LinearLayout**>  
 </**ScrollView**>  
 <**Button  
 android:id="@+id/back"  
 android:layout\_width="117dp"  
 android:layout\_height="0dp"  
 android:layout\_marginBottom="8dp"  
 android:text="@string/back"  
 android:textColor="#fff"  
 android:background="#333"  
 android:layout\_marginLeft="8dp"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 tools:layout\_constraintBottom\_creator="1"  
 android:layout\_marginStart="8dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"** />  
</**android.support.constraint.ConstraintLayout**>

1. **activity\_login.xml:**

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="aayush.randompatrolling.Login"  
 android:background="#4d8fac"**>  
 <**EditText  
 android:id="@+id/user\_username"  
 android:layout\_width="0dp"  
 android:layout\_height="53dp"  
 android:ems="10"  
 android:hint="@string/username"  
 android:inputType="textPersonName"  
 tools:ignore="LabelFor"  
 tools:layout\_constraintLeft\_creator="1"  
 tools:layout\_constraintTop\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 app:layout\_constraintRight\_toRightOf="@+id/user\_password"  
 android:layout\_marginTop="27dp"  
 app:layout\_constraintTop\_toBottomOf="@+id/Sign\_in"  
 app:layout\_constraintLeft\_toLeftOf="@+id/user\_password"  
 app:layout\_constraintHorizontal\_bias="0.0"** />  
  
 <**Button  
 android:id="@+id/sign\_in\_button"  
 android:layout\_width="0dp"  
 android:layout\_height="52dp"  
 android:layout\_marginEnd="1dp"  
 android:layout\_marginTop="22dp"  
 android:text="@string/Sign\_in"  
 android:background="#44333333"  
 android:textColor="#fff"  
 app:layout\_constraintTop\_toBottomOf="@+id/user\_password"  
 tools:layout\_constraintRight\_creator="1"  
 tools:layout\_constraintTop\_creator="1"  
 android:layout\_marginStart="1dp"  
 app:layout\_constraintRight\_toRightOf="@+id/user\_password"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintLeft\_toLeftOf="@+id/user\_password"  
 app:layout\_constraintHorizontal\_bias="0.444"** />  
 <**TextView  
 android:id="@+id/Sign\_in"  
 android:layout\_width="88dp"  
 android:layout\_height="34dp"  
 android:layout\_marginStart="120dp"  
 android:layout\_marginTop="33dp"  
 android:text="@string/Sign\_in"  
 android:textAlignment="center"  
 android:textAppearance="@style/TextAppearance.AppCompat"  
 android:textSize="24sp"  
 app:layout\_constraintTop\_toTopOf="parent"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 android:layout\_marginLeft="120dp"  
 tools:ignore="RtlHardcoded"** />  
 <**EditText  
 android:id="@+id/user\_password"  
 android:layout\_width="308dp"  
 android:layout\_height="49dp"  
 android:layout\_marginLeft="27dp"  
 android:layout\_marginStart="27dp"  
 android:layout\_marginTop="16dp"  
 android:ems="10"  
 android:hint="@string/password"  
 android:inputType="textPassword"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/user\_username"  
 tools:layout\_constraintTop\_creator="1"  
 tools:ignore="RtlHardcoded"** />  
 <**Button  
 android:id="@+id/register\_button"  
 android:layout\_width="0dp"  
 android:layout\_height="48dp"  
 android:layout\_marginEnd="2dp"  
 android:layout\_marginStart="2dp"  
 android:layout\_marginTop="11dp"  
 android:text="@string/register"  
 android:background="#44333333"  
 android:textColor="#fff"  
 app:layout\_constraintHorizontal\_bias="0.25"  
 app:layout\_constraintLeft\_toLeftOf="@+id/user\_password"  
 app:layout\_constraintRight\_toRightOf="@+id/user\_password"  
 app:layout\_constraintTop\_toBottomOf="@+id/sign\_in\_button"  
 tools:layout\_constraintLeft\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 tools:layout\_constraintTop\_creator="1"** />  
</**android.support.constraint.ConstraintLayout**>

1. **activity\_main.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="aayush.randompatrolling.MainActivity"  
 tools:layout\_editor\_absoluteY="81dp"  
 tools:layout\_editor\_absoluteX="0dp"  
 android:background="#4d8faa"**>  
 <**TextView  
 android:id="@+id/textView"  
 android:layout\_width="338dp"  
 android:layout\_height="103dp"  
 android:layout\_marginTop="57dp"  
 android:paddingBottom="5dp"  
 android:paddingTop="5dp"  
 android:text="@string/app\_name"  
 android:textAlignment="center"  
 android:textColor="#f6f6f6"  
 android:textSize="35sp"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 tools:layout\_constraintTop\_creator="1"** />  
 <**ImageView  
 android:id="@+id/imageView"  
 android:layout\_width="261dp"  
 android:layout\_height="196dp"  
 app:srcCompat="@mipmap/map8"  
 tools:layout\_constraintRight\_creator="1"  
 tools:layout\_constraintBottom\_creator="1"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 tools:layout\_constraintLeft\_creator="1"  
 android:layout\_marginBottom="76dp"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.504"  
 android:layout\_marginTop="8dp"  
 app:layout\_constraintTop\_toBottomOf="@+id/textView"  
 app:layout\_constraintVertical\_bias="1.0"** />  
</**android.support.constraint.ConstraintLayout**>

1. **activity\_map\_select.xml**

<**fragment xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:map="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/map"  
 android:name="com.google.android.gms.maps.SupportMapFragment"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="aayush.randompatrolling.mapSelect"** />

1. **Activity\_maps.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 tools:layout\_editor\_absoluteY="81dp"  
 tools:layout\_editor\_absoluteX="0dp"**>  
 <**fragment  
 android:id="@+id/map"  
 android:name="com.google.android.gms.maps.SupportMapFragment"  
 android:layout\_width="413dp"  
 android:layout\_height="match\_parent"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 tools:context="aayush.randompatrolling.MapsActivity"  
 tools:layout\_constraintBottom\_creator="1"  
 tools:layout\_constraintLeft\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 tools:layout\_constraintTop\_creator="1"  
 app:layout\_constraintHorizontal\_bias="1.0"  
 app:layout\_constraintVertical\_bias="0.0"** />  
 <**Button  
 android:id="@+id/places"  
 android:layout\_width="0dp"  
 android:layout\_height="50dp"  
 android:layout\_alignParentBottom="true"  
 android:layout\_alignParentStart="true"  
 android:layout\_marginEnd="1dp"  
 android:background="#333"  
 android:textColor="#fff"  
 android:text="@string/places"  
 app:layout\_constraintBaseline\_toBaselineOf="@+id/alarms"  
 app:layout\_constraintHorizontal\_bias="1.0"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toLeftOf="@+id/alarms"  
 tools:layout\_constraintBaseline\_creator="1"  
 tools:layout\_constraintLeft\_creator="1"  
 tools:layout\_constraintRight\_creator="1"** />  
 <**Button  
 android:id="@+id/alarms"  
 android:layout\_width="190dp"  
 android:layout\_height="50dp"  
 android:layout\_alignParentBottom="true"  
 android:layout\_alignParentEnd="true"  
 android:layout\_alignTop="@+id/places"  
 android:layout\_marginEnd="1dp"  
 android:background="#333"  
 android:text="@string/alarms"  
 android:textColor="#fff"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintRight\_toRightOf="@+id/map"  
 tools:layout\_constraintBottom\_creator="1"  
 tools:layout\_constraintRight\_creator="1"** />  
</**android.support.constraint.ConstraintLayout**>

1. **activity\_places.xml:**

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="aayush.randompatrolling.places"  
 tools:layout\_editor\_absoluteY="81dp"  
 tools:layout\_editor\_absoluteX="0dp"**>  
 <**Button  
 android:id="@+id/addPlaceButton"  
 android:layout\_width="186dp"  
 android:layout\_height="55dp"  
 android:background="#333"  
 android:text="@string/addPlace"  
 android:textColor="#fff"  
 tools:layout\_constraintBottom\_creator="1"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintLeft\_toLeftOf="parent"** />  
 <**Button  
 android:id="@+id/placesBack"  
 android:layout\_width="0dp"  
 android:layout\_height="56dp"  
 android:background="#333"  
 android:text="@string/back"  
 android:textColor="#fff"  
 tools:layout\_constraintRight\_creator="1"  
 android:layout\_marginStart="2dp"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintBaseline\_toBaselineOf="@+id/addPlaceButton"  
 tools:layout\_constraintBaseline\_creator="1"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintLeft\_toRightOf="@+id/addPlaceButton"** />  
 <**TextView  
 android:id="@+id/result"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:layout\_marginBottom="13dp"  
 android:layout\_marginEnd="18dp"  
 android:layout\_marginStart="18dp"  
 android:layout\_marginTop="7dp"  
 android:text="TextView"  
 android:textSize="16dp"  
 app:layout\_constraintBottom\_toTopOf="@+id/addPlaceButton"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 tools:layout\_constraintBottom\_creator="1"  
 tools:layout\_constraintLeft\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 tools:layout\_constraintTop\_creator="1"** />  
</**android.support.constraint.ConstraintLayout**>

1. **Activity\_register.xml:**

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="aayush.randompatrolling.register"**>  
 <**EditText  
 android:id="@+id/register\_username"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:layout\_marginBottom="21dp"  
 android:layout\_marginEnd="10dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginStart="10dp"  
 android:layout\_marginTop="7dp"  
 android:ems="10"  
 android:hint="@string/username"  
 android:inputType="textPersonName"  
 app:layout\_constraintBottom\_toTopOf="@+id/register\_password"  
 app:layout\_constraintHorizontal\_bias="0.0"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.0"  
 tools:ignore="HardcodedText,MissingConstraints,RtlHardcoded"  
 tools:layout\_constraintBottom\_creator="1"  
 tools:layout\_constraintLeft\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 tools:layout\_constraintTop\_creator="1"** />  
 <**EditText  
 android:id="@+id/register\_password"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginStart="10dp"  
 android:ems="10"  
 android:hint="@string/password"  
 android:inputType="textPassword"  
 tools:ignore="HardcodedText,MissingConstraints,RtlHardcoded"  
 tools:layout\_constraintTop\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 android:layout\_marginEnd="10dp"  
 app:layout\_constraintRight\_toRightOf="parent"  
 android:layout\_marginTop="75dp"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.0"** />  
 <**EditText  
 android:id="@+id/register\_Name"  
 android:layout\_width="0dp"  
 android:layout\_height="59dp"  
 android:layout\_marginTop="12dp"  
 android:ems="10"  
 android:hint="@string/name"  
 android:inputType="textPersonName"  
 app:layout\_constraintHorizontal\_bias="0.0"  
 app:layout\_constraintLeft\_toLeftOf="@+id/register\_password"  
 app:layout\_constraintRight\_toRightOf="@+id/register\_password"  
 app:layout\_constraintTop\_toBottomOf="@+id/register\_password"  
 tools:ignore="MissingConstraints,RtlHardcoded"  
 tools:layout\_constraintLeft\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 tools:layout\_constraintTop\_creator="1"** />  
 <**EditText  
 android:id="@+id/register\_user\_type"  
 android:layout\_width="0dp"  
 android:layout\_height="53dp"  
 android:ems="10"  
 android:hint="UserType (User or Admin)"  
 android:inputType="text"  
 tools:ignore="HardcodedText,MissingConstraints,RtlHardcoded"  
 tools:layout\_constraintTop\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 app:layout\_constraintRight\_toRightOf="@+id/register\_Name"  
 app:layout\_constraintTop\_toBottomOf="@+id/register\_Name"  
 tools:layout\_constraintLeft\_creator="1"  
 app:layout\_constraintLeft\_toLeftOf="@+id/register\_Name"  
 android:layout\_marginTop="83dp"  
 app:layout\_constraintHorizontal\_bias="0.0"** />  
 <**EditText  
 android:id="@+id/register\_organization"  
 android:layout\_width="0dp"  
 android:layout\_height="53dp"  
 android:layout\_marginTop="11dp"  
 android:ems="10"  
 android:hint="@string/organization"  
 android:inputType="text"  
 app:layout\_constraintHorizontal\_bias="1.0"  
 app:layout\_constraintLeft\_toLeftOf="@+id/register\_Name"  
 app:layout\_constraintRight\_toRightOf="@+id/register\_Name"  
 app:layout\_constraintTop\_toBottomOf="@+id/register\_Name"  
 tools:ignore="HardcodedText,MissingConstraints,RtlHardcoded"  
 tools:layout\_constraintLeft\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 tools:layout\_constraintTop\_creator="1"** />  
 <**Button  
 android:id="@+id/register\_button"  
 android:layout\_width="0dp"  
 android:layout\_height="48dp"  
 android:text="@string/register"  
 tools:ignore="MissingConstraints,RtlHardcoded"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 tools:layout\_constraintTop\_creator="1"  
 tools:layout\_constraintRight\_creator="1"  
 android:layout\_marginStart="58dp"  
 android:layout\_marginEnd="58dp"  
 android:layout\_marginTop="21dp"  
 app:layout\_constraintTop\_toBottomOf="@+id/register\_user\_type"  
 tools:layout\_constraintLeft\_creator="1"** />  
</**android.support.constraint.ConstraintLayout**>

# 13) Reference

1)Staff, I (2017). *Game Theory*. [online] Investopedia. Available at: http://www.investopedia.com/terms/g/gametheory.asp.

2) Rao, Y., Li, P., Shao, X. and Shi, K. (2006). Agile manufacturing system control based on cell re-configuration. *International Journal of Production Research*, 44(10), pp.1881-1905.

3) Anon, (2017). [online] Available at: <https://www.researchgate.net/publication/266892481_A_Study_of_the_Agile_Software_Development_Methods_Applicability_and_Implications_in_Industry>

4) Begel, A., Nagappan, N. and Nagappan, N. (2017). *Usage and Perceptions of Agile Software Development in an Industrial Context: An Exploratory Study*. [online] Microsoft Research. Available at: <https://www.microsoft.com/en-us/research/publication/usage-and-perceptions-of-agile-software-development-in-an-industrial-context-an-exploratory-study/>