AUTO POOLING

A Project Report submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Goutham Rohan	Sai Subash	Ganesh Bhaskar	Sarath Chandra
1210315612	1210315621	1210315636	1210315660

Under the esteemed guidance of

Mr. Dasari Veera Reddy,

Asst. Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

GITAM

(Deemed to be University)

VISAKHAPATNAM

MARCH 2019

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM INSTITUTE OF TECHNOLOGY

GITAM

(Deemed to be University)

DECLARATION

We, hereby declare that the project review entitled "**Auto Pooling**" is an original work done in the Department of Computer Science and Engineering, GITAM Institute of Technology, GITAM (Deemed to be University) submitted in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering. The work has not been submitted to any other college or University for the award of any degree or diploma.

Date: 02/04/2019

Registration No.	Name	Signature
1210315612	D. Goutham Rohan	
1210315621	K. Sri Sai Subash	
1210315636	P. Ganesh Bhaskar	
1210315660	G. Sarath Chandra	

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM INSTITUTE OF TECHNOLOGY

GITAM

(Deemed to be University)

CERTIFICATE

This is to certify that the project report entitled “**Auto Pooling**” is a bonafide record of work carried out by Goutham Rohan (1210315612), Sri Sai Subash (1210315621), Ganesh Bhaskar (1210315636) and Sarath Chandra (1210315660) students submitted in partial fulfillment of requirement for the award of degree of Bachelors of Technology in Computer Science and Engineering.

Project Guide	**Head of Department**
Mr. Dasari Veera Reddy	Dr. K. Thammi Reddy
Assistant Professor	Professor

ACKNOWLEDGEMENT

We would like to thank our project guide **Mr. DASARI VEERA REDDY, Assistant Professor,** Department of Computer Science Engineering, for his stimulating guidance and profuse assistance. We shall always cherish our association with him for his guidance, encouragement and valuable suggestions throughout the progress of this work. We consider it a great privilege to work under his guidance and constant support.

We would like to thank our Project Coordinators **Dr. T. SITA MAHALAKSHMI, Professor, Dept. of CSE** and **Dr. P. ANURADHA, Associate Professor, Dept. of CSE** in helping to complete the project by taking frequent reviews and for their valuable suggestions throughout the progress of this work.

We consider it as a privilege to express our deepest gratitude to **Dr. K. THAMMI REDDY, Head of the Department, Computer Science and Engineering** for his valuable suggestions and constant motivation that greatly helped us to successfully complete the project work.

Our sincere thanks to **Dr. K. LAKSHMI PRASAD, Principal, GITAM Institute of Technology, GITAM (Deemed to be University)** for inspiring us to learn new technologies and tools.

Finally, we deem it a great pleasure to thank one and all who helped us directly and indirectly throughout this project.

By

Goutham Rohan	1210315612
Sri Sai Subash	1210315621
Ganesh Bhaskar	1210315636
Sarath Chandra	1210315660

Table of Contents

List of Figures

1. Abstract

The project primarily aims to reduce the cost of transportation from one location to another location in a city through the concept of Pooling i.e., sharing the space available in a vehicle.. It mainly targets the students belonging to one respective college, people belonging to same community. The application groups the people who are willing to travel in same route in a particular interval of time and gets connected with the nearest auto-rickshaw. Since the application concentrates on the students, we are choosing the mobile platform for the application development as mobiles became part of one's life. An android application with a good interface for the users will be built and a sophisticated database is used.

2. Introduction

Auto Pooling (also, ride-sharing and lift-sharing) is the sharing of journeys so that more than one person travels, and prevents the need for others to have to drive to a location themselves.

By having more people using one vehicle, auto-pooling reduces each person's travel costs such as: fuel costs, tolls, and the stress of driving. Auto-pooling is also a more environmentally friendly and sustainable way to travel as sharing journeys reduces air pollution, auto emissions, traffic congestion on the roads, and the need for parking spaces.

Authorities often encourage auto-pooling, especially during periods of high pollution or high fuel prices. Auto sharing is a good way to use up the full seating capacity of a auto, which would otherwise remain unused if it were just the driver using the auto.

In 2009, auto-pooling represented 43.5% of all trips in the United States and 10% of commute trips. The majority of auto-pool commutes (over 60%) are "fam-pools" with family members.

Auto-pool commuting is more popular for people who work in places with more jobs nearby, and who live in places with higher residential densities. Auto-pooling is significantly correlated with transport operating costs, including fuel prices and commute length, and with measures of social capital, such as time spent with others, time spent eating and drinking and being unmarried.

However, auto-pooling is significantly less likely among people who spend more time at work, elderly people, and homeowners

3. Problem Identification and Objectives

On an average about 30% of travel time is done empty in searching for the passengers by an auto-rickshaw. People used to travel in different vehicles by third party transports like cabs, etc. in between the same places, paying higher prices and also leads to wasteful usage of fossil fuels along with environmental damage.

So, considering scenario of a student in a university, there is probability for large no of students trying to book a cab for same route with less number of cabs available w.r.t the demand. As a result, the surge feature within the cab application will automatically rises the fare for the travel. This makes the ride costly.

In addition to them, limited scope of a person, lack of awareness of fellow passengers that are willing to share their/s ride.

So this project aims at resolving the above issues through reduction in the consumption of fossil fuels through efficient usage car space available via Pooling. Enable customers to complete their rides economically, when we're riding using 3rd party applications. Reduce the number of vehicles running on roads there by resulting in bringing down the carbon emissions into the atmosphere.

4. Evolution of Proposed System

Existing System

A person using a 3rd party taxi application providing the source and destination details. As a result, he/she will be provided with the various categories of cab services available at that time along with the estimation charge and seating capacity of the vehicle.

The price shown is dynamic and varies based on the number of cab requests raised at the time and is charged same for 1/2/3/4 persons.

Disadvantages

- Dynamic charging structure would make rides costly and is not viable for a single person to travel.
- A person is not aware of whether there are any fellow passengers travelling along the same route (to same destination).
- Cars travel the same routes, with empty space available resulting in more number of vehicles on roads and more consumption of fossil fuels.

Proposed System

A simple android application that takes input (travelling details) from the user that includes source, destination, date of journey and departure time.

There will be space provided for the end-user to search for passengers between 2 points that will be giving out the list of persons travelling in that route.

Advantages

- It bridges the gap of connectivity through providence of details.
- Make rides economical as the space in the car is shared.

5. System Methodology

On entering the application, the user registers (if new user) else log in to the application. He/she is displayed with 3 options to be chosen from. A user would post his journey details so that he/she is made available at the time searching.

Searching for co-passenger’s option would display the details of the users that are available for that journey. A user would now contact manually the persons and complete their ride and successfully shares the price.

The Architecture of the model is described in fig.6.1.

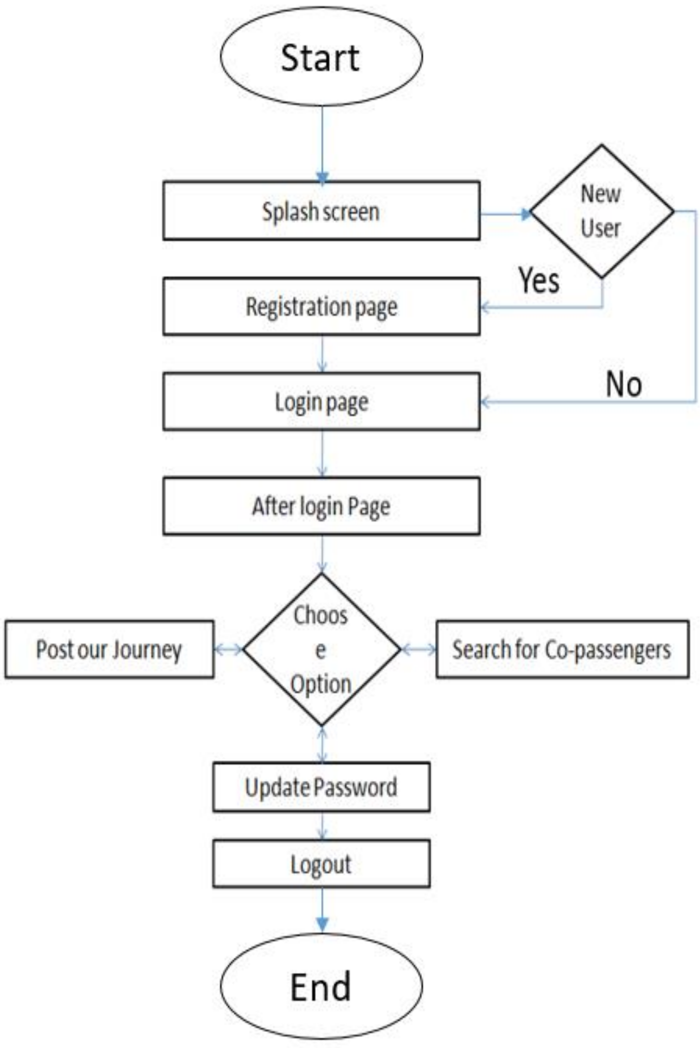


Fig.6.1. Architecture of the model

6. Overview of Technologies

6.1.Android

Android is a mobile operating system developed by Google, based on a modified version of the Linux kernel and other open source software and designed primarily for touch screen mobile devices such as smart phones and tablets.

Android has been the best-selling OS worldwide on smart phones since 2011 and on tablets since 2013. As of May 2017, it has over two billion monthly active users, the largest installed base of any operating system, and as of June 2018, the Google Play store features over 3.3 million apps.

Android has a growing selection of third-party applications, which can be acquired by users by downloading and installing the application's APK (Android application package) file, or by downloading them using an application store program that allows users to install, update, and remove applications from their devices. Google Play Store is the primary application store installed on Android devices that comply with Google's compatibility requirements and license the Google Mobile Services software.

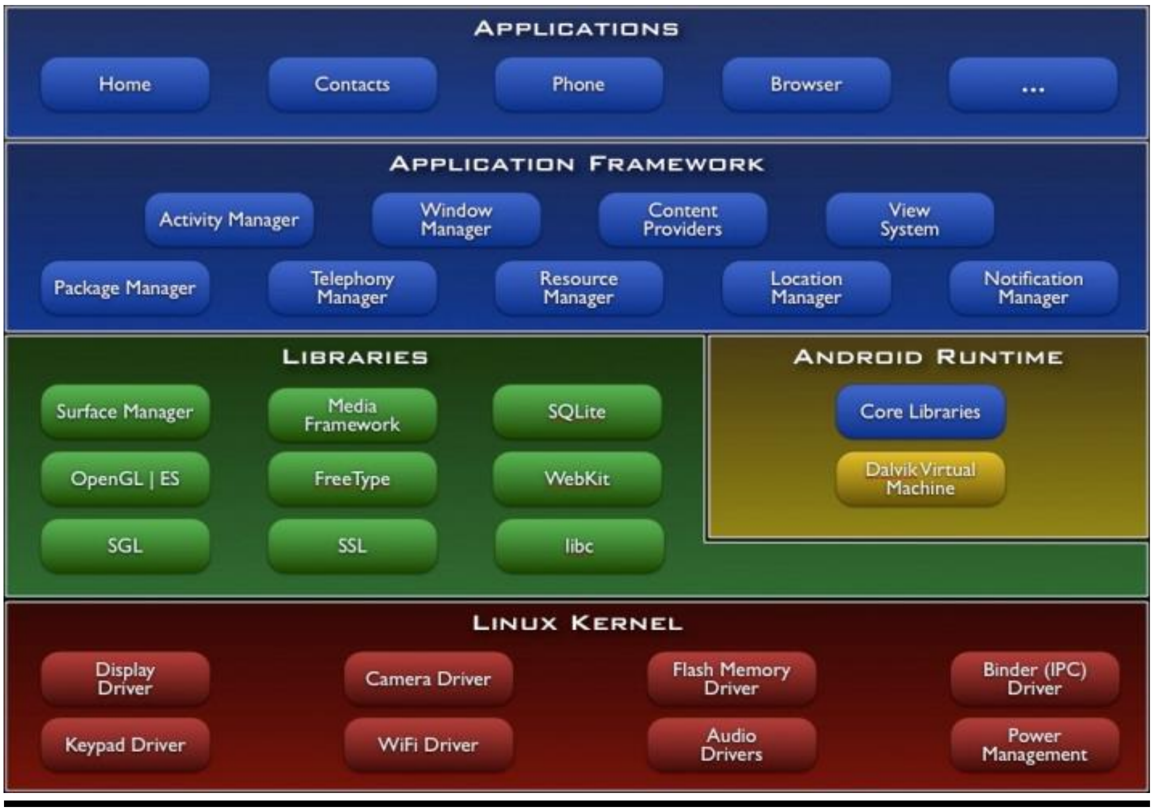


Fig.7.1. Android Architecture

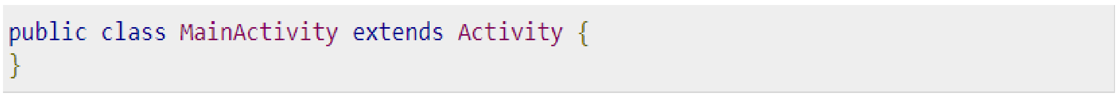
6.2. Android Components

Application components are the essential building blocks of an Android application. These components are loosely coupled by the application manifest file AndroidManifest.xml that describes each component of the application and how they interact.

6.2.1. Activities

An activity represents a single screen with a user interface, in-short Activity performs actions on the screen. For example, an email application might have one activity that shows a list of new emails, another activity to compose an email, and another activity for reading emails. If an application has more than one activity, then one of them should be marked as the activity that is presented when the application is launched.

An activity is implemented as a subclass of **Activity** class as follows-



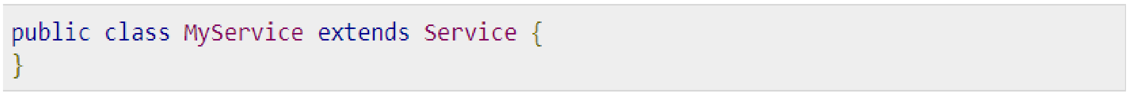
```
public class MainActivity extends Activity {
}
```

Fig.7.2.1 Implementation of Activity

6.2.2. Services

A service is a component that runs in the background to perform long-running operations. For example, a service might play music in the background while the user is in a different application, or it might fetch data over the network without blocking user interaction with an activity.

A service is implemented as a subclass of **Service** class as follows –



```
public class MyService extends Service {
}
```

Fig.7.2.2 Implementation of Services

6.2.3. Content Providers

A content provider component supplies data from one application to others on request. Such requests are handled by the methods of the ContentResolver class. The data may be stored in the file system, the database or somewhere else entirely.

A content provider is implemented as a subclass of ContentProvider class and must implement a standard set of APIs that enable other applications to perform transactions.

```
public class MyContentProvider extends  ContentProvider {
   public void onCreate(){}
}
```

Fig.7.2.3 Implementation of Content Providers

6.2.4. Broadcast Receivers

Broadcast Receivers simply respond to broadcast messages from other applications or from the system. For example, applications can also initiate broadcasts to let other applications know that some data has been downloaded to the device and is available for them to use, so this is broadcast receiver who will intercept this communication and will initiate appropriate action. A broadcast receiver is implemented as a subclass of BroadcastReceiver class and each message is broadcaster as an Intent object.

```
public class MyReceiver  extends  BroadcastReceiver {
   public void onReceive(context,intent){}
}
```

Fig.7.2.4 Implementation of Broadcast Receivers

6.3. Android Runtime

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the java programming language.

Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple VMs efficiently. The Dalvik VM executes files in the Dalvik Executable **(.**dex**)** format which is optimized for minimal memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been transformed into the .dex format by the included “dx” tool.

The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management.

6.4. Hardware Running Android

The main supported platform for Android is the ARM architecture. The Android OS can be used as an operating system for cell phones, netbooks and tablets, including the TV and other devices. The first commercially available phone to run the Android operating system was the HTC Dream, released on 22 October 2008. In early 2010 Google collaborated with HTC to launch its flagship Android device, the Nexus one. This was followed later in 2010 with the Samsung -made Nexus.

Android applications are packaged in .apk format and stored under /data/app folder on the Android OS (the folder is accessible to root user only for security reasons). APK package contains **.**dex files (compiled byte code files called Dalvik executables**),** resource files, etc.

6.5. Generation of APK

Android uses a special virtual machine e.g. the Dalvik Virtual Machine. Dalvik uses special byte code. Therefore you cannot run standard Java bytecode on Android. Android provides a tool “dx” which allows converting Java Class files into “dex” (Dalvik Executable) files. Android applications are packed into an **.**apk (Android package) file by the program “aapt” (Android Asset packaging Tool).

6.6. SQLite Database

SQLite is an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. The code for SQLite is in the public domain and is thus free for use for any purpose, commercial or private. SQLite is the most widely deployed database in the world with more applications than we can count, including several high-profile projects.

SQLite is an embedded SQL database engine. Unlike most other SQL databases, SQLite does not have a separate server process. SQLite reads and writes directly to ordinary disk files. A complete SQL database with multiple tables, indices, triggers, and views, is contained in a single disk file. The database file format is cross-platform - you can freely copy a database between 32-bit and 64-bit systems or between big-endian and little-endian architectures.

These features make SQLite a popular choice as an Application File Format. SQLite database files are a recommended storage format by the US Library of Congress. SQLite is a compact library. With all features enabled, the library size can be less than 600KiB, depending on the target platform and compiler optimization settings. (64-bit code is larger. And some compiler optimizations such as aggressive function in lining and loop unrolling can cause the object code to be much larger.)

There is a trade-off between memory usage and speed. SQLite generally runs faster the more memory you give it. Nevertheless, performance is usually quite good even in low-memory environments. Depending on how it is used, SQLite can be faster than direct file system I/O.

6.7.Java

Java is a general purpose programming language that is concurrent, class based, object-based, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to "byte code" that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture.

The language derives much of its original features from Small-Talk, with syntax similar to C and C++, but it has fewer low-level facilities than either of them. As of 2016, Java was one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers.

The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun had relicensed most of its Java technologies under the GNU General Public License.

Meanwhile, others have developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java (byte code compiler), GNU Class-path (standard libraries), and Iced Tea Web (browser plugin for applets).

The latest version is Java SE 12, released in March 2019, which is a currently supported long-term support (LTS) version by Oracle. Since Java 9 is no longer supported, Oracle advises its users to "immediately transition" to Java 12.

Oracle released the last public update for the legacy Java 8 LTS, which is free for commercial use, in January 2019. Java 8 will be supported with public updates for personal use up to at least December 2020. Oracle and others "highly recommend that you uninstall older versions of Java" because of serious risks due to unresolved security issues.

6.7.1. Setting up path for Windows

Assuming you have installed Java in c:\Program Files\java\jdk directory −

1. Right-click on 'My Computer' and select 'Properties'.
2. Click the 'Environment variables' button under the 'Advanced' tab.
3. Now, alter the 'Path' variable so that it also contains the path to the Java executable. Example, if the path is currently set to 'C:\WINDOWS\SYSTEM32', then change your path to read 'C:\WINDOWS\SYSTEM32;c:\Program Files\java\jdk\bin'.

6.7.2. Setting up path for Linux, UNIX, Solaris, FreeBSD

Environment variable PATH should be set to point to where the Java binaries have been installed. Refer to your shell documentation, if you have trouble doing this.

Example, if you use bash as your shell, then you would add the following line to the end of your '.bashrc: export PATH = /path/to/java:$PATH'.

6.8.XML

Extensible Markup Language (XML) is a simple, very flexible text format derived from SGML (ISO 8879). Originally designed to meet the challenges of large-scale electronic publishing, XML is also playing an increasingly important role in the exchange of a wide variety of data on the Web and elsewhere.

It is a text-based markup language derived from Standard Generalized Markup Language (SGML).XML tags identify the data and are used to store and organize the data, rather than specifying how to display it like HTML tags, which are used to display the data. XML is not going to replace HTML in the near future, but it introduces new possibilities by adopting many successful features of HTML.

There are three important characteristics of XML that make it useful in a variety of systems and solutions −

1. **XML is extensible** − XML allows you to create your own self-descriptive tags, or language, that suits your application.
2. **XML carries the data, does not present it** − XML allows you to store the data irrespective of how it will be presented.
3. **XML is a public standard** − XML was developed by an organization called the World Wide Web Consortium (W3C) and is available as an open standard.

6.8.1. XML Usage

A short list of XML usage says it all −

1. XML can work behind the scene to simplify the creation of HTML documents for large web sites.
2. XML can be used to exchange the information between organizations and systems.
3. XML can be used for offloading and reloading of databases.
4. XML can be used to store and arrange the data, which can customize your data handling needs.
5. XML can easily be merged with style sheets to create almost any desired output.
6. Virtually, any type of data can be expressed as an XML document.

6.8.2. What is Markup?

XML is a markup language that defines set of rules for encoding documents in a format that is both human-readable and machine-readable.

Markup is information added to a document that enhances its meaning in certain ways, in that it identifies the parts and how they relate to each other. More specifically, a markup language is a set of symbols that can be placed in the text of a document to demarcate and label the parts of that document.

Following example shows how XML markup looks, when embedded in a piece of text –

```
<message>
   <text>Hello, world!</text>
</message>
```

Fig.8.1.1 XML markup embedded in a text

This snippet includes the markup symbols, or the tags such as <message>...</message> and <text>... </text>. The tags <message> and </message> mark the start and the end of the XML code fragment. The tags <text> and </text> surround the text Hello, world!

6.8.3. XML Declaration

The XML document can optionally have an XML declaration. It is written as follows –

```
<?xml version = "1.0" encoding = "UTF-8"?>
```

Fig.8.1.2 XML declaration

Where version is the XML version and encoding specifies the character encoding used in the document.

Syntax Rules for XML Declaration

1. The XML declaration is case sensitive and must begin with "**<?**xml**>**" where "xml" is written in lower-case.
2. If document contains XML declaration, then it strictly needs to be the first statement of the XML document.
3. The XML declaration strictly needs be the first statement in the XML document.
4. An HTTP protocol can override the value of *encoding* that you put in the XML declaration.

7. Implementation

7.1. Coding

7.1.1. Splash screen

XML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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=".SplashScreen"
    android:orientation="vertical"
    android:layout_gravity="center"
    android:gravity="center"
    android:background="@drawable/formsbkg">
    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/connect" />
</LinearLayout>
```

JAVA

```
package com.example.booking;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import java.sql.Time;
import java.util.Timer;
import java.util.TimerTask;
public class SplashScreen extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_splash_screen);
        new Timer().schedule(new TimerTask() {
            @Override
            public void run() {
                Intent intent = new Intent(SplashScreen.this,LoginPage.class);
                startActivity(intent);
            }
        },3000);
    }
```

7.1.2. Login screen

XML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
android:background="@drawable/formsbkg"
tools:context=".LoginPage">

<ScrollView
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:scrollbars="vertical">

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginLeft="40dp"
android:layout_marginTop="50dp"
android:layout_marginRight="50dp"
android:orientation="vertical">

<TextView
android:layout_width="match_parent"
android:layout_height="80dp"
android:gravity="center"
android:text="Login Page"
android:textAllCaps="true"
android:textColor="@color/colorNavyBlue"
android:textSize="32dp"
android:textStyle="bold" />

<EditText
android:id="@+id/txtusername"
android:layout_width="match_parent"
android:layout_height="74dp"
android:layout_marginTop="36dp"
android:ems="10"
android:hint="UserName" />

<EditText

android:id="@+id/txtpassword"
```

```
android:layout_width="match_parent"
android:layout_height="74dp"
android:ems="10"
android:hint="Password"
android:inputType="textPassword" />

<Button
android:id="@+id/btnlogin"
android:layout_width="match_parent"
android:layout_height="54dp"
android:layout_marginTop="50dp"
android:text="Login"
android:textColor="@color/colorPrimary" />

<TextView
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:gravity="center"
android:text="Or"
android:textAllCaps="true"
android:textStyle="bold" />

<Button
android:id="@+id/btnregister"
android:layout_width="match_parent"
android:layout_height="54dp"
android:layout_marginTop="10dp"
android:text="register"
android:textColor="@color/colorPrimary" />

</LinearLayout>
</ScrollView>
</LinearLayout>
```

JAVA

```
package com.example.booking;
import android.content.Intent;
import android.database.Cursor;
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;

public class LoginPage extends AppCompatActivity {
    UserRegistartionHelper userRegistartionHelper;
    Button _btnlogin,_btnregister;
    EditText _txtusername, _txtpassword;
    String username ="", password="";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login_page);

        _txtusername = (EditText) findViewById(R.id.txtusername);
        _txtpassword = (EditText) findViewById(R.id.txtpassword);
        _btnlogin = (Button) findViewById(R.id.btnlogin);
        _btnregister = (Button)findViewById(R.id.btnregister);
        userRegistartionHelper = new UserRegistartionHelper(this);

        _btnregister.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(LoginPage.this, Register.class);
                startActivity(intent);
            }
        });

        _btnlogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                 username = _txtusername.getText().toString();
                 password = _txtpassword.getText().toString();

                if(username.isEmpty() || password.isEmpty()){
                    Toast.makeText(getApplicationContext(), "Fields are missing",
Toast.LENGTH_SHORT).show();
                }
                else{
                    searchforusers(username, password);
                }
            }
        });
    }

    public void searchforusers(String u_name, String pass) {
        Cursor results = userRegistartionHelper.getAllUsers(u_name,pass);
        if (results.getCount() == 0){
            Toast.makeText(getApplicationContext(), "Incorrect Username or Password",
Toast.LENGTH_SHORT).show();
```

```
        }
        else {
            Intent intent = new Intent(LoginPage.this, Afterlogin.class);
            startActivity(intent);
            Toast.makeText(getApplicationContext(), "Welcome User",
Toast.LENGTH_SHORT).show();
        }
        }
    }
```

7.1.3. Register screen

XML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:context=".Register"
    android:background="@drawable/formsbkg"
    >

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:scrollbars="vertical">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="40dp"
            android:layout_marginTop="50dp"
            android:layout_marginRight="50dp"
            android:orientation="vertical">

            <TextView
                android:layout_width="match_parent"
                android:layout_height="80dp"
                android:gravity="center"
                android:text="registration Details"
                android:textAllCaps="true"
                android:textColor="@color/colorNavyBlue"
                android:textSize="32dp"
                android:textStyle="bold" />
        <EditText
            android:id="@+id/txtregusername"
            android:layout_width="match_parent"
            android:layout_height="74dp"
            android:layout_marginTop="36dp"
            android:ems="10"
            android:hint="Choose Username" />

        <EditText
            android:id="@+id/txtregpassword"
            android:layout_width="match_parent"
            android:layout_height="74dp"
            android:ems="10"
            android:hint="Choose Password"
            android:inputType="textPassword" />

        <Button
            android:id="@+id/btnregister"
            android:layout_width="match_parent"
            android:layout_height="54dp"
            android:text="Register"
            android:textColor="@color/colorPrimary" />

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:text="Or"
            android:textAllCaps="true"
            android:textStyle="bold" />

        <Button
            android:id="@+id/btnlogin"
            android:layout_width="match_parent"
            android:layout_height="54dp"
            android:text="Login"
            android:textColor="@color/colorPrimary" />


    </LinearLayout>
  </ScrollView>
</LinearLayout>
```

JAVA

```
package com.example.booking;
import android.content.Intent;
import android.database.Cursor;
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;

public class Register extends AppCompatActivity {
   UserRegistartionHelper userRegistartionHelper;
   Button _btnregister, _btnlogin;
   EditText _txtregusername, _txtregpassword;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_register);

      userRegistartionHelper = new UserRegistartionHelper(this);
      _txtregusername = (EditText) findViewById(R.id.txtregusername);
      _txtregpassword = (EditText) findViewById(R.id.txtregpassword);
      _btnregister = (Button) findViewById(R.id.btnregister);
      _btnlogin = (Button) findViewById(R.id.btnlogin);


      _btnregister.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            String username = _txtregusername.getText().toString();
            String password = _txtregpassword.getText().toString();
            if(username.isEmpty() || password.isEmpty()){
               Toast.makeText(getApplicationContext(), "Fields are missing",
Toast.LENGTH_SHORT).show();
            }
            else if(username.equals("Sarath") || password.equals("Sarath123")) {
               Toast.makeText(getApplicationContext(), "Already User exists, Choose
some other", Toast.LENGTH_SHORT).show();
            }
            else {
               boolean state = searchforexistingusers(username, password); //Checking
if user already registred
               if (state) {
                  addUser(username, password);
               }
```

```
                else
                {
                    Toast.makeText(getApplicationContext(), "Already User exists,
Choose some other", Toast.LENGTH_SHORT).show();
                }
            }
        }
    });

    _btnlogin.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new Intent(Register.this, LoginPage.class);
            startActivity(intent);
        }
    });

}

public void addUser(String name, String password){
    boolean inserData = userRegistartionHelper.insertUser(name,password);
    if(inserData)
    {
        Toast.makeText(getApplicationContext(), " Succesfully Registered",
Toast.LENGTH_SHORT).show();
        Intent intent = new Intent(Register.this, LoginPage.class);
        startActivity(intent);
    }
    else
    {
        Toast.makeText(getApplicationContext(), "Failed Registration",
Toast.LENGTH_SHORT).show();

    }
}

public boolean searchforexistingusers(String u_name, String pass) {
    Cursor results = userRegistartionHelper.getAllUsers(u_name,pass);
    if (results.getCount() > 0){
        return false; //User already registred
    }
    else
    {
        return true; //New User registration
    }
}
```

7.1.4. AfterLogin screen

XML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
    android:background="@drawable/formsbkg"
    tools:context=".LoginPage">

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:scrollbars="vertical">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="40dp"
            android:layout_marginTop="50dp"
            android:layout_marginRight="50dp"
            android:orientation="vertical">

            <TextView
                android:layout_width="match_parent"
                android:layout_height="80dp"
                android:gravity="center"
                android:text="Login Page"
                android:textAllCaps="true"
                android:textColor="@color/colorNavyBlue"
                android:textSize="32dp"
                android:textStyle="bold" />

            <EditText
                android:id="@+id/txtusername"
                android:layout_width="match_parent"
                android:layout_height="74dp"
                android:layout_marginTop="36dp"
                android:ems="10"
                android:hint="UserName" />

            <EditText
```

```
                android:id="@+id/txtpassword"
                android:layout_width="match_parent"
                android:layout_height="74dp"
                android:ems="10"
                android:hint="Password"
                android:inputType="textPassword" />

            <Button
                android:id="@+id/btnlogin"
                android:layout_width="match_parent"
                android:layout_height="54dp"
                android:layout_marginTop="50dp"
                android:text="Login"
                android:textColor="@color/colorPrimary" />

            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:text="Or"
                android:textAllCaps="true"
                android:textStyle="bold" />

            <Button
                android:id="@+id/btnregister"
                android:layout_width="match_parent"
                android:layout_height="54dp"
                android:layout_marginTop="10dp"
                android:text="register"
                android:textColor="@color/colorPrimary" />


        </LinearLayout>
    </ScrollView>
</LinearLayout>
```

JAVA

```
package com.example.booking;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageButton;

public class Afterlogin extends AppCompatActivity {

    ImageButton _btnpostjourneys,_btnchangepass, _btnsearchpassengers, _btnlogout;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_afterlogin);

        _btnpostjourneys = (ImageButton)findViewById(R.id.btnpostjourneys);
        _btnchangepass = (ImageButton)findViewById(R.id.btnchangepass);
        _btnsearchpassengers = (ImageButton)findViewById(R.id.btnsearchpassengers);
        _btnlogout = (ImageButton)findViewById(R.id.btnlogout);

        _btnpostjourneys.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(Afterlogin.this, MainActivity.class );
                startActivity(intent);
            }
        });

        _btnchangepass.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(Afterlogin.this, UpdateProfile.class );
                startActivity(intent);
            }
        });

        _btnsearchpassengers.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(Afterlogin.this, Search.class );
                startActivity(intent);
            }
        });

        _btnlogout.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(Afterlogin.this, LoginPage.class );
                startActivity(intent);
                finish();
            }
        });
    }
}
```

7.1.5. PostJourneys screen

XML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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=".MainActivity"
    android:orientation="vertical"
    android:background="@drawable/formsbkg"
    >


    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:scrollbars="vertical">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="40dp"
            android:layout_marginTop="20dp"
            android:layout_marginRight="50dp"
            android:orientation="vertical">

            <TextView
                android:layout_width="match_parent"
                android:layout_height="80dp"
                android:gravity="center"
                android:text="Travelling Details"
                android:textAllCaps="true"
                android:textColor="@color/colorNavyBlue"
                android:textSize="32dp"
                android:textStyle="bold" />

            <EditText
                android:id="@+id/txtname"
                android:layout_width="match_parent"
                android:layout_height="74dp"
                android:ems="10"
                android:hint="Name" />

            <EditText
    android:id="@+id/txtsource"
    android:layout_width="match_parent"
    android:layout_height="74dp"
    android:ems="10"
    android:hint="Source"
    android:inputType="textPersonName" />

<EditText
    android:id="@+id/txtdestination"
    android:layout_width="match_parent"
    android:layout_height="74dp"
    android:ems="10"
    android:hint="Destination" />


<EditText
    android:id="@+id/txtmobile"
    android:layout_width="match_parent"
    android:layout_height="74dp"
    android:ems="10"
    android:hint="Mobile No, Do not include +91"
    android:inputType="number"
    android:maxLength="10" />

<EditText
    android:id="@+id/txttime"
    android:layout_width="match_parent"
    android:layout_height="74dp"
    android:ems="10"
    android:focusable="false"
    android:hint="Date"
    android:textIsSelectable="true" />

<EditText
    android:id="@+id/txtslot"
    android:layout_width="match_parent"
    android:layout_height="74dp"
    android:ems="10"
    android:focusable="false"
    android:hint="SLOT" />


<Button
    android:id="@+id/btnsubmit"
    android:layout_width="match_parent"
    android:layout_height="54dp"
```

```
            android:text="Submit"
            android:textColor="@color/colorPrimary" />

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:text="Or"
            android:textAllCaps="true"
            android:textStyle="bold" />

        <Button
            android:id="@+id/btngotosearch"
            android:layout_width="match_parent"
            android:layout_height="54dp"
            android:layout_marginBottom="10dp"
            android:text="Search Passengers"
            android:textColor="@color/colorPrimary" />

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:text=" " />
    </LinearLayout>
  </ScrollView>

</LinearLayout>
```

JAVA

```
package com.example.booking;
import android.app.DatePickerDialog;
import android.app.TimePickerDialog;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.CalendarView;
import android.text.format.DateFormat;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.TimePicker;
import android.widget.Toast;
import java.util.Calendar;
```

```
public class MainActivity extends AppCompatActivity implements
DatePickerDialog.OnDateSetListener, TimePickerDialog.OnTimeSetListener {

    DatabaseHelper databaseHelper;
    Button _btnsubmit,_btngotosearchpassengers;
    EditText _txtname, _txtsource, _txtdestination, _txtmobile, _txttime, _txtslot;
    int day, month, year, hour, minute, AM_PM;
    int dayFinal, monthFinal, yearFinal, hourFinal, minuteFinal;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        _txtname = (EditText) findViewById(R.id.txtname);
        _txtsource = (EditText) findViewById(R.id.txtsource);
        _txtdestination = (EditText) findViewById(R.id.txtdestination);
        _txtmobile = (EditText) findViewById(R.id.txtmobile);
        _txttime = (EditText) findViewById(R.id.txttime);
        _txtslot = (EditText) findViewById(R.id.txtslot);
        _btnsubmit = (Button) findViewById(R.id.btnsubmit);
        _btngotosearchpassengers = (Button)findViewById(R.id.btngotosearch);
        databaseHelper = new DatabaseHelper(this);

        _btnsubmit.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String name = _txtname.getText().toString();
                String source = _txtsource.getText().toString();
                String destination = _txtdestination.getText().toString();
                String mobile = _txtmobile.getText().toString();
                String date = _txttime.getText().toString();
                String time = _txtslot.getText().toString();
                if(name.isEmpty() || source.isEmpty() || destination.isEmpty() ||
mobile.isEmpty() || time.isEmpty() || date.isEmpty()){
                    Toast.makeText(getApplicationContext(), "Fields are missing",
Toast.LENGTH_SHORT).show();
                }
                else if(source.equalsIgnoreCase(destination)){
                    Toast.makeText(getApplicationContext(), "Same src and destintion",
Toast.LENGTH_SHORT).show();
                }
                    else {
                    addData(name, source, destination, time, date, mobile);
                }
            }
        });
        _btngotosearchpassengers.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this, Search.class);
                startActivity(intent);
            }
        });

        _txttime.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Calendar c = Calendar.getInstance();
                year = c.get(Calendar.YEAR);
                month = c.get(Calendar.MONTH);
                day = c.get(Calendar.DAY_OF_MONTH);

                DatePickerDialog datePickerDialog = new
DatePickerDialog(MainActivity.this, MainActivity.this, year,month,day);
                datePickerDialog.show();
            }
        });

    }

    public void addData(String name, String source, String destination, String time,
String date, String mobile){
        boolean inserData =
databaseHelper.addData(name,source,destination,time,date,mobile);
        if(inserData)
        {
            Toast.makeText(getApplicationContext(), " Details Successfully Posted",
Toast.LENGTH_SHORT).show();
            Intent intent = new Intent(MainActivity.this, Afterlogin.class);
            startActivity(intent);

        }
        else
        {
            Toast.makeText(getApplicationContext(), "Failed posting",
Toast.LENGTH_SHORT).show();
        }
    }


    @Override
    public void onDateSet(DatePicker view, int year, int month, int dayOfMonth) {
            yearFinal = year;
            monthFinal = month+1;
            dayFinal = dayOfMonth;

            Calendar c = Calendar.getInstance();
            hour = c.get(Calendar.HOUR_OF_DAY);
            minute = c.get(Calendar.MINUTE);

            TimePickerDialog timePickerDialog = new
TimePickerDialog(MainActivity.this, MainActivity.this, hour, minute,
DateFormat.is24HourFormat(this));
            timePickerDialog.show();
    }

    @Override
    public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
        String AM_PM ="AM";
        hourFinal = hourOfDay;
        minuteFinal = minute;
        if(hourFinal > 12) {
            hourFinal -= 12;
            AM_PM = "PM";
        }
        _txttime.setText(dayFinal+"/"+monthFinal+"/"+yearFinal);
        _txtslot.setText(String.format("%02d:%02d %s", hourFinal,
minuteFinal,AM_PM));
    }
}
```

7.1.6. Update screen

XML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:context=".UpdateProfile"
    android:background="@drawable/formsbkg">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginLeft="40dp"
        android:layout_marginTop="50dp"
        android:layout_marginRight="50dp"
        android:orientation="vertical">

        <TextView
            android:layout_width="match_parent"
            android:layout_height="80dp"
            android:gravity="center"
            android:text="Update Profile"
            android:textAllCaps="true"
            android:textColor="@color/colorNavyBlue"
            android:textSize="32dp"
            android:textStyle="bold" />

        <EditText
            android:id="@+id/txtupdateassword"
            android:layout_width="match_parent"
            android:layout_height="74dp"
            android:ems="10"
            android:hint="Choose new Password"
            android:inputType="textPassword" />

        <Button
            android:id="@+id/btnupdatepassword"
            android:layout_width="match_parent"
            android:layout_height="54dp"
            android:layout_marginTop="50dp"
            android:text="Update"
            android:textColor="@color/colorPrimary" />

    </LinearLayout>
</LinearLayout>
```

JAVA

```
package com.example.booking;
import android.content.Intent;
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.Toast;

public class UpdateProfile extends AppCompatActivity {

    LoginPage loginPage;
    UserRegistartionHelper userRegistartionHelper;
    Button _btnupdatepassword;
    EditText _edittextpassword;



    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_updateprofile);

        _edittextpassword = (EditText) findViewById(R.id.txtupdateassword);
        _btnupdatepassword = (Button) findViewById(R.id.btnupdatepassword);
        loginPage = new LoginPage();
        userRegistartionHelper = new UserRegistartionHelper(this);

        _btnupdatepassword.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String username = loginPage.username;
                String newpassword = _edittextpassword.getText().toString();
                if(newpassword.isEmpty()){
                    Toast.makeText(getApplicationContext(), "Field is empty",
Toast.LENGTH_SHORT).show();
                }
                else{
                    updatepassword(username,newpassword);
                }
            }

        });

    }

    public void updatepassword(String username,String newpass){

        boolean updateData = userRegistartionHelper.updateUser(username, newpass);
        if(updateData){
            Toast.makeText(getApplicationContext(), "Update Successfull",
Toast.LENGTH_SHORT).show();
            Intent intent = new Intent(UpdateProfile.this,LoginPage.class);
            startActivity(intent);
        }
        else
            Toast.makeText(getApplicationContext(), "Update Failed",
Toast.LENGTH_SHORT).show();
    }
```

7.2.Testing

7.2.1. Unit testing

Unit testing is usually conducted as part of a combined code and unit phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

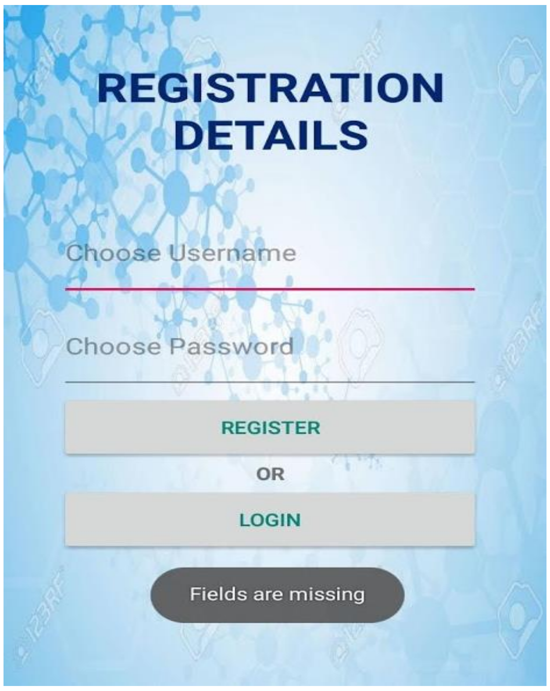
Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify the entries are of the correct format.
- No duplicated entries should be allowed.
- All links should take the user to the correct page.

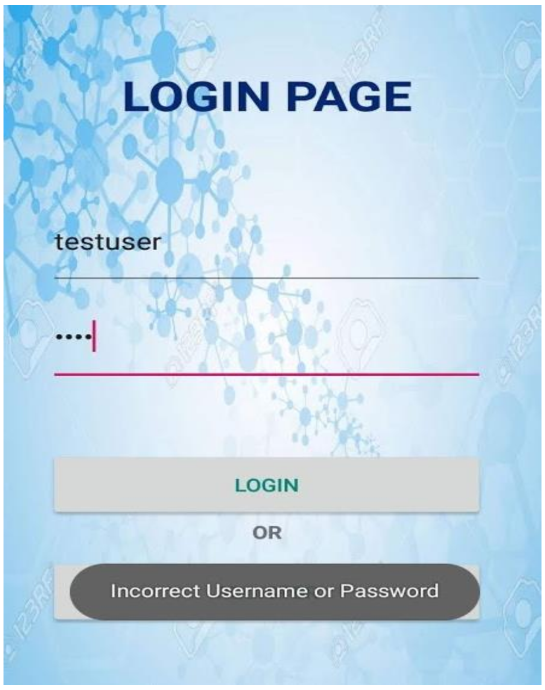

REGISTRATION DETAILS

Choose Username

Choose Password

Fields are missing

Fig.8.2.1 Field entries verification

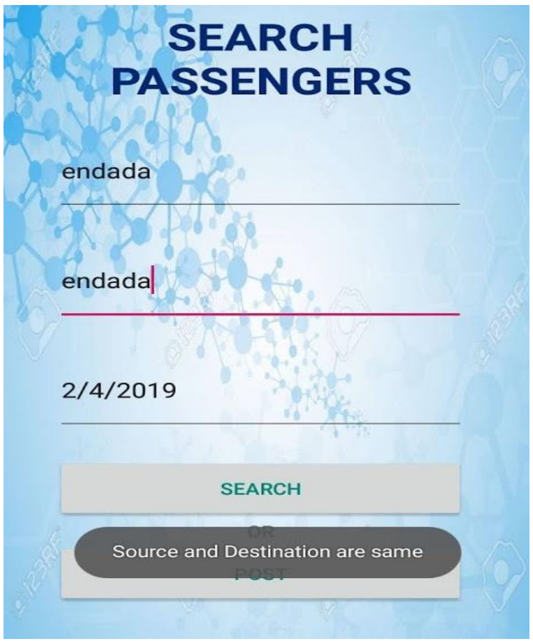

LOGIN PAGE

testuser

••••

Incorrect Username or Password

Fig.8.2.2 Fast Responses

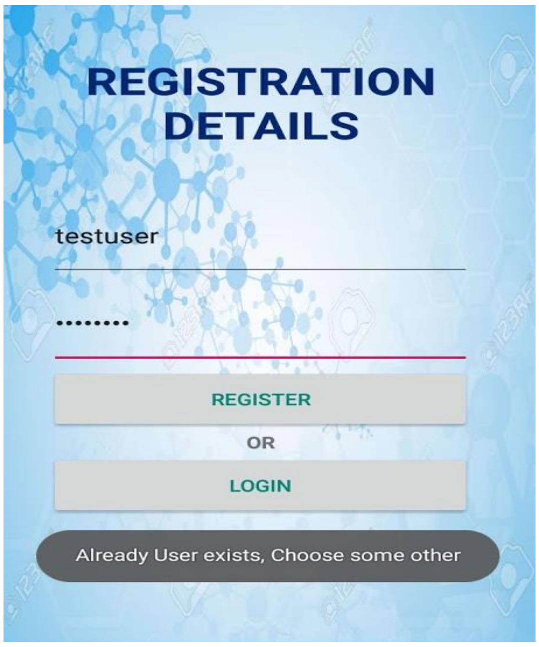

SEARCH PASSENGERS

endada

endada

2/4/2019

Source and Destination are same


REGISTRATION DETAILS

testuser

••••••••

Already User exists, Choose some other

Fig.8.2.3 No Duplicate entries

7.2.2. Integration testing

Software integration testing is the incremental integration of 2 or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one set up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

7.2.3. Acceptance testing

User acceptance testing is a critical phase of any project and requires significant Participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered**.**

8. Results and Discussions

8.1. Splash screen

Fig.9.1 Splash screen

8.2. Registration screen

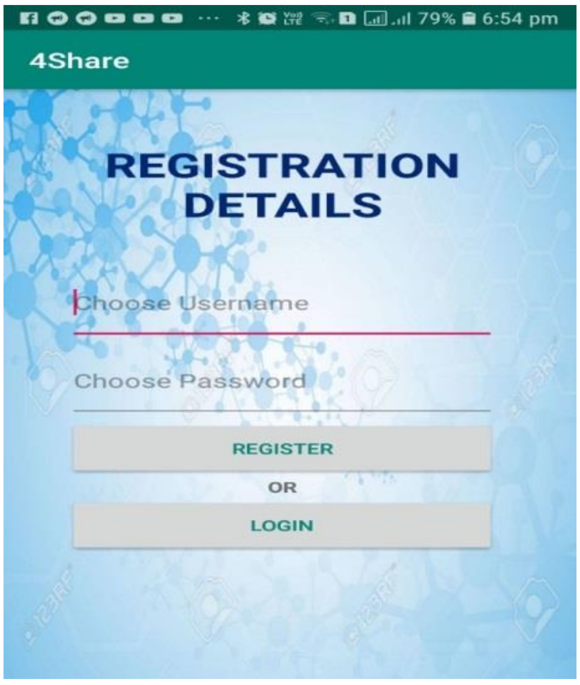


Fig.9.2 Registration page

8.3. Login screen

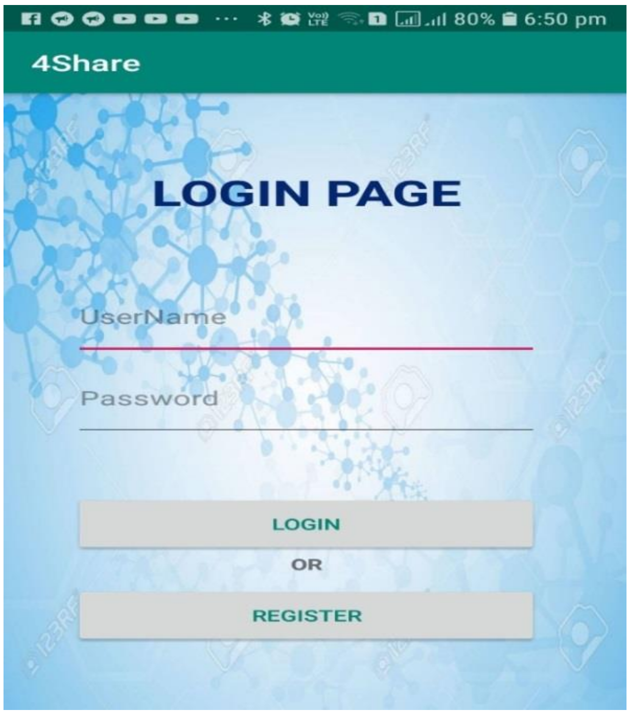


Fig 9.3 Login Page

8.4. AfterLogin screen

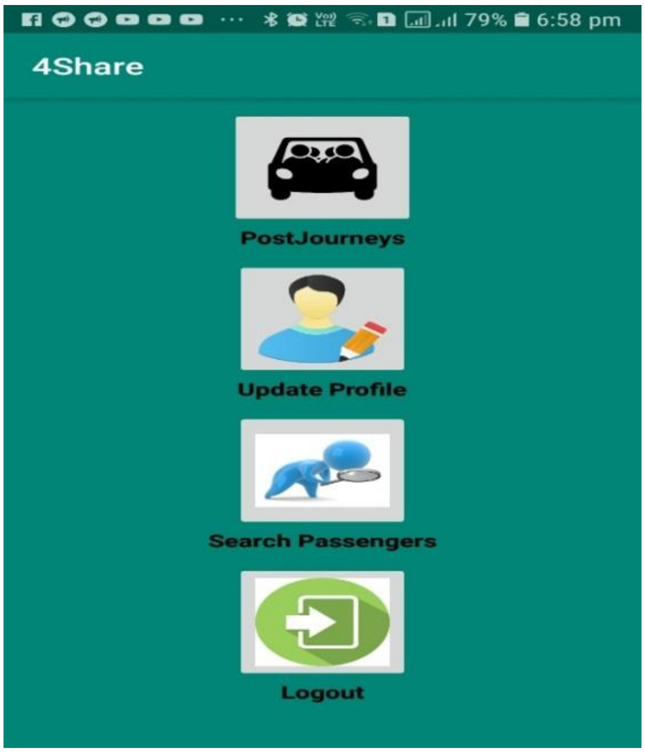
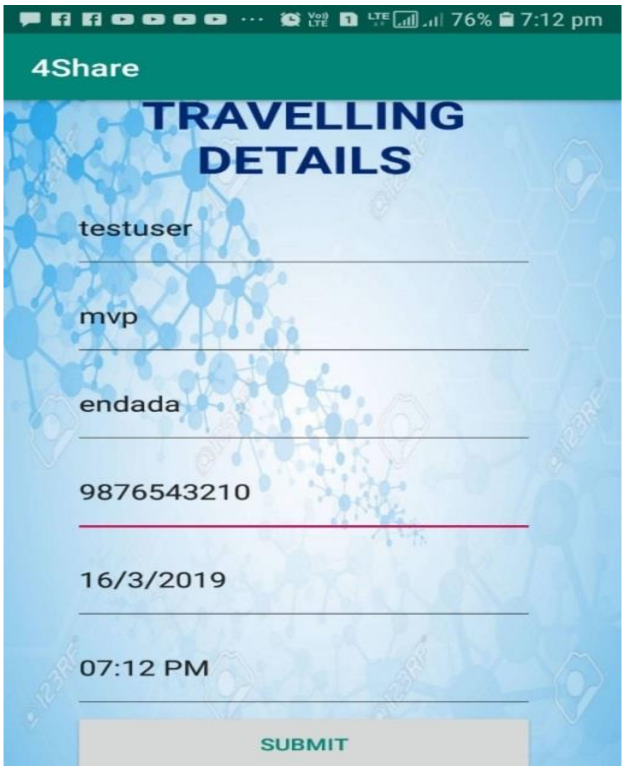


Fig 9.4 AfterLogin Page

8.5. Post Journeys screen


4Share

TRAVELLING DETAILS

testuser

mvp

endada

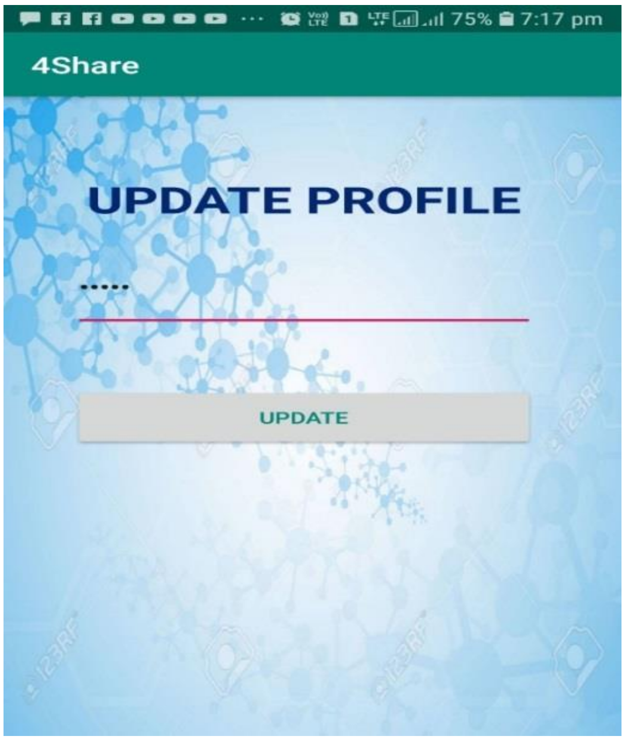
9876543210

16/3/2019

07:12 PM

Fig 9.5 Post Journeys page

8.6. Update Profile screen


4Share

UPDATE PROFILE

......

Fig 9.6 Update profile

8.7. Search Passengers

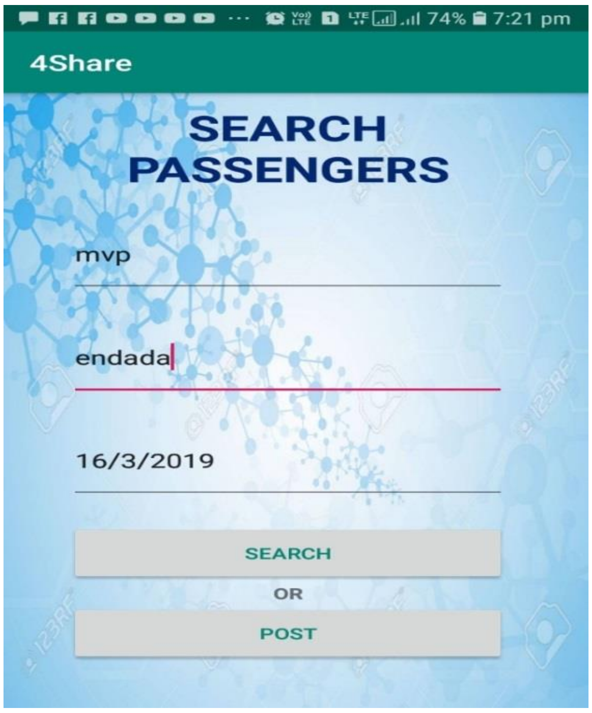
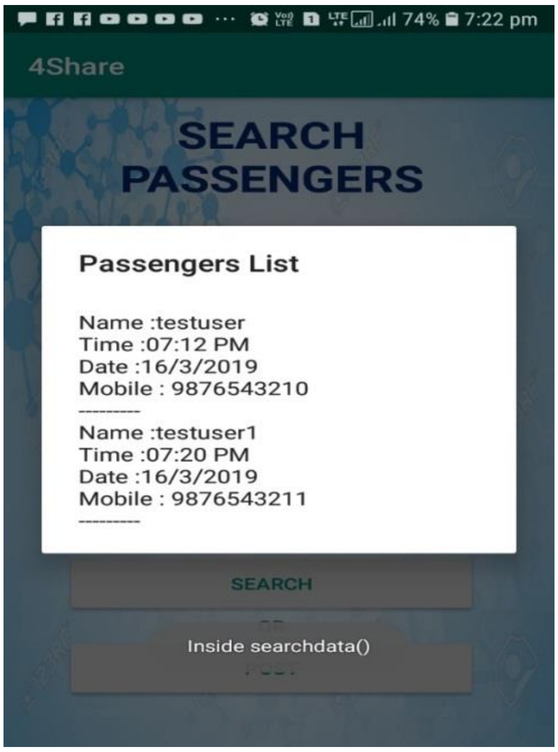


Fig 9.7 Search Passengers



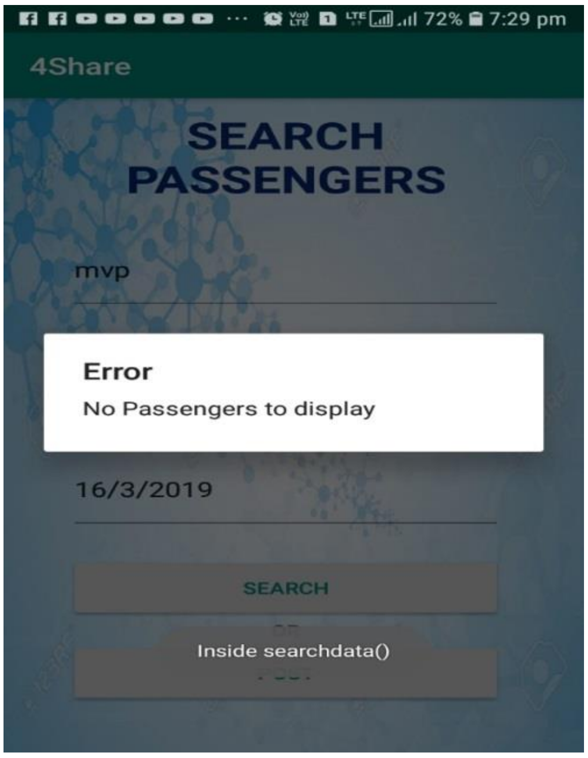
Passengers List

Name :testuser
Time :07:12 PM
Date :16/3/2019
Mobile : 9876543210

Name :testuser1
Time :07:20 PM
Date :16/3/2019
Mobile : 9876543211

Inside searchdata()

Fig 9.8 Successfully found



Error
No Passengers to display

Inside searchdata()

Fig 9.9 No Passengers

9. Conclusion and Future Scope

The problem of increasing fare rate w.r.t the demand in the scenario of “local cab services” is solved in a different manner unlike using the regular solution “Increasing the services with increase in the demand”. Here we instead, grouped people travelling along same direction there by reducing the demand without failing the needs of the customers.

In addition, the lack of connectivity and communication are the other sources which are resolved, by making every user knowing the fellow passengers that are available via posting their journey details and providing a search feature that puts out the list of passengers who are willing to travel.

The problem of high consumption of fossil fuels, damage to the environment, can be addressed in this way as it decreases the vehicle count on roads in considerable numbers.

The scope of project is limited to same community people because of one’s security and can be eliminated through up-gradation of application through implementation of “Live tracking, Emergency call buttons, validation of mobile numbers through usage of OTP (One time password) mechanism, implementation of algorithm that can group people through putting few constraints (gender, age, time of travel, etc.)”.

It could be more successful in countries with huge population densities, mobile users, and is proportional with the urban population whose travelling distance would be high and are willing to have a comfortable rides with economical rates.

This methodology can be integrated or can be implemented in the cab applications directly and avoid using of an additional application. This works all around the world and would reduce the travelling charges compared to the current one.

10. References

1. https://www.openhub.net/p/android/analyses/latest/languages_summary.

2. https://android-developers.googleblog.com/2008/09/announcing-android-10-sdk-release-1.html.

3. https://developer.android.com/preview/download.

4. https://www.android.com/versions/nougat-7-0/.

5. https://android.googlesource.com/platform/bionic/+/master/libc/.

6. https://web.archive.org/web/20160121112754/https://android.googlesource.com/platform/external/mksh/%2B/master.

7. https://android.googlesource.com/platform/external/toybox/+/master/toys/.

8. https://lwn.net/Articles/629362/.

9. https://source.android.com/source/licenses.html.

10. https://www.android.com/gms/.