**1. INTRODUCTION**

This application “Helpers” is an Android application which aims to help the people in solving a problem. It is mainly built to solve problems simply by using mobile application. The overall Application was developed in android for the android mobile users and is also user friendly.We have built an application which can be accessed by multiple categories of technicians. It also helps the user with a new profile to access the information of the registered technicians.The fire base will have the details of the registrants. It can be drawn out if the technician isn’t available or prefers to keep the services down.

* 1. **Purpose**

A systems engineering, a requirement can be a description of what a system must do, referred to as a Functional Requirement. This type of requirement specifies something that the delivered system must be able to do. Another type of requirement specifies something about the system itself, and how well it performs its functions. Such requirements are often called Nonfunctional requirements, or performance requirements or quality of service requirements. Examples of such requirements include usability, availability, reliability, supportability, testability and maintainability.

* 1. **Scope**

Prior to the development of this application, To provide service to the people when required for technicians. Many technicians, who are involved in these app can solve the problems of people in case of household works and complete the task in time.This project not only helps in solving their problems but also connecting the users.

**1.3 Features**

1. Easy to install and run.
2. Quite simple and easy to use.
3. Consumes less storage space.

**2. EXISTING SYSTEM**

In the existing system, the app provides facilities with few technicians. In this system, it provides different type of technicians with different functionalities.The user can only select technician for their requirements. There exists few android applications which do not have all the features and they do not have location tracking facilitybut these app provides the technicians in that specific area. So, we initialized to create an application to reduce the Manual work and stress to save time.

**Disadvantages:**

1. All the damage occurs before the help arrives.
2. Avoiding or preventing the harm is impossible.

**3. PROBLEM STATEMENT**

In today’s world,people are facing difficulties infinding technician to solve their problems specially in home needs. These app provides facilities for user in solving their problems. User can login into the app and select any type of technician for their need and use.

**4. PROPOSED SYSTEM**

In the proposed system, the user overcomes the disadvantage in existing system by using this ‘Helpers’ application. It has the different types of technicians with different functionalities.It helps the user/ technician to save time and energy of looking out for options. Based on the fire base data, the details are easier to access and helpful to maintain the status of the technician.

**Advantages:**

1. Quite simple and easy to use.
2. User can save time and energy.
3. Reduces time in searching for technicians.
4. To help the technicians grow by connecting people across world.
5. User can know about technician status and can select whenever needed.

**5. TECHNOLOGIES USED**

**5.1 Software Interfaces**

We used Java as Programming language for writing the code for the project.ANDROID STUDIO or ECLIPSE IDE for writing the programs. Operating system will be Windows 10. SQLite database server is used for creating the local and global database (server) and JSON format file is used for interchanging the data between server and application.

**Software Requirements**

1. JDK 1.8 and above
2. SDK: Android 4.4(Kitkat) and above
3. Android studio 2.1.10 and above
4. ADT(Android Developer Tools) V.22.3.0
5. Operating System: Windows 7 or 10
6. IDE: Eclipse Version No (optional)

**Hardware Requirements**

1. RAM **:**  2GB Ram or above
2. Hard Disk **:**  80GB or above
3. Processor **:** Dual core or above.

**6. ARCHITECTURE**

**6.1 Software Development Life Cycle (SDLC)**

**6.1.1 SDLC Methodologies**

This document play a vital role in the Software Development Life Cycle (SDLC) as it describes the complete requirement of the system. It means for use by developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

**6.1.2Spiral Model**

Spiral Model was defined by Barry Boehm in his 1988 article, “A spiral Model of Software Development and Enhancement. This model was not the first model to discuss iterative development. The spiral model is similar to the [incremental model](http://istqbexamcertification.com/what-is-incremental-model-advantages-disadvantages-and-when-to-use-it/" \o "What is Incremental model- advantages, disadvantages and when to use it?), with more emphasis placed on risk analysis. The spiral model has four phases: Planning, Risk Analysis, Engineering and Evaluation. A software project repeatedly passes through these phases in iterations (called Spirals in this model). The baseline spirals, starting in the planning phase, requirements are gathered and risk is assessed. Each subsequent spiral builds on the baseline spiral.

As originally envisioned, the iterations were typically 6 months to 2 years long. Each phase starts with a design goal and ends with a client reviewing the progress thus far. Analysis and engineering efforts are applied at each phase of the project, with an eye toward the end goal of the project.

The steps for Spiral Model can be generalized as follows:

* The new system requirements are defined in as much details as possible. This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.
* A preliminary design is created for the new system.
* A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.
* A second prototype is evolved by a fourfold procedure:

1. Evaluating the first prototype in terms of its strengths, weakness, and risks.
2. Defining the requirements of the second prototype.
3. Planning and designing the second prototype.
4. Constructing and testing the second prototype.

* At the customer option, the entire project can be aborted if the risk is deemed too great. Risk factors might involve in development of cost overruns, operating-cost miscalculation, or any other factor that could, in the customer’s judgment, result in a less-than-satisfactory final product.
* The existing prototype is evaluated in the same manner as was the previous prototype, and if necessary, another prototype is developed from it according to the fourfold procedure outlined above.
* The preceding steps are iterated until the customer is satisfied that the refined prototype represents the final product desired.
* The final system is constructed, based on the refined prototype.
* The final system is thoroughly evaluated and tested. Routine maintenance is carried on a continuing basis to prevent large scale failures and to minimize down time.

**The following diagram shows how a spiral model acts like:**

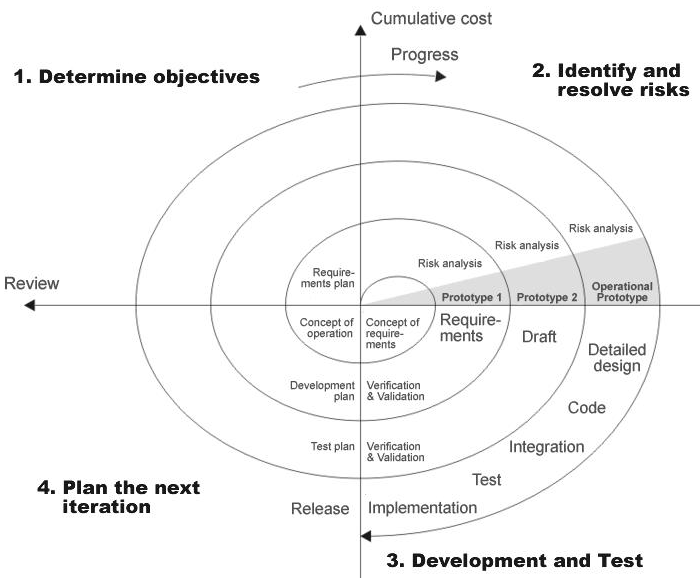


Figure 6.1 shows Spiral model

* **Planning Phase:**Requirements are gathered during the planning phase. Requirements like ‘BRS’ that is ‘Business Requirement Specifications’ and ‘SRS’ that is ‘Software Requirement specifications’.
* **Risk Analysis:** In the**risk analysis phase**, a process is undertaken to identify risk and alternate solutions.  A prototype is produced at the end of the risk analysis phase. If any risk is found during the risk analysis then alternate solutions are suggested and implemented.
* **Engineering Phase:** In this phase software is **developed**, along with testing at the end of the phase. Hence in this phase the development and testing is done.
* **Evaluation phase:**This phase allows the customer to evaluate the output of the project to date before the project continues to the next spiral.

**6.2 Application Architecture**

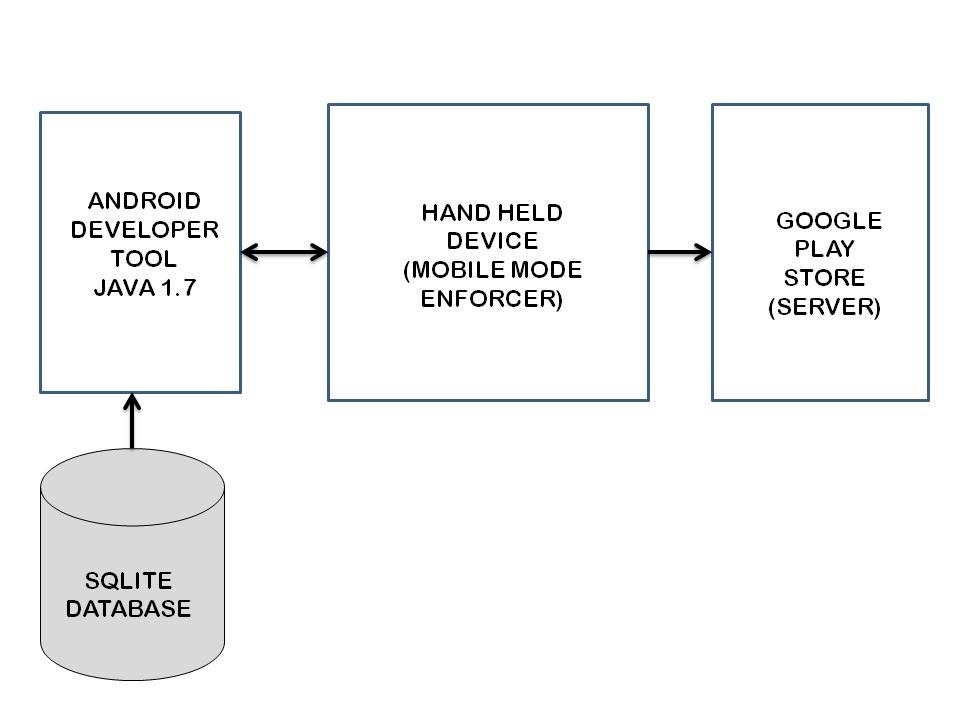
****

Figure 6.2 shows Application Architecture

**7. DESIGN MODULES**

**7.1 Input Design**

Considering the requirements, procedures to collect the necessary input data in most efficiently designed. The input design has been done keeping in view that, the interaction of the user with the system being the most effective and simplified way.

Also the measures are taken for the following:

* + - Controlling the amount of input
    - Eliminating extra steps
    - Keeping the process simple
    - At this stage the input forms and screens are designed.

**7.2 Output Design**

All the screens of the system are designed with a view to provide the user with easy operations in simpler and efficient way, minimum key strokes possible. Instructions and important information is emphasized on the screen. Almost every screen is provided with no error and option selection facilitates. Emphasis is given for speedy processing and speedy transaction between the screens. Each screen assigned to make it as much user friendly as possible by using interactive procedures. So to say user can operate the system without much help from the operating manual.

**7.3Software Requirement Specification**

**7.3.1SRS:**

Software Requirement Specification (SRS) is the starting point of the software developing activity. As system grew more complex it became evident that the goal of the entire system cannot be easily comprehended. Hence the need for the requirement phase arose. The software project is initiated by the client needs. The SRS is the means of translating the ideas of the minds of clients (the input) into a formal document (the output of the requirement phase.)

The SRS phase consists of two basic activities:

**7.3.1.1 Problem/Requirement Analysis:**

The process is order and more nebulous of the two, deals with understand the problem, the goal and constraints.

**7.3.1.2 Requirement Specification:**

Here, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools, and checking the specifications are addressed during this activity.

The Requirement phase terminates with the production of the validate SRS document. Producing the SRS document is the basic goal of this phase.

## 7.3.1.3 Document Conventions:

We have used Times New Roman (text size 12).Bold Font is used for Main Headings (text size of 16). Normal font is used for sub headings (text size of 14).

**Font:** Times New Roman

**Main Heading:** Bold Font

## 7.4 Intended Audience and Reading Suggestions

This document is for better understanding for Remote desktop control. Mainly intended for Head of the Dept., Internal guide, External guide, Staff members, Users and colleagues. This detail given below guides every normal user to how to go through this document for better understanding. The sequence to follow for better understanding is here Purpose, Scope, Features, Operating requirements, Modules present in the project, Advantages, References etc.

**7.4.1 Role of SRS**

The purpose of the Software Requirement Specification is to reduce the communication gap between the clients and the developers. Software Requirement Specification is the medium though which the client and user needs are accurately specified. It forms the basis of software development. A good SRS should satisfy all the parties involved in the system.

## 7.4.2 Scope

This application helps the user to login and select the technicians of different functionalities. This application is user friendly in which the user can select any technician. Many technicians, who are involved in these app can solve the problems of people in case of household works and complete the task in time. This project not only helps in solving their problems but also connecting the users across the world.

**7.5 UML Diagrams**

The Unified Modeling Language (UML) is a standard language for writing software blue prints. The UML is a language for

* Visualizing
* Specifying
* Constructing
* Documenting the artifacts of a software intensive system.

The UML is a language which provides vocabulary and the rules for combining words in that vocabulary for the purpose of communication. A modeling language is a language whose vocabulary and the rules focus on the conceptual and physical representation of a system. Modeling yields an understanding of a system.

**7.5.1 Use case Diagram:**

* The use case model defines the outside (actors) and inside (use case) of the system’s behaviour.
* Use case diagram is quite simple in nature and depicts two types of elements: one representing the business roles and the other representing the business processes.

The given diagram depicts the use-case diagram of the application.

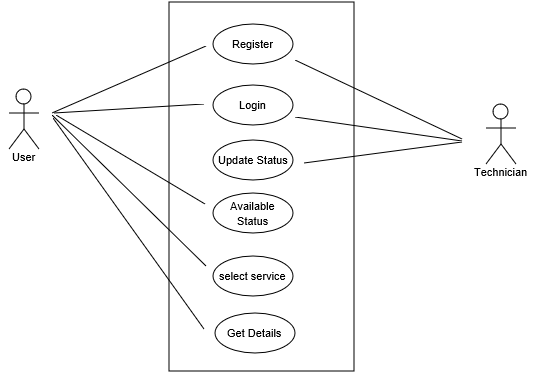


Figure 7.1 Usecase Diagram for “Helpers” application

**7.5.2 Class Diagram:**

* Classes are typically modeled as rectangles with three sections: the top section for the name of the class, the middle section for the attributes of the class, the bottom section for the methods of the class.
* Attributes are the information stored about an object, while methods are the things an object or class do.

The given diagram depicts the class diagram of the application.

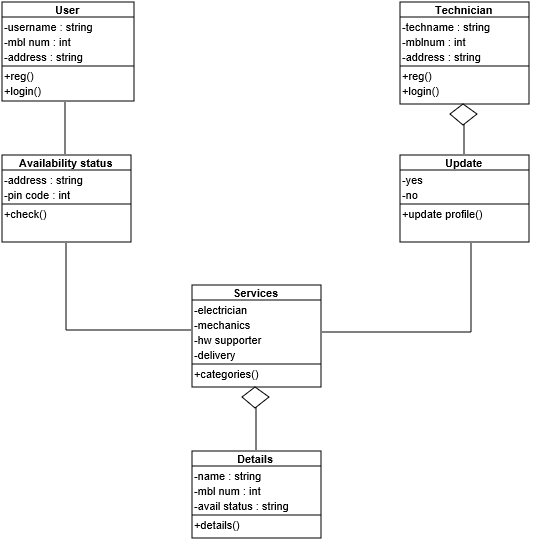


Figure 7.2 Class Diagram for “Helpers” application

**7.5.3 Sequence Diagram:**

* UML sequence diagrams are used to represent the flow of messages, events and actions between the objects or components of a system. Time is represented in the vertical direction showing the sequence of interactions of the header elements, which are displayed horizontally at the top of the diagram.

The given diagram depicts the sequence diagram of the application.

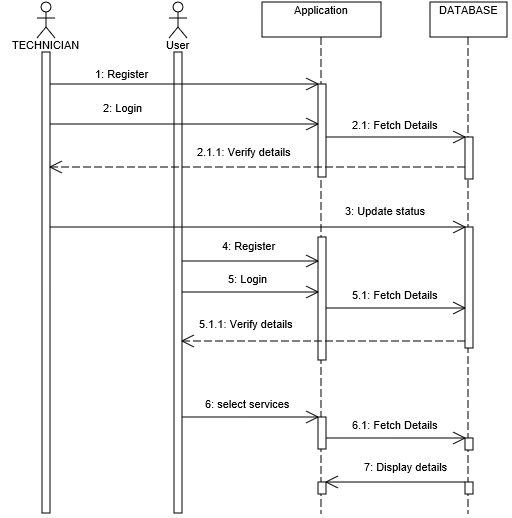


Figure 7.3 Sequence Diagram for “Helpers” application

**7.5.4Activity Diagram:**

* Activity diagrams represent the business and operational workflows of a system.
* An Activity diagram is a dynamic diagram that shows the activity andthe event that causes the object to be in the particular state.

The given diagram depicts the activity diagram of the application.

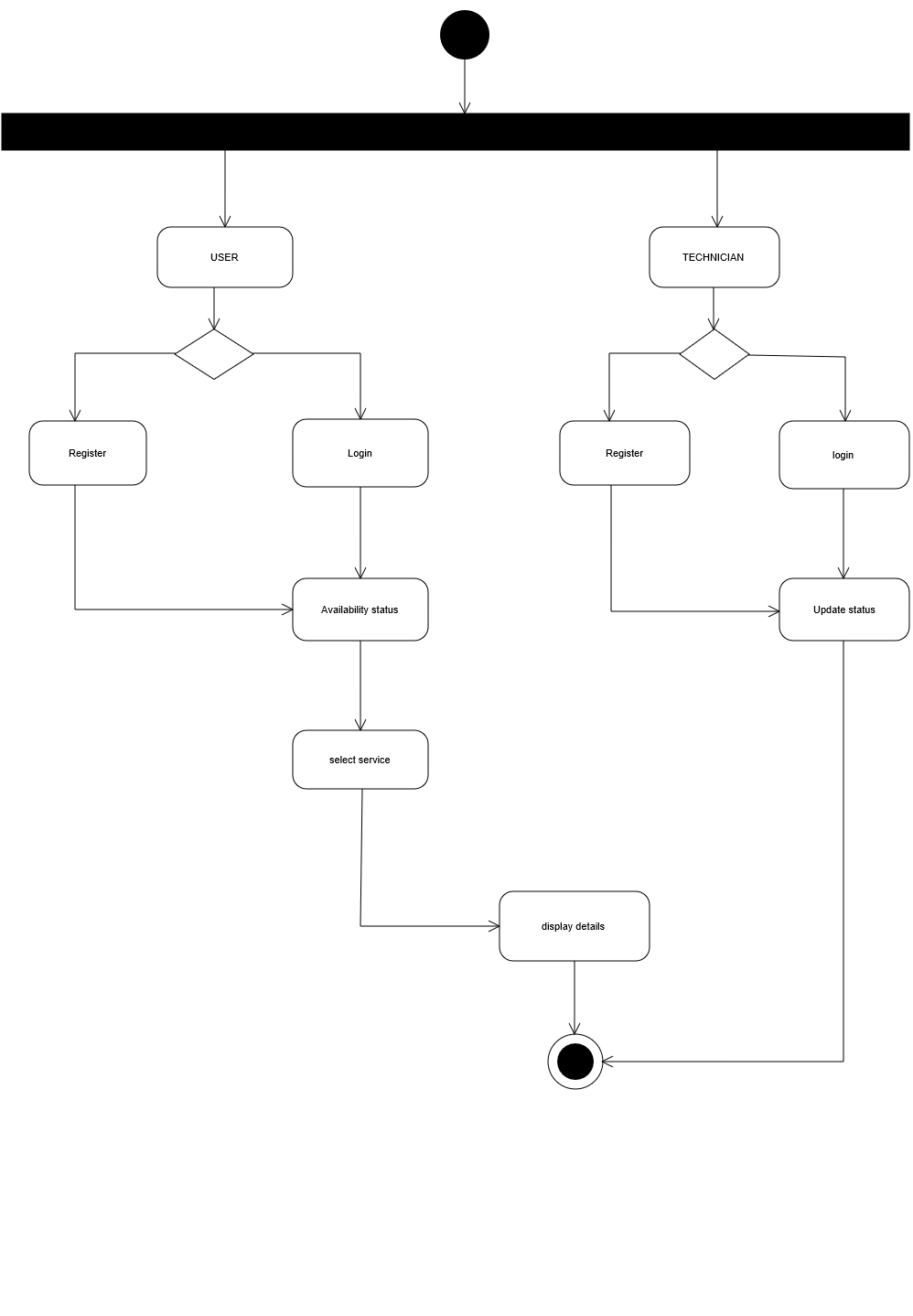


Figure 7.4 Activity Diagram for “Helpers” application

**8. IMPLEMENTATION**

**8.1 Android Technologie**

**8.1.1 Android**

Android is a [mobile operating system](http://en.wikipedia.org/wiki/Mobile_operating_system) initially developed by Android Inc. Android was purchased by [Google](http://en.wikipedia.org/wiki/Google) in [2005](http://en.wikipedia.org/wiki/2005). Android is based upon a modified version of the [Linux kernel](http://en.wikipedia.org/wiki/Linux_kernel). Google and other members of the [Open Handset Alliance](http://en.wikipedia.org/wiki/Open_Handset_Alliance) collaborated to develop and release Android to the world. The Android Open Source Project (AOSP) is tasked with the maintenance and further development of Android. Unit sales for Android OS Smartphone ranked first among all [Smartphone](http://en.wikipedia.org/wiki/Smartphone) OS handsets sold in the U.S. in the second and third quarters of 2010, with a third quarter market share of 43.6%.

Android has a large community of developers writing [application programs](http://en.wikipedia.org/wiki/Application_software) ("apps") that extend the functionality of the devices. There are currently over 100,000 apps available for Android.[Android Market](http://en.wikipedia.org/wiki/Android_Market) is the online app store run by Google, though apps can be downloaded from third party sites (except on AT&T, which disallows this). Developers write in the [Java language](http://en.wikipedia.org/wiki/Java_%28programming_language%29), controlling the device via Google-developed Java libraries.

The unveiling of the Android distribution on 5 November 2007 was announced with the founding of the [Open Handset Alliance](http://en.wikipedia.org/wiki/Open_Handset_Alliance), a consortium of 79 [hardware](http://en.wikipedia.org/wiki/Computer_hardware), [software](http://en.wikipedia.org/wiki/Computer_software), and [telecom](http://en.wikipedia.org/wiki/Telecommunication) companies devoted to advancing [open standards](http://en.wikipedia.org/wiki/Open_standard) for mobile devices. Google released most of the Android code under the [Apache License](http://en.wikipedia.org/wiki/Apache_License), a [free software](http://en.wikipedia.org/wiki/Free_software_license) and [open source license](http://en.wikipedia.org/wiki/Open_source_license).

The Android operating system [software stack](http://en.wikipedia.org/wiki/Software_stack) consists of [Java applications](http://en.wikipedia.org/wiki/Java_%28programming_language%29) running on a Java based [object oriented](http://en.wikipedia.org/wiki/Object-oriented)[application framework](http://en.wikipedia.org/wiki/Application_framework) on top of [Java core libraries](http://en.wikipedia.org/wiki/Java_Class_Library) running on a [Dalvik virtual machine](http://en.wikipedia.org/wiki/Dalvik_%28software%29) featuring [JIT compilation](http://en.wikipedia.org/wiki/Just-in-time_compilation). Libraries written in C include the surface manager, OpenCore[media framework](http://en.wikipedia.org/wiki/Multimedia_framework), [SQLite](http://en.wikipedia.org/wiki/SQLite) relational [database management system](http://en.wikipedia.org/wiki/Relational_database_management_system), [OpenGL ES 2.0](http://en.wikipedia.org/wiki/OpenGL_ES)[3D graphics](http://en.wikipedia.org/wiki/3D_computer_graphics)[API](http://en.wikipedia.org/wiki/Application_programming_interface), [Web Kit layout engine](http://en.wikipedia.org/wiki/WebKit), [SGL](http://en.wikipedia.org/wiki/Skia_Graphics_Engine) graphics engine, [SSL](http://en.wikipedia.org/wiki/Transport_Layer_Security), and [Bionic libc](http://en.wikipedia.org/wiki/GNU_C_Library#Use_in_small_devices). The Android operating system consists of 12 million lines of code including 3 million lines of [XML](http://en.wikipedia.org/wiki/Xml), 2.8 million lines of [C](http://en.wikipedia.org/wiki/C_%28programming_language%29), 2.1 million lines of [Java](http://en.wikipedia.org/wiki/Java_%28programming_language%29), and 1.75 million lines of [C++](http://en.wikipedia.org/wiki/C%2B%2B).

# 8.1.2 The History of Google Android

The fact that hitherto dedicated devices such as mobile phones can now count themselves among the venerable general-computing platforms is great news for programmers. This new trend makes mobile devices accessible through general-purpose computing languages, which increases the range and share for mobile applications. [**The Android Platform**](http://code.google.com/android/)  embraces the idea of general-purpose computing for handheld devices. It is a comprehensive platform that features a Linux-based operating system stack for managing devices, memory, and processes. Android’s libraries cover telephony, video, graphics, UI programming, and a number of other aspects of the device.

The Android SDK supports most of the Java Platform, Standard Edition (Java SE) except for the Abstract Window Toolkit (AWT) and Swing. In place of AWT and Swing, Android SDK has its own extensive modern UI framework. Because you’re programming your applications in Java, you could expect that you need a Java Virtual Machine (JVM) that is responsible for interpreting the runtime Java byte code. A JVM typically provides the necessary optimization to help Java reach performance levels comparable to compiled languages such as C and C++. Android offers its own optimized JVM to run the compiled Java class files in order to counter the handheld device limitations such as memory, processor speed, and power. This virtual machine is called the [**Dalvik virtual machine.**](http://www.dalvikvm.com/)

The familiarity and simplicity of the Java programming language coupled with Android’s extensive class library makes Android a compelling platform to write programs for.

Let us look at how Android arrived on the Mobile OS landscape. Mobile phones use a variety of operating systems such as Symbian OS, Microsoft’s Windows Mobile, Mobile Linux, iPhone operating system (based on Mac OS X), Moblin (from Intel), and many other proprietary operating systems. So far no single operating system has become the de facto standard. The available APIs and environments for developing mobile applications are too restrictive and seem to fall behind when compared to desktop frameworks. This is where Google comes in. The Android platform promised openness, affordability, open source code, and a high-end development framework.

Google [**acquired the startup company Android Inc**](http://www.businessweek.com/technology/content/aug2005/tc20050817_0949_tc024.htm). in 2005 to start the development of the Android Platform The key players at Android Inc. included Andy Rubin, Rich Miner, Nick Sears, and Chris White. In late 2007, a group of industry leaders came together around the Android Platform to form the [**Open Handset Alliance**](http://www.openhandsetalliance.com/) some of the alliance’s prominent members are as follows:

1.Sprint Nextel  
2.T-Mobile  
3.Motorola  
4.Samsung  
5.Sony Ericsson  
6.Toshiba  
7.Vodafone  
8.Google  
9.Intel  
10.Texas Instruments

Part of the alliance’s goal is to innovate rapidly and respond better to consumer needs, and its first key outcome was the Android Platform. Android was designed to serve the needs of mobile operators, handset manufacturers, and application developers. The members have committed to release significant intellectual property through the open source Apache License, Version 2.0.  
The Android SDK was first issued as an “early look” release in November 2007. In September 2008, T-Mobile announced the availability of T-Mobile G1, the first Smartphone based on the Android platform. A few days after that, Google announced the availability of Android SDK Release Candidate 1.0. In October 2008, Google made the source code of the Android platform available under Apache’s open source license. When Android was released, one of its key architectural goals was to allow applications to interact with one another and reuse components from one another. This reuse not only applies to services, but also to data and the user interface (UI). As a result, the Android platform has a number of architectural features that keep this openness a reality.

Android has also attracted an early following because of its fully developed features to exploit the cloud-computing model offered by web resources and to enhance that experience with local data stores on the handset itself. Android’s support for a relational database on the handset also played a part in early adoption.

In late 2008 Google released a handheld device called Android Dev Phone 1 that was capable of running Android applications without being tied to any cell phone provider network. The goal of this device (at an approximate cost of $400.00) was to allow developers to experiment with a real device that could run the Android OS without any contracts. At around the same time, Google also released a bug fix, version 1.1 of the OS that is solely based on version 1.0. In releases 1.0 and 1.1 Android did not support soft keyboards, requiring the devices to carry physical keys. Android fixed this issue by releasing the 1.5 SDK in April 2009, along with a number of other features, such as advanced media-recording capabilities, widgets, and live folders.

## 8.2 Hardware running on Android

The Android OS can be used as an operating system for cellphones, netbooks and tablet PCs, including the [Dell Streak](http://en.wikipedia.org/wiki/Dell_Streak), [Samsung Galaxy Tab](http://en.wikipedia.org/wiki/Samsung_Galaxy_Tab) and other devices.

The world's first TV running Android, called Scandinavia, has also been launched by the company People of Lava.

The first commercially available phone to run the Android operating system was the [HTC Dream](http://en.wikipedia.org/wiki/HTC_Dream), released on 22 October 2008.

## 8.3 Software development

The early feedback on developing applications for the Android platform was mixed. Issues cited include bugs, lack of documentation, inadequate QA infrastructure, and no public issue-tracking system. (Google announced an issue tracker on 18 January 2008.) In December 2007, MergeLab mobile startup founder Adam MacBeth stated, "Functionality is not there, is poorly documented or just doesn't work... It's clearly not ready for prime time." Despite this, Android-targeted applications began to appear the week after the platform was announced. The first publicly available application was the [Snake game](http://en.wikipedia.org/wiki/Snake_%28video_game%29). The [Android Dev Phone](http://en.wikipedia.org/wiki/Android_Dev_Phone) is a [SIM](http://en.wikipedia.org/wiki/Subscriber_Identity_Module)-unlocked and hardware-unlocked device that is designed for advanced developers. While developers can use regular consumer devices purchased at retail to test and use their applications, some developers may choose not to use a retail device, preferring an unlocked or no-contract device.

### 8.3.1 Software development kit

The Android [software development kit](http://en.wikipedia.org/wiki/Software_development_kit) (SDK) includes a comprehensive set of development tools.[[85]](http://en.wikipedia.org/wiki/Android_%28operating_system%29#cite_note-84) These include a [debugger](http://en.wikipedia.org/wiki/Debugger), [libraries](http://en.wikipedia.org/wiki/Software_library), a handset [emulator](http://en.wikipedia.org/wiki/Emulator) (based on [QEMU](http://en.wikipedia.org/wiki/QEMU)), documentation, sample code, and tutorials. Currently supported development platforms include computers running [Linux](http://en.wikipedia.org/wiki/Linux_kernel) (any modern desktop [Linux distribution](http://en.wikipedia.org/wiki/List_of_GNU/Linux_distributions)), [Mac OS X](http://en.wikipedia.org/wiki/Mac_OS_X) 10.4.9 or later, [Windows XP](http://en.wikipedia.org/wiki/Windows_XP) or later. The officially supported [integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) is [Eclipse](http://en.wikipedia.org/wiki/Eclipse_%28software%29) (currently 3.4 or 3.5) using the Android Development Tools (ADT) Plugin, though developers may use any text editor to edit Java and XML files then use [command line](http://en.wikipedia.org/wiki/Command_line) tools ([Java Development Kit](http://en.wikipedia.org/wiki/Java_Development_Kit) and [Apache Ant](http://en.wikipedia.org/wiki/Apache_Ant) are required) to create, build and debug Android applications as well as control attached Android devices (e.g., triggering a reboot, installing software package(s) remotely).

A preview release of the Android SDK was released on 12 November 2007. On 15 July 2008, the Android Developer Challenge Team accidentally sent an email to all entrants in the Android Developer Challenge announcing that a new release of the SDK was available in a "private" download area. The email was intended for winners of the first round of the Android Developer Challenge. The revelation that Google was supplying new SDK releases to somedevelopers and not others (and keeping this arrangement private) led to widely reported frustration within the Android developer community at the time.

On 18 August 2008 the Android 0.9 SDK beta was released. This release provided an updated and extended API, improved development tools and an updated design for the home screen. Detailed instructions for upgrading are available to those already working with an earlier release. On 23 September 2008 the Android 1.0 SDK (Release 1) was released. According to the release notes, it included "mainly bug fixes, although some smaller features were added". It also included several API changes from the 0.9 version. Multiple versions have been released since.

Enhancements to Android's SDK go hand in hand with the overall Android platform development. The SDK also supports older versions of the Android platform in case developers wish to target their applications at older devices. Development tools are downloadable components, so after one has downloaded the latest version and platform, older platforms and tools can also be downloaded for compatibility testing.

Android applications are packaged in [.apk](http://en.wikipedia.org/wiki/APK_%28file_format%29) 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 .dexfiles (compiled byte code files called [Dalvik](http://en.wikipedia.org/wiki/Dalvik_Virtual_Machine) executable), resource files, etc.

**8.4 Sample Code**

package com.example.hp.helpers;

import android.content.Intent;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.support.v4.graphics.drawable.RoundedBitmapDrawable;

import android.support.v4.graphics.drawable.RoundedBitmapDrawableFactory;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.text.Spannable;

import android.text.SpannableString;

import android.view.View;

import android.widget.Button;

import android.widget.ImageButton;

import android.widget.ImageView;

import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

ImageButton user,tech;

TextView tv1,tv2,tv3,tv4,tv5;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

tech=(ImageButton) findViewById(R.id.button);

ImageButton user=(ImageButton)findViewById(R.id.button2);

ImageView im=(ImageView)findViewById(R.id.imageView2);

ImageView im2=(ImageView)findViewById(R.id.imageView3);

Bitmap bitmap= BitmapFactory.decodeResource(getResources(),R.drawable.logo); RoundedBitmapDrawable roundedBitmapDrawable= RoundedBitmapDrawableFactory.create(getResources(),bitmap);

roundedBitmapDrawable.setCircular(true);

im.setImageDrawable(roundedBitmapDrawable);

Bitmap bitmap2= BitmapFactory.decodeResource(getResources(),R.drawable.about); RoundedBitmapDrawable roundedBitmapDrawable2= RoundedBitmapDrawableFactory.create(getResources(),bitmap2);

roundedBitmapDrawable2.setCircular(true);

im2.setImageDrawable(roundedBitmapDrawable2);

im2.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent it2=new Intent(MainActivity.this,Main4Activity.class);

startActivity(it2);

}

});

tv1=(TextView)findViewById(R.id.textView4);

tv1.setText(" CONTACT INFO");

tv1.setTextColor(0xFF000000);

tv2=(TextView)findViewById(R.id.textView5);

tv2.setText("MAIN BRANCH:UPPAL");

tv2.setTextColor(0xFF000000);

tv3=(TextView)findViewById(R.id.textView6);

tv3.setText(" MBL NO:8008748589");

tv3.setTextColor(0xFF000000);

tv4=(TextView)findViewById(R.id.textView28);

tv4.setText("TECHNICIAN");

tv4.setTextColor(0xFF000000);

tv5=(TextView)findViewById(R.id.textView29);

tv5.setText("USER");

tv5.setTextColor(0xFF000000);

tech.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent it=new Intent(MainActivity.this,Main3Activity.class);

startActivity(it);

}

});

user.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent it1=new Intent(MainActivity.this,Main6Activity.class);

startActivity(it1);

}

});

}

}

**9. TESTING**

**9.1 Introduction**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**9.2 Types of Tests**

**9.2.1 Functional Testing**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation and user manuals. Organization and preparation of functional tests is focused on requirements, key functions or special test cases. In addition systematic coverage pertaining to identify Business process flows, data fields, predefined processes and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**9.2.2 Unit Testing**

Unit testing is essentially for the verification of the code produced during the coding phase and the goal is test the internal logic of the module/program. In the Generic code project, the unit testing is done during coding phase of data entry forms whether the functions are working properly or not. In this phase all the drivers are tested they are rightly connected or not. Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two

different phases.

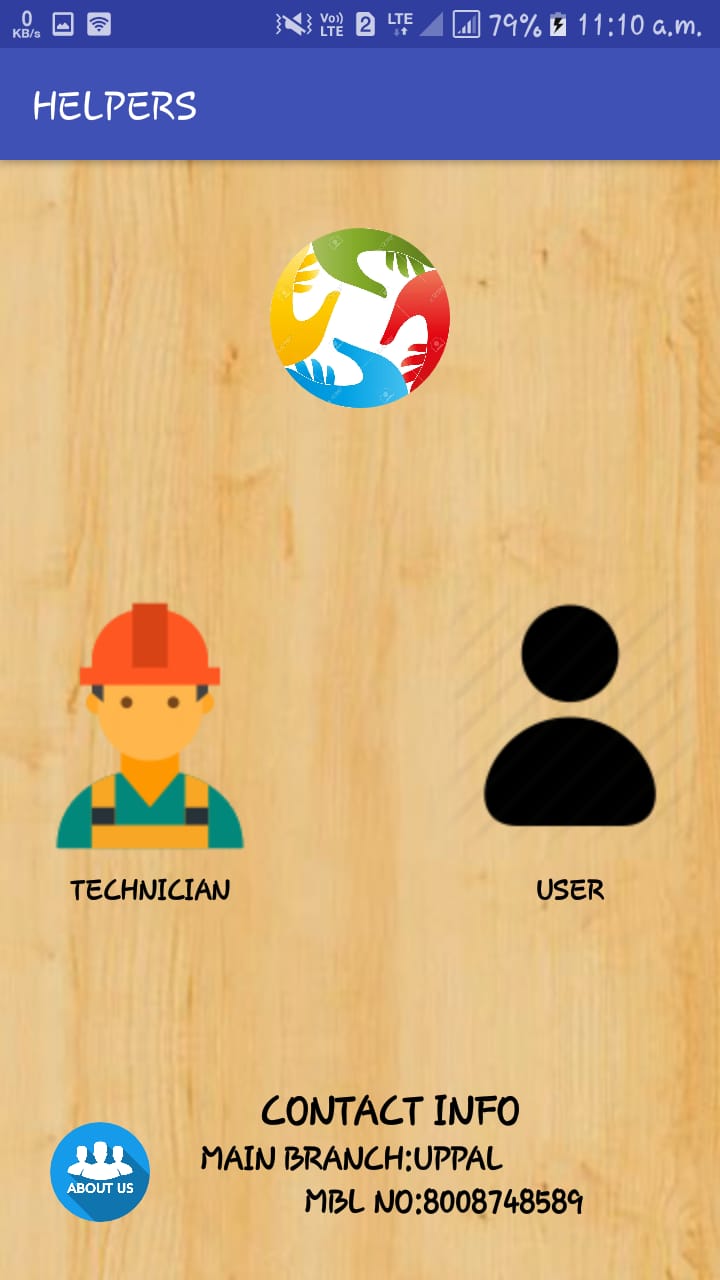
**9.3 Test Cases**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S.NO** | **TEST CASE NAME** | **TEST CASE DESCRIPTION** | **EXPECTED RESULT** | **ACTUAL RESULT** | **TEST CASE STATUS** | **REMARKS** |
| **1.** | Enter Details of technician | Whether details are valid or not | Registered | Registered | Pass | Success |
| **2.** | Login for technician | Verification of login | Update Status page | Updated Status page | Pass | Success |
| **3.** | Enter details of user | Whether details are valid or not | Registered | Registered | Pass | Success |
| **4.** | Login for user | Verification of login | Availability page | Availability page | Pass | Success |
| **5.** | Enter Details of technician | Whether details are valid or not | Registered | Invalid details | Fail | Failure |
| **6.** | Login for technician | Verification of login | Update Status page | Invalid input | Fail | Failure |
| **7.** | Enter details of user | Whether details are valid or not | Registered | Invalid details | Fail | Failure |
| **8.** | Login for user | Verification of login | Availability page | Invalid input | Fail | Failure |

**10. SCREEN SHOTS**

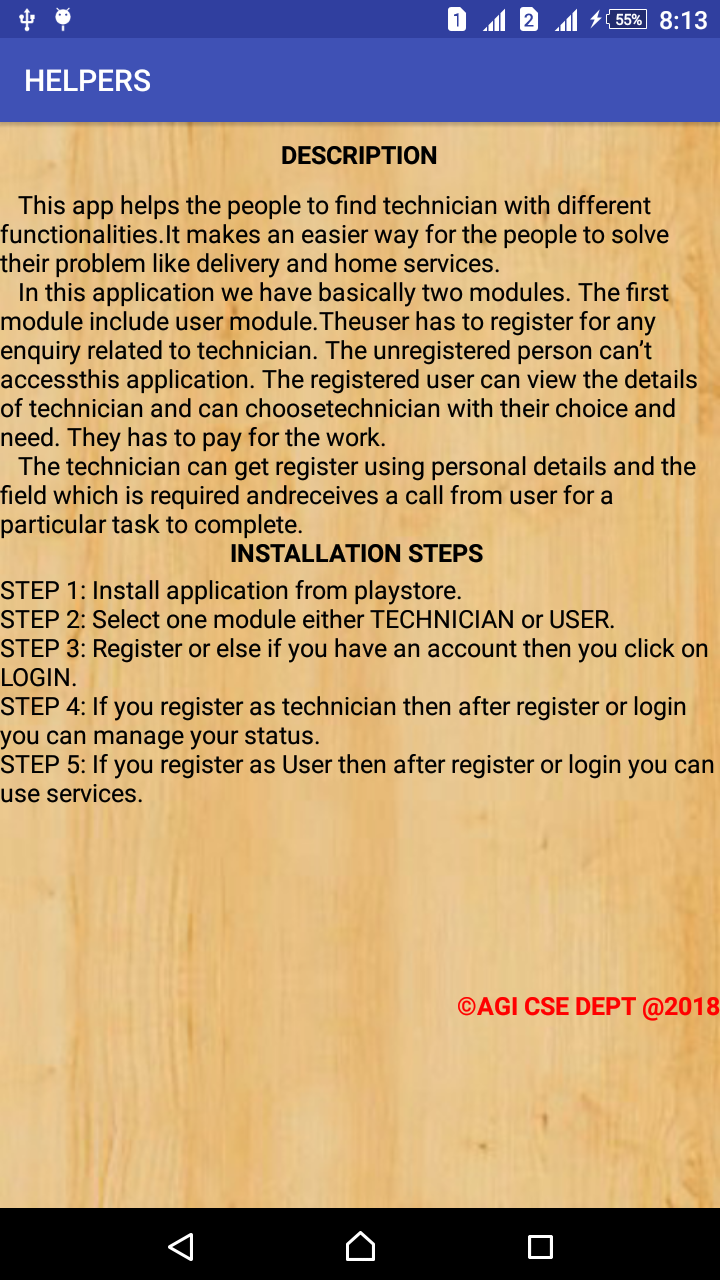
**Screen Shots:**

Figure 10.1 Home page



**Output Screens:**

Figure 10.2 Description



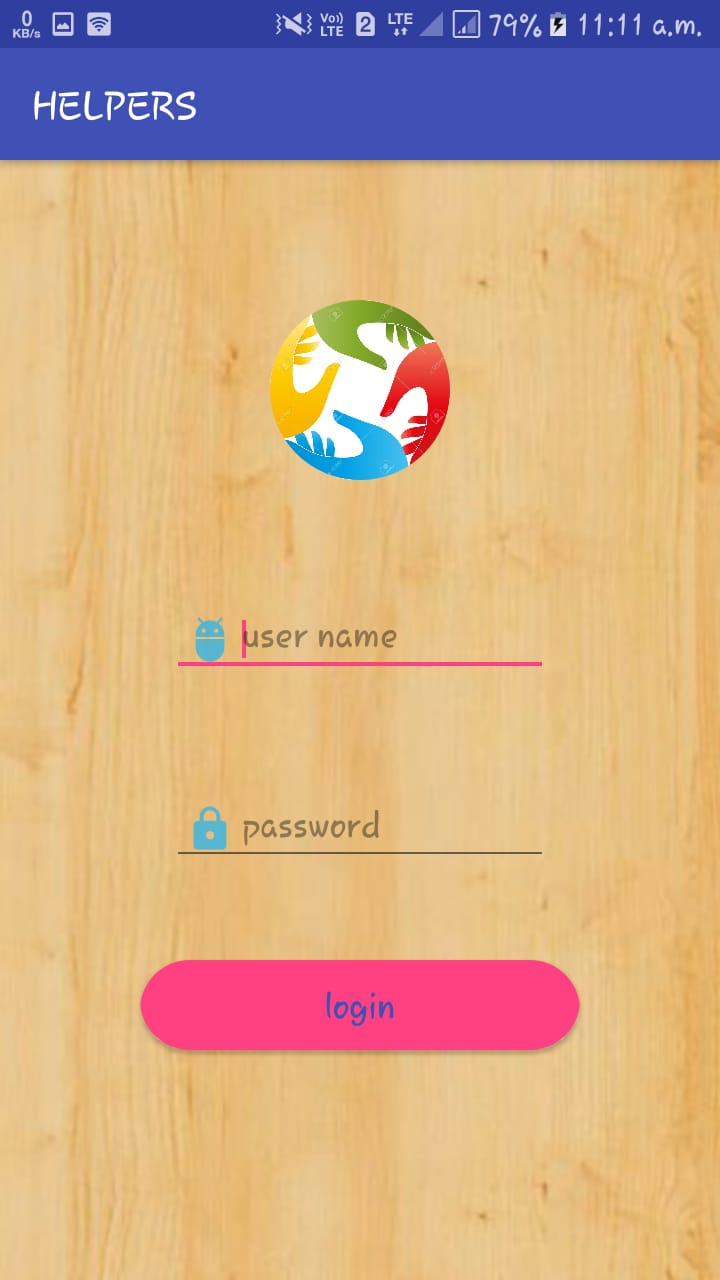
**Output Screens:**

Figure 10.3 Technician registration



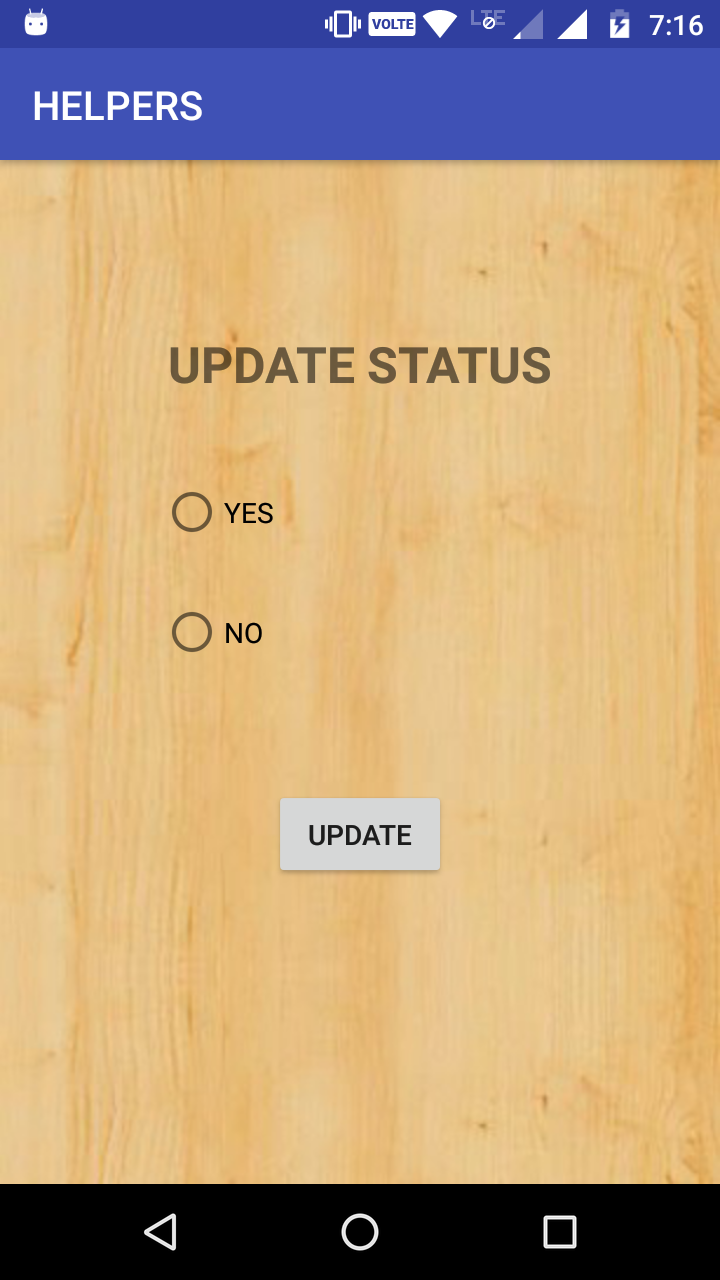
**Output Screens:**

Figure 10.4 Technician Login



**Output screens**:

Figure 10.5 Update Status for Technician



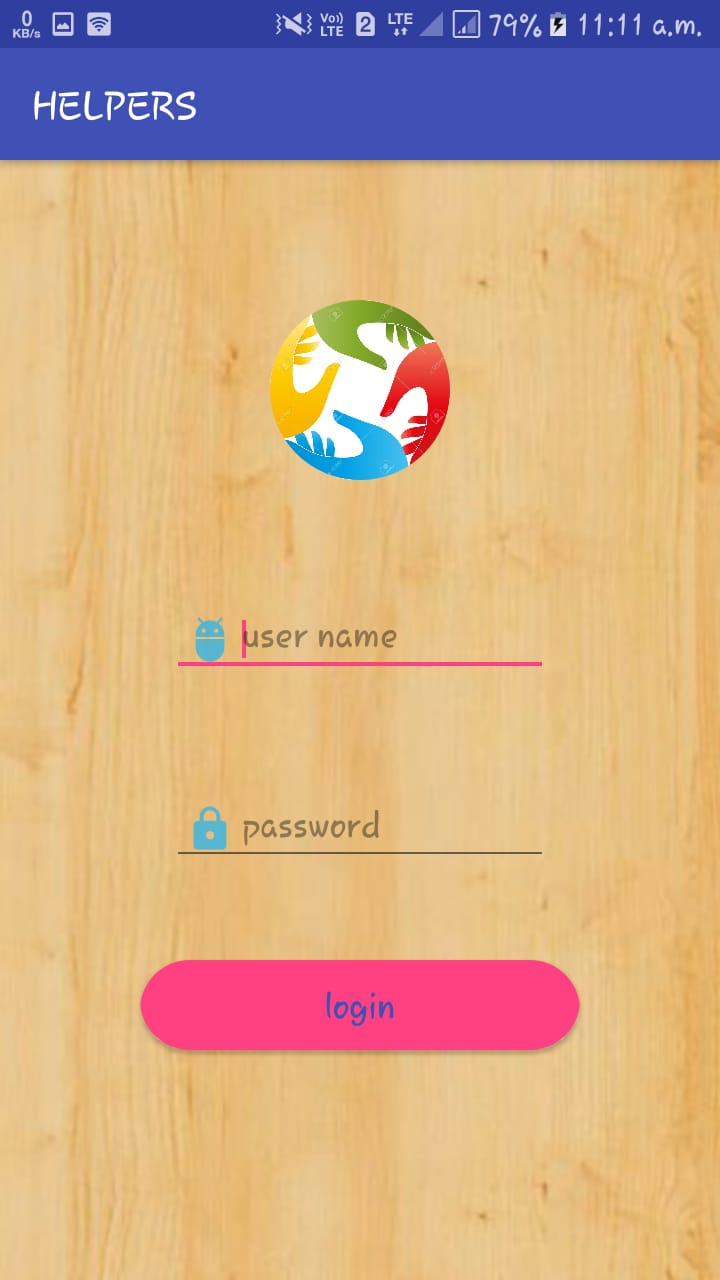
**Output Screens:**

Figure 10.6 User Registration



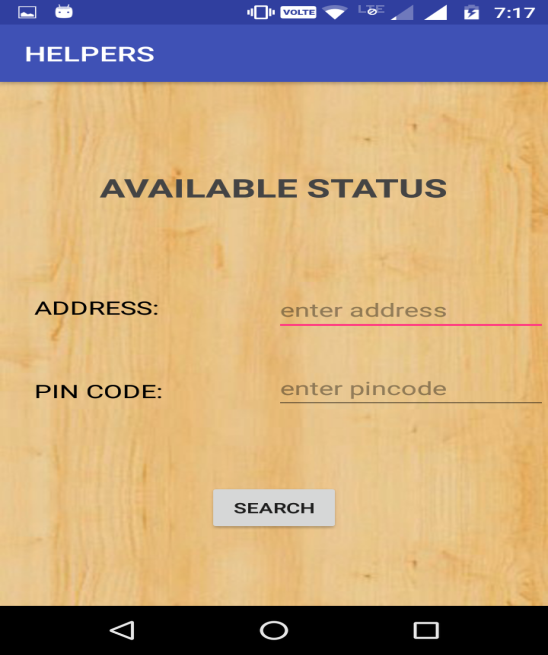
**Output Screens:**

Figure 10.7 User Login



**Output Screens:**

Figure 10.8 Availability Status



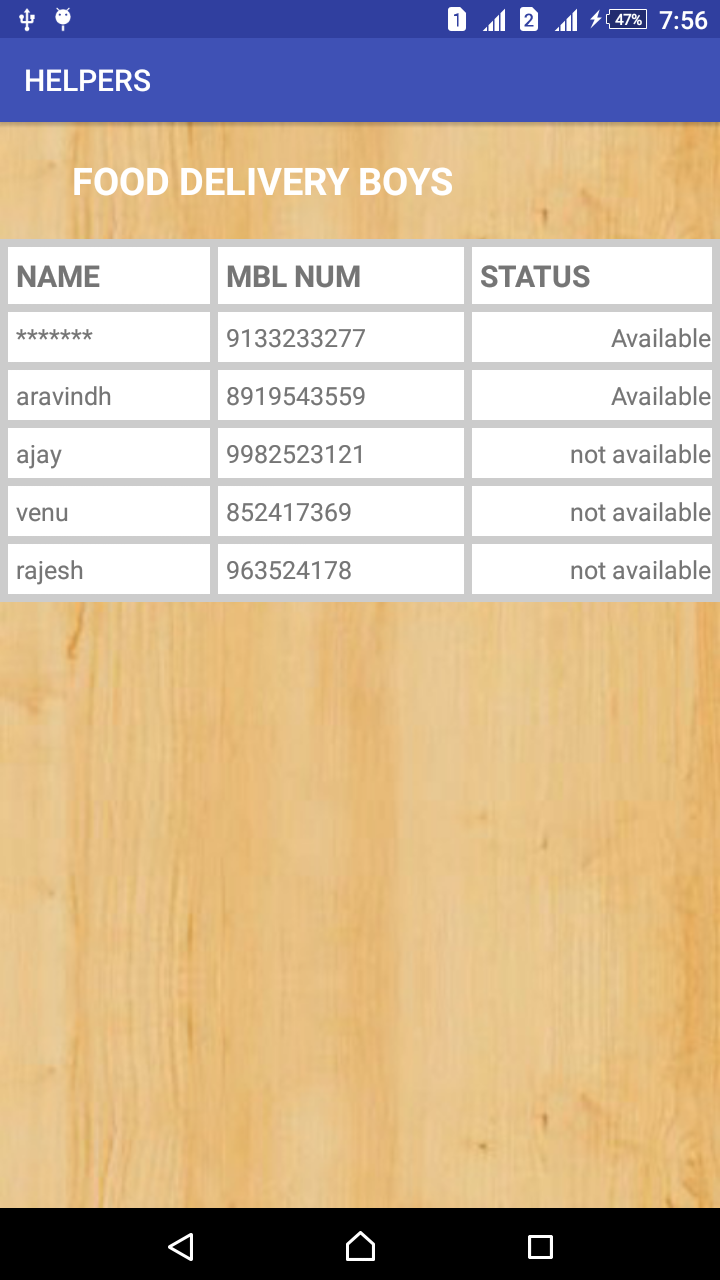
**Output Screens:**

Figure 10.9 Service Page



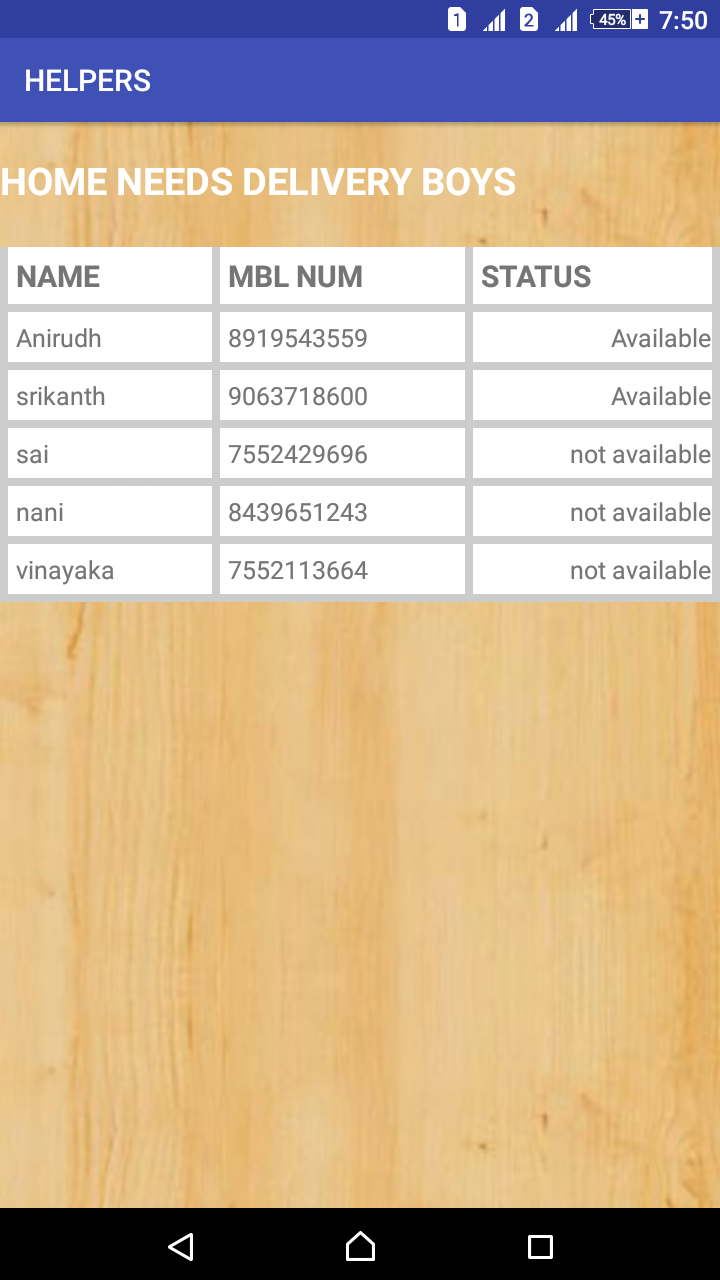
**Output Screens:**

Figure 10.10 Details of food delivery boys



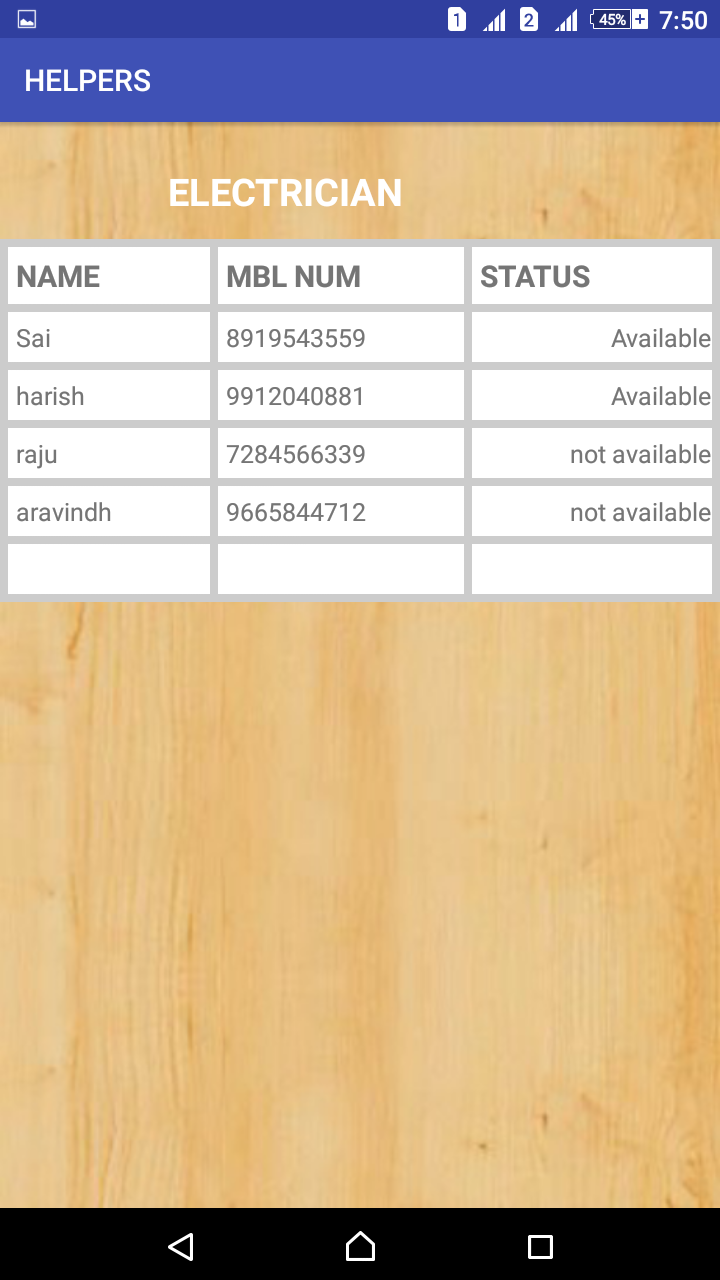
**Output Screens:**

Figure 10.11 Details of home need delivery boys



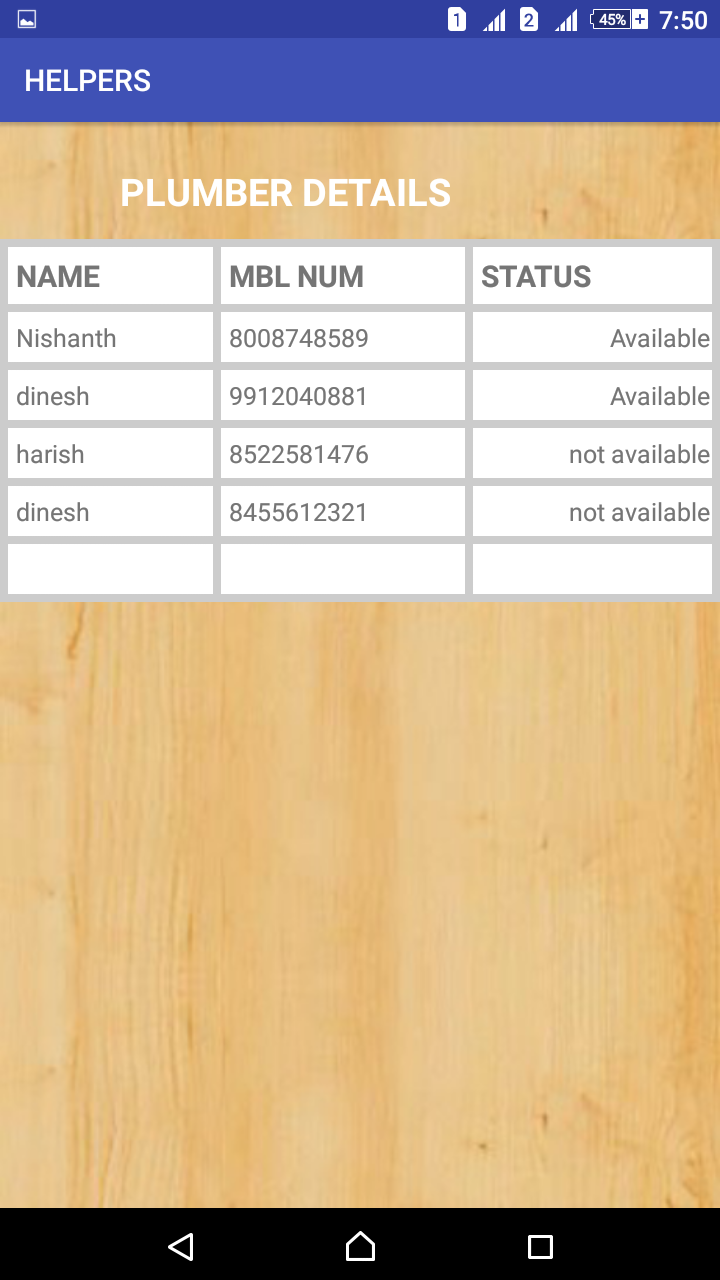
**Output Screens:**

Figure 10.12 Details of electrician



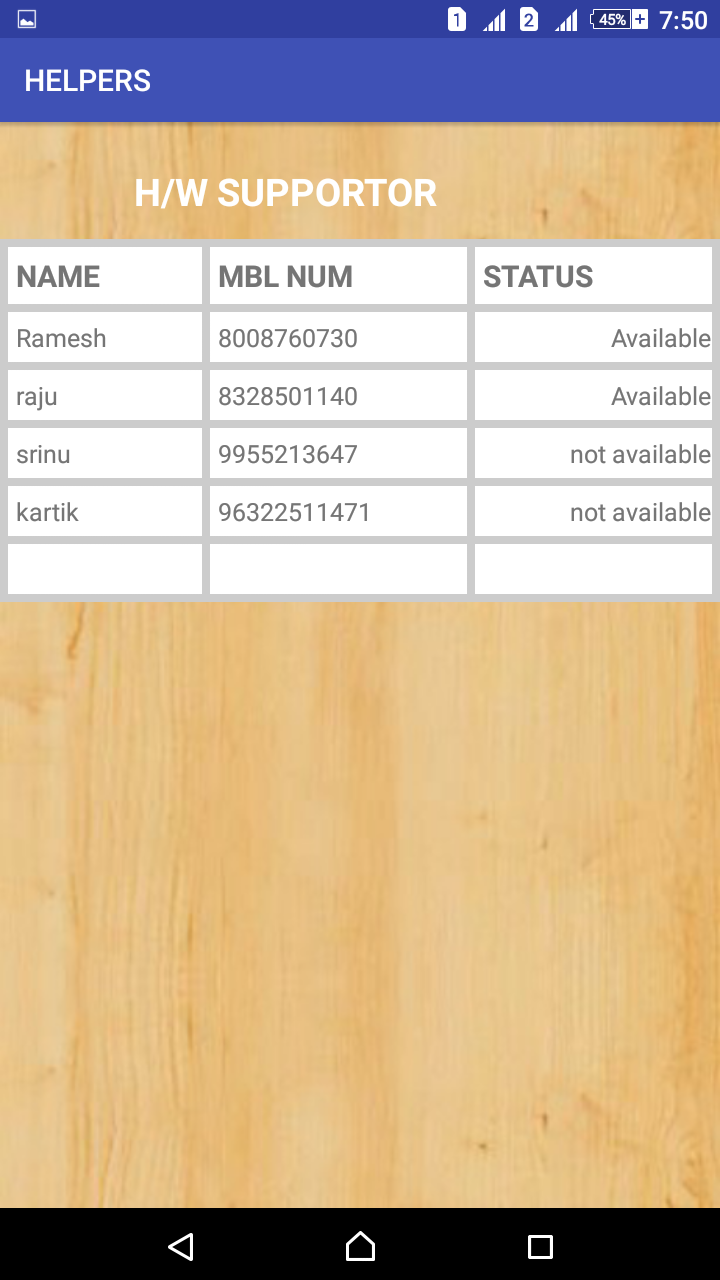
**Output Screens:**

Figure 10.13Details of plumber



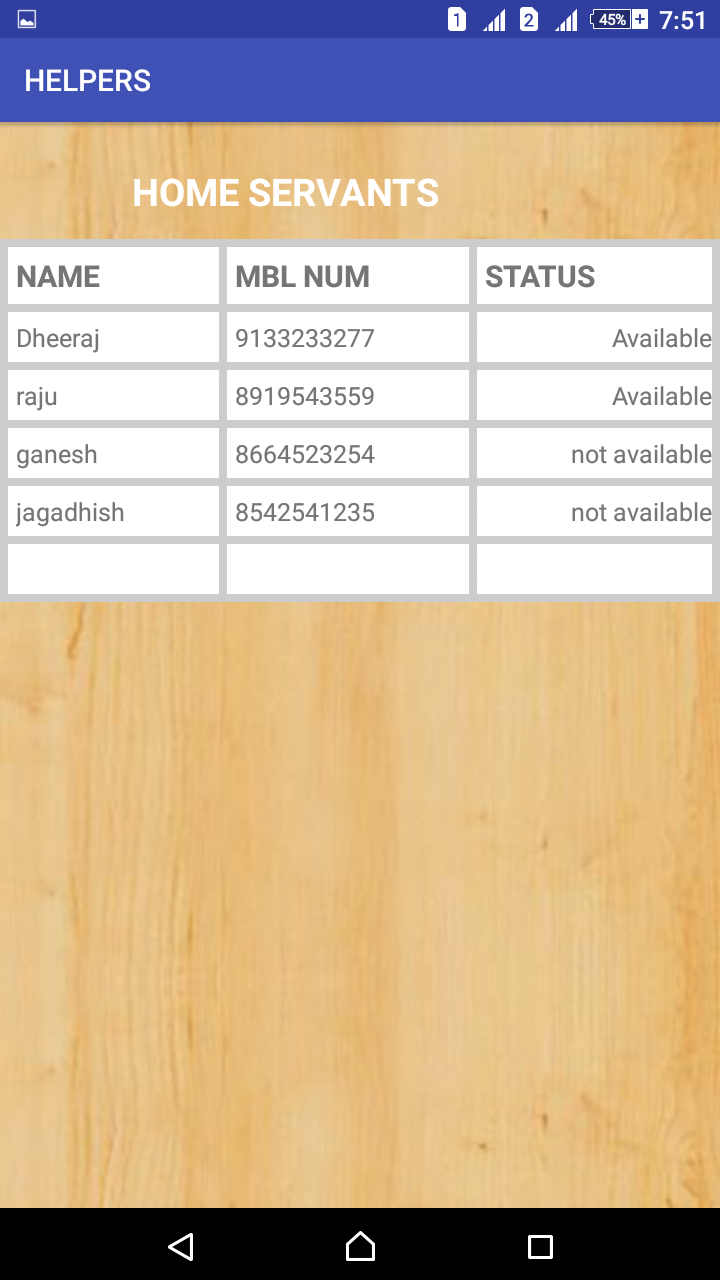
**Output Screens:**

Figure 10.14 Details of hardware supporter



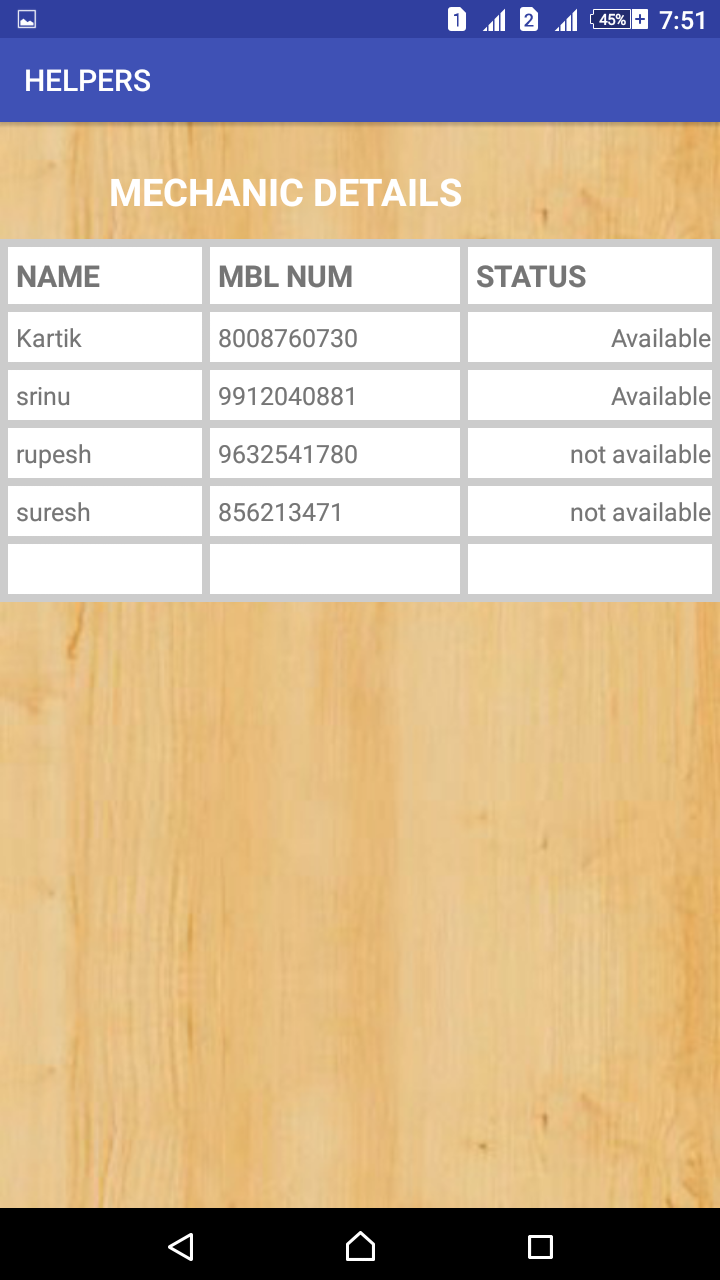
**Output Screens:**

Figure 10.15Details of home servants



**Output Screens:**

Figure 10.16 Details of mechanics



**11. CONCLUSION AND FUTURE ENHANCEMENTS**

From the proper analysis of positive points and the constraints on the android application, it can be safely concluded that the application a highly efficient android-based application for storing, managing, displaying the relevant information in a user-friendly manner.In this android application, the user can access the details of technicians from various genres and can contact them. The system is utilized to maintain and store the details using a firebase. Our further enhancement to the app would be that user is allowed to share their opinion to their friends. We also provide the paytm facility in easier way to transfer the money.