CALL THROUGH WIFI

A PROJECT REPORT

submitted by

SATHISH KUMAR.S	**212211104073**
UDHAYAKUMAR.A	**212211104096**
VIGNESH.M	**212211104100**

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

SAVEETHA ENGINEERING COLLEGE,THANDALAM

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2015

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"CALL THROUGH WIFI"** is the bonafide work of **"SATHISH KUMAR.S (212211104073),UDHAYAKUMAR.A (212211104096)** and **VIGNESH.M (212211104100)"** who carried out the project under my supervision, for the partial fulfilment of the requirements for the award of the Degree of Bachelor of Engineering in Computer Science and Engineering.

SIGNATURE

Mr.R.SARAVANAN, M.E.,(Ph.D.,)

Associate Professor,

HEAD OF THE DEPARTMENT,
Department of Computer Science

and Engineering,

Saveetha Engineering College,

Thandalam,

Chennai-602 105.

SIGNATURE

Ms.A.MANJU M.E., (Ph.D.,)

Assistant Professor(SG),

SUPERVISOR,
Department of Computer Science

and Engineering,

Saveetha Engineering College,

Thandalam,

Chennai-602 105.

Submitted for the Project work and Viva voce Examination held on ____________

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

Our sincere thanks to Founder President Dr***. N. M. VEERAIYAN,*** President Dr***. SAVEETHA RAJESH,*** Director ***Dr. S. RAJESH*** for providing us with the facilities for the completion of our project. We also thank our Principal ***Dr. R. VENKATASAMY,*** B.Sc., M.Tech, Ph.D., and ***Prof. R. DHEENADHAYALU,*** Dean (ICT) for their guidance and encouragement in carrying out our project work.

We would like to express our sincere thanks to ***Mr.R.SARAVANAN,*** M.E, (Ph.D.), Head of Computer Science and Engineering Department for his review of our thesis and his useful opinions.

Our deep appreciation and sincere thanks to our Project Supervisor ***Ms.A.MANJU*** M.E., (Ph.D.), Assistant Professor and Coordinator **Dr. *G.NAGAPPAN,*** M.E., Ph.D., Associate Professor, Department of Computer Science and Engineering for their wisdom, guidance and constant encouragement.

We owe our thanks to all the members of our **College Faculty, Staff and Technicians** for their kind and valuable co-operation during the course of the project. Finally, to our family, for their constant encouragement, support and motivation throughout our under graduate career and for always being there for us.

ABSTRACT

Wi-Fi technology is a form of telecommunication that allows voice data transmissions to be sent across a wide range of interconnected networks. The success in deploying services and applications based on this concept needs the development of applications that allows intermediate users the reception and forwarding of information to destination users. In this work we are going to present the development of an integrated Java application to send data and make a call among user mobile phones through its Wi-Fi interface, without the use of SIM card. A software application is developed using Android platform which allows free and secured communication between selected phones in the Wi-Fi network.

The basic idea is unifying voice and data into a single Wi-Fi network infrastructure by digitizing success of wireless and mobile communications. The mobile communication is carried out in secured manner by sending the packets of data in the encrypted manner. The communication is completely safe from the external attack. This model will be a prototype of different devices communicating through Wi-Fi bandwidth and will reduce the communication cost in large organizations.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	**i**
	LIST OF FIGURES	**v**
	LIST OF TABLES	**vi**
	LIST OF ABBREVIATIONS	**vii**
1	**INTRODUCTION**	**1**
	1.1 AIM	1
	1.2 PROBLEM STATEMENT	1
	1.3 DESCRIPTION	2
	1.4 FEATURES	3
	1.5 VOIP OVERVIEW	4
	1.6 ANDROID OVERVIEW	4
2	**LITERATURE SURVEY**	**5**
	2.1 EXISTING SYSTEM	5
	2.2 PROPOSED SYSTEM	6
3	**SYSTEM ANALYSIS**	**7**
	3.1 FEASIBILITY STUDY	7
	3.1.1 Technical Feasibility	7
	3.1.2 Economical Feasibility	7

CHAPTER NO.	**TITLE**	**PAGE NO.**
3	3.1.3 Operational Feasibility	8
	3.2 HARDWARE USED	9
	3.3 SOFTWARE USED	9
	3.4 SPECIFIC REQUIREMENTS	9
	3.4.1External Interface Requirements	9
	3.4.1.1 User Interfaces	9
	3.4.1.2 Software Interfaces	10
	3.4.2 Performance Requirements	10
4	**DETAILED DESIGN**	**11**
	4.1 SYSTEM ARCHITECTURE	11
	4.1.1 MODULES	12
	4.2 FLOW DIAGRAM	14
	4.3 UML DIAGRAMS	15
	4.3.1 Use Case Diagram	15
	4.3.2 Sequence Diagram	16
	4.3.3 Activity Diagram	17
5	**IMPLEMENTATION AND TESTING**	**18**
	5.1 IMPLEMENTATION	18
	5.1.1Installing an application	19
	5.1.2 Making a call	20

CHAPTER NO.	**TITLE**	**PAGE NO.**
5.	5.2 TESTING	20
	5.2.1 Sign Up Testing	21
	5.2.2 Login Testing	21
	5.2.3 Making call Testing	22
	5.3 TEST PLAN	23
	5.4 TEST ANALYSIS	23
	5.5 RESULT	23
6	**CONCLUSION AND FUTURE ENHANCEMENT**	**24**
	6.1 CONCLUSION	24
	6.2 FUTURE ENHANCEMENT	24
	APPENDIX- A SAMPLE SOURCE CODE	**25**
	APPENDIX -B SCREEN SHOTS	**49**
	REFERENCES	**58**

LIST OF FIGURES

FIGURE NO.	CAPTION	PAGE NO.
3.1	FEASIBILITY ANALYSIS	8
4.1	SYSTEM ARCHITECTURE	11
4.2	FLOW DIAGRAM	14
4.3	USE CASE DIAGRAM	15
4.4	SEQUENCE DIAGRAM	16
4.5	ACTIVITY DIAGRAM	17
5.1	PROCESS OF TESTING	20
b.1	LOGIN PAGE	49
b.2	REGISTRATION	50
b.3	ENTERING THE NAME	51
b.4	DISPLAYING THE USER LIST	52
b.5	MAIN PAGE	53
b.6	CONNECTING TO USER	54
b.7	TURNING AP ON	55
b.8	USER CONNECTED	56
b.9	TRANSMISSION	57

LIST OF TABLES

CHAPTER NO	TITLE	PAGE NO.
5.2	SIGN UP	24
5.3	LOGIN	24
5.4	MAKING A CALL	25

LIST OF ABBREVIATIONS

JDK	**-**	JAVA DEVELOPMENT KIT
UMA	**-**	UNLICENSED MOBILE ACCESS
WLAN	**-**	WIRELESS LAN
WI-FI	**-**	WIRELESS FIDELITY
UDP	-	USER DATAGRAM PROTOCOL
SIP	-	SESSION INITIATION PROTOCOL
VOIP	**-**	VOICE OVER INTERNET PROTOCOL
IP	-	INTERNET PROTOCOL
TCP	**-**	TRANSITION CONTROL PROTOCOL

CHAPTER 1
INTRODUCTION

1.1 AIM

The scope of this project is to make calls in single Wi-Fi network, without using SIM card. The basic idea is unifying voice and data onto a single network infrastructure and communication is carried out between the user available in single Wi-Fi network. The aim is to build an android application that allows free and secured communication between selected phones in the Wi-Fi network.

1.2 PROBLEM STATEMENT

Voice over Internet Protocol (VoIP) provides the ideas for connecting two clients through voice over the internet. Although IP phone communication over the data network such as LAN exists but these IP phone are fixed type. Many top MNC companies use the Intranet communication to share the data, which is expensive to setup (uses the VoIP). This application aims at developing a system that eliminates the drawback of existing systems. We implement wireless IP phone communication using the Wi-Fi network. This application can be used in mobile device that is Wi-Fi enable to communicate with a router. In this communication, voice data is transmitted from one user to another in the encrypted format. Then this is packetized and forwarded to router through Wi-Fi channel. The router performs header decryption and finds the destination phone from it routing table.

1.3 DESCRIPTION

In the voice calls over Wi-Fi, the requirements are:

- Mobile phones that are Wi-Fi enabled.
- Wireless router is used to send the data to particular user.

- Routing table maintain the router and Wi-Fi connection.

Voice Calls over Wi-Fi involves making a free call within the Wi-Fi range. The sender and receiver should have a Wi-Fi enabled mobile. The mobile phones should be connected to Wi-Fi range. Senders make a call and receiver receive the call within the Wi-Fi range.

Our approach adopts wireless communication and mobile phones instead of using pc. In existing model, call was initiatiated with fixed IP address. The advantage of this method is that, using mobile phones for making a call through Wi-Fi network, and it can be implemented over many organizations and operate companies. The proposed model is developed using the Android Studio, which is an android application development tool. In this model the users is provided with a Username which can be identified in user available list. Once the user initiated the call, it search for destination user in network, and wait for the another user to connect to the same network. Once both users connect to the network the communication can be carried out in the encrypted format.

The Voice data is transmitted over the Wi-Fi network in packetizised form, which is secured from the external attack. This can be improved to share the data among the user over the Wi-Fi network and can be implemented in the many co-operate companies, which can replace the intranet and intercom connections.

1.3.1 FEATURES

The features that are available to the user are:

- Can communicate to the users in same network.
- Can replace the Intranet and Intercom connection in the co-operate companies and organizations.
- Can be communicate with users anywhere within the Wi-Fi range (portable).

1.3.2 VoIP TECHNOLOGY OVERVIEW

VoIP refers to the movement of voice traffic over internet protocol IP Based network. To permit the traffic over the computer network, the analog signals are turned into digital packets. The digital packets have a destination address but they follow no fixed path. To enable VoIP , broadband access, computer, and software. Additional hardware such as server ,switches, router, and other may be required depending on the volume and nature if traffic. VoIP permits the integration of data, voice into one communication channel.
The term digital convergence refers to this phenomenon of multiple media delivered over a single network. Some of the application and service include PC based distance learning solutions, video conferencing and team management software. Early providers of voice over IP services offered business models (and technical solutions) that mirrored the architecture of the legacy telephone network.

1.3.3 ANDROID OVERVIEW

Android is software for mobile devices that includes an operating system middleware and key applications. The Android SDK provides the tool and APIs necessary to begin developing application on the android platform using the java programming language.

Android will ship with a set of core application including an email client, SMS program, calendar, maps, browser, contact, and other. All applications are written using the java programming language.

Android include a set of C/C++ libraries used by the various component of the Android System. These capabilities are exposed to developers through the android application framework.

Some of the core libraries are listed below:

- **System C Library –** a BSD-derived implementation of the standard C system libraries (libc), tuned for embedded Linux-based devices.
- **Media Libraries -** based on the packet video Open COR, the libraries support playback and recording of many popular audio and video formats, as well as static images files, including MPEG4, H.264,MP3,AAC,AMR,JPG and PNG.
- **Surface Manager -** manages to access the display the subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING SYSTEM

2.1.1 WI Call (VoIP)

- WI call allows you to make and receive calls within the wireless network. It access unlimited high-speed data to send and receive messages over a wireless connection. WI call is very simple to connect an available Wi-Fi network of your choice
- Voice over Internet Protocol (VoIP) provides the ideas for connecting two client through voice over the internet. The advent of voice over Internet Protocol (VoIP) had fundamentally been transforming the way telecommunication evolves.
- Driven by the ongoing deployment of broadband infrastructure and the increasing demand of telecommunication service, VoIP technologies and applications have led to the development of economical IP phone equipment based on embedded systems.
- IP phone application can satisfyingly provide the necessary interfaces between telephony signals and IP networks. Although IP phone communication over the data networks such as LAN exists but these IP phones are fixed type. We implement wireless IP phone communication using the Wi-Fi network, VoIP phones call without the use of a computer, instead they connect directly to the IP network(using technologies such as Wi-Fi router).

DISADVANTAGES

- Vulnerable to attacks.
- Vulnerable to hack an important data.
- Not cost effective mechanisms.
- Several issues for security threats.
- Security enable communication channels are costly.

2.2 PROPOSED SYSTEM

- Our application tries to overcome these difficulties by providing a dedicated platform to transmit the data in secure and cost efficiently. We implement wireless IP phone communication using the Wi-Fi network, within the same Wi-Fi region.
- This proposal allows free calls within the network with high quality voice transmission, and allows mobile phones to communicate with other in the same Wi-Fi enabled region.
- This model will be a prototype of inherent devices communicating through the Wi-Fi bandwidth and greatly reduce the communication cost in large organisation.

ADVANTAGES

- Communication is carried out between users in same network.
- Replace the Intranet and Intercom connection in the co-operate companies and organizations.
- Portability.

CHAPTER 3

SYSTEM ANALYSIS

3.1 FEASIBILITY STUDY

The main objective of feasibility study is to test the Technical, operational and economical for adding new modules and debugging old running system. All system is feasible for adding new modules and debugging old running system.

There are aspects in the feasibility study portion of the preliminary investigation:

- Technical feasibility
- Economic feasibility
- Operational feasibility

3.1.1 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. To develop this system, we first worked with web applications and then found that Android would be technically feasible. The requirements here are very modest because the system here supports very basic android version which is 3.2.1 and above. So, this system is technically feasible because it needs only android with a basic version.

3.1.2 ECONOMICAL FEASIBILITY

Economic feasibility is the most frequently used method for evaluating the effectiveness of the proposed system. The System is cost effective because it is freely available as android application and can be downloaded at free of cost from the internet and installs the app on their phones and can run it.

3.1.3 OPERATIONAL FEASIBILITY

The aspect of the study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. This system will not threaten the user instead it is friendly in its operation. All the user needs to have is an Wi-Fi connection provided to his mobile.

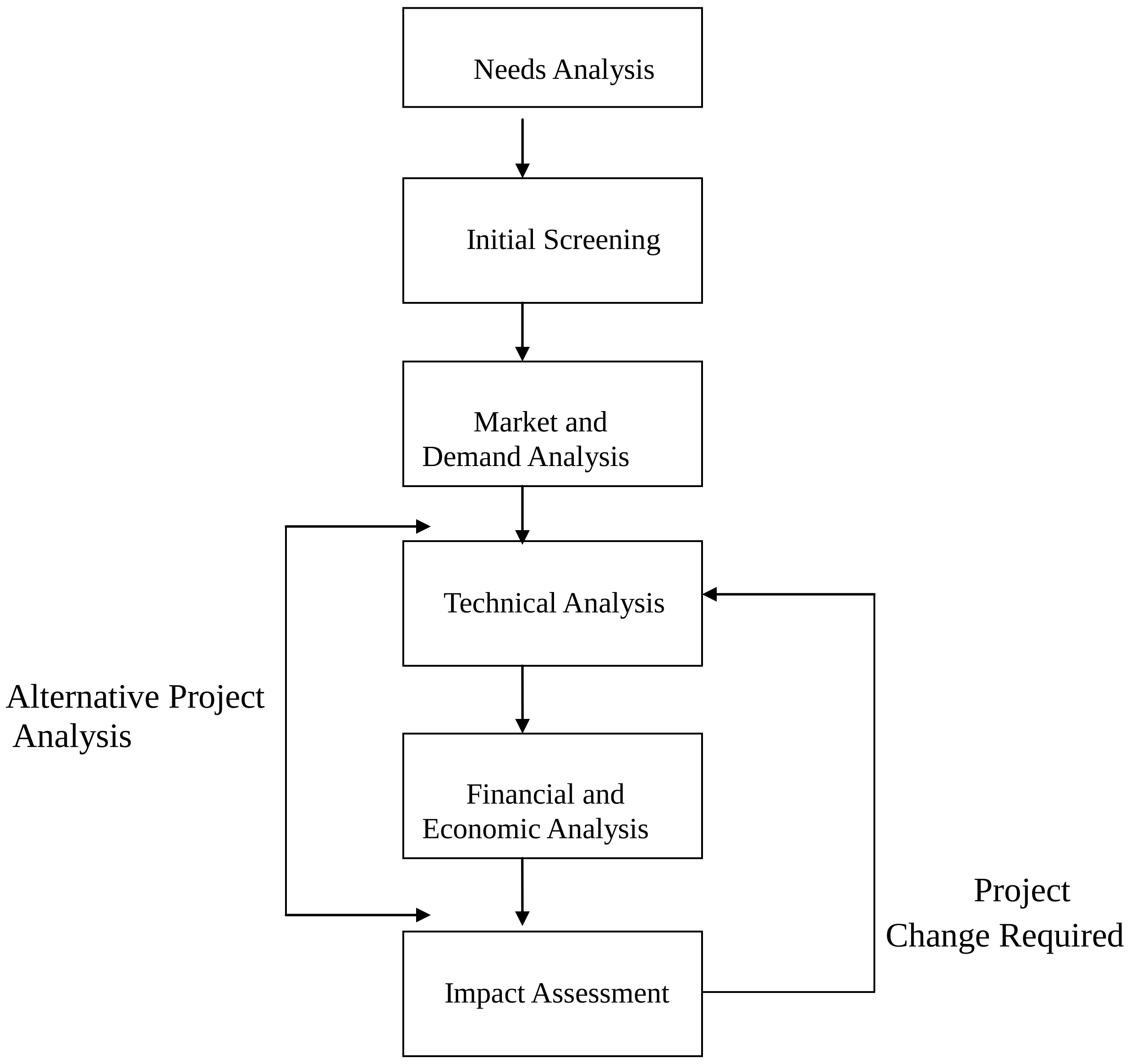


Fig. 3.1 Feasibility Analysis

3.2 HARDWARE USED

1. Wi-Fi enabled Android Mobile Phone.
2. Android Mobile Phones with Hotspot.

3.3 SOFTWARE USED

1. *Windows OS*- Computer operating system (OS) developed by Microsoft Corporation to run personal computers

2. *JDK 1.8.0_31*- The JDK is a superset of the JRE, and contains everything that is in the JRE, plus tools such as the compilers and debuggers

3. *Android Studio 1.0.1*- It is an Android application development tools to develop the android applications

4. *Android SDK 3.2.1*- It is an platform where the android application can be implemented by using the android simulator.

5. *Sqlite* - It is an local Database where the data is stored in the form of tables.

3.4 SPECIFIC REQUIREMENTS

It mainly analyzes the user requirements and gives criteria to acquire the requirement.

3.4.1 EXTERNAL INTERFACE REQUIRMENTS

3.4.1.1 User Interfaces

1. All the contents in the project are implemented using Graphical User Interface (GUI) – Android SDK.

2. Every conceptual part of the projects is reflected using the Android SDK.

3. System gets the input and delivers through the Android’s GUI based.

3.4.1.2 Software Interfaces

1.Using xml we have created front end tool design and java for logical operation.

2.Sqlite database is used for back end to store the user and retrieve in the list view.

3.4.1.3 PERFORMANCE REQUIREMENTS

In our proposed model the performance can be improved by giving the Wi-Fi signal strength. The routers also change the performance of the project. The system performance and system configuration also increase the performance of the project.

CHAPTER 4

DETAILED DESIGN

4.1 SYSTEM ARCHITECTURE FOR CALL THOUGH Wi-Fi

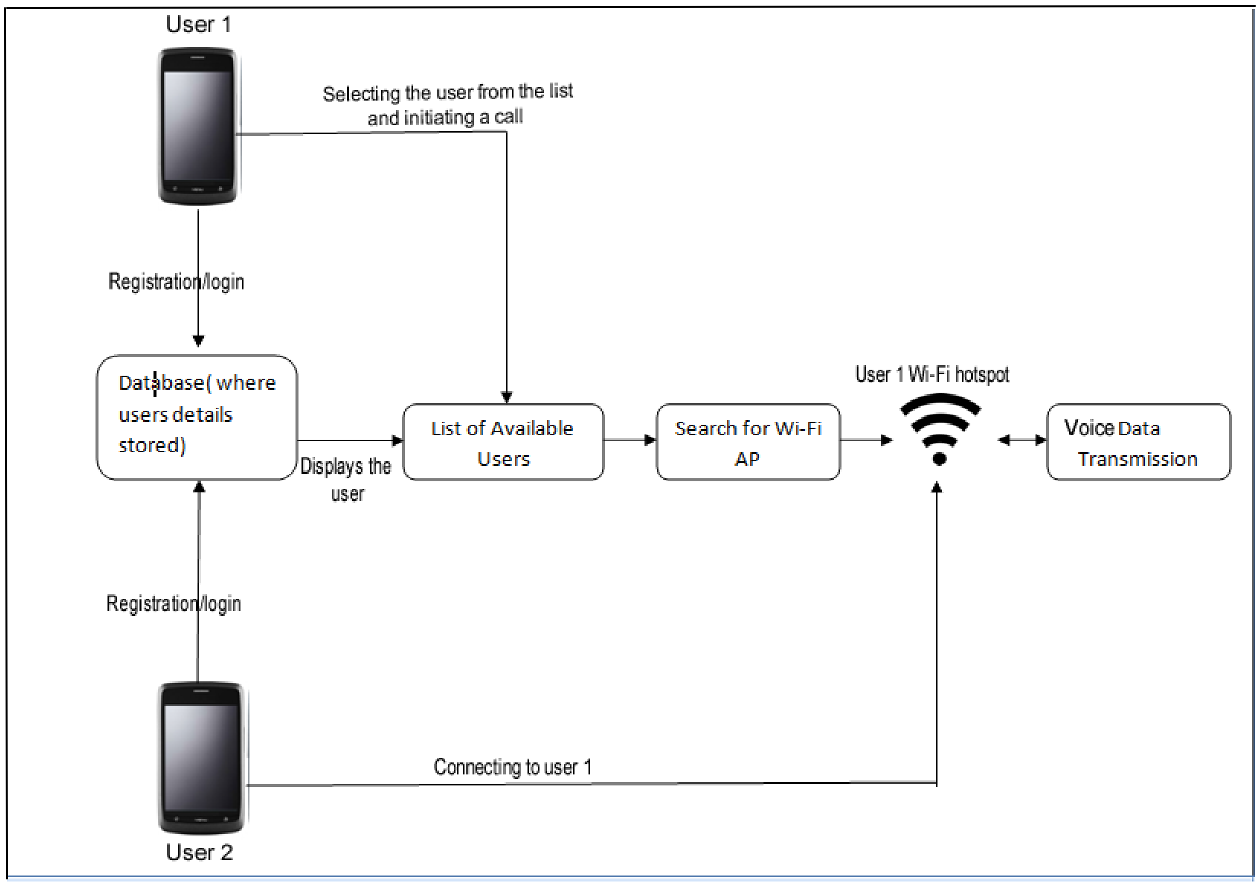


Fig 4.1:System Architecture

In the architecture, once user login to application list of available users are displayed, and then to communicate with the particular user, the call must be initiated and both the users connected to the router. Anyone in available user list can intimate the call, router send the data to the particular receiver and communication is carried out through the network within the range of the Wi-Fi router. If the user goes out of the Wi-Fi range then calls get disconnected.

4.1.1 MODULES

1. Signing up

Creating a User account is the first step for anyone to access the application. New users should choose the signup option in the launcher page and provide the necessary details and register into application.

2. Login Screen

Once the user is registered they can use the username and password to log into the application, view list of available users to communicate over the network.

3. Entering the name

Once the valid user enters into application, it ask for name to be identified in the list of users available in Network. After entering the name it redirect to the list of available users in the Wi-Fi network.

4. Database

The Database would contain the list of users available in networks. The database would be an online entity like cloud storage, from which the data can be shared to other people through the internet. The details stored as, username and password for an authentication. The details stored in the database are retrieved in the form of list for communication.

5. List of user (status)

Once the user enters the name it displays the user available in the Wi-Fi network. There is a status field which determines what the current activity taking place in the application. The any user in the list can initiate the call to communicate with any user in the list.

6. Make a call

Once user select the users from the list, it redirect to the next screen which user can make a call to communicate with the users. It display Connect , Make a Call and Exit button to initiate the call for communication and to get connected to available user in respective Wi-Fi router. The user should connect the both devices to transmit the voice data. Once the user (sender) clicks the Connect button it search for the connections in the Wi-Fi network. If the network not found it will initiate a access point and wait for an destination user to get connected. Once both users get connected, they can start transmitting the voice data by selecting Make a call button.(The xml for layout designs and java for programming language for functionalities.)

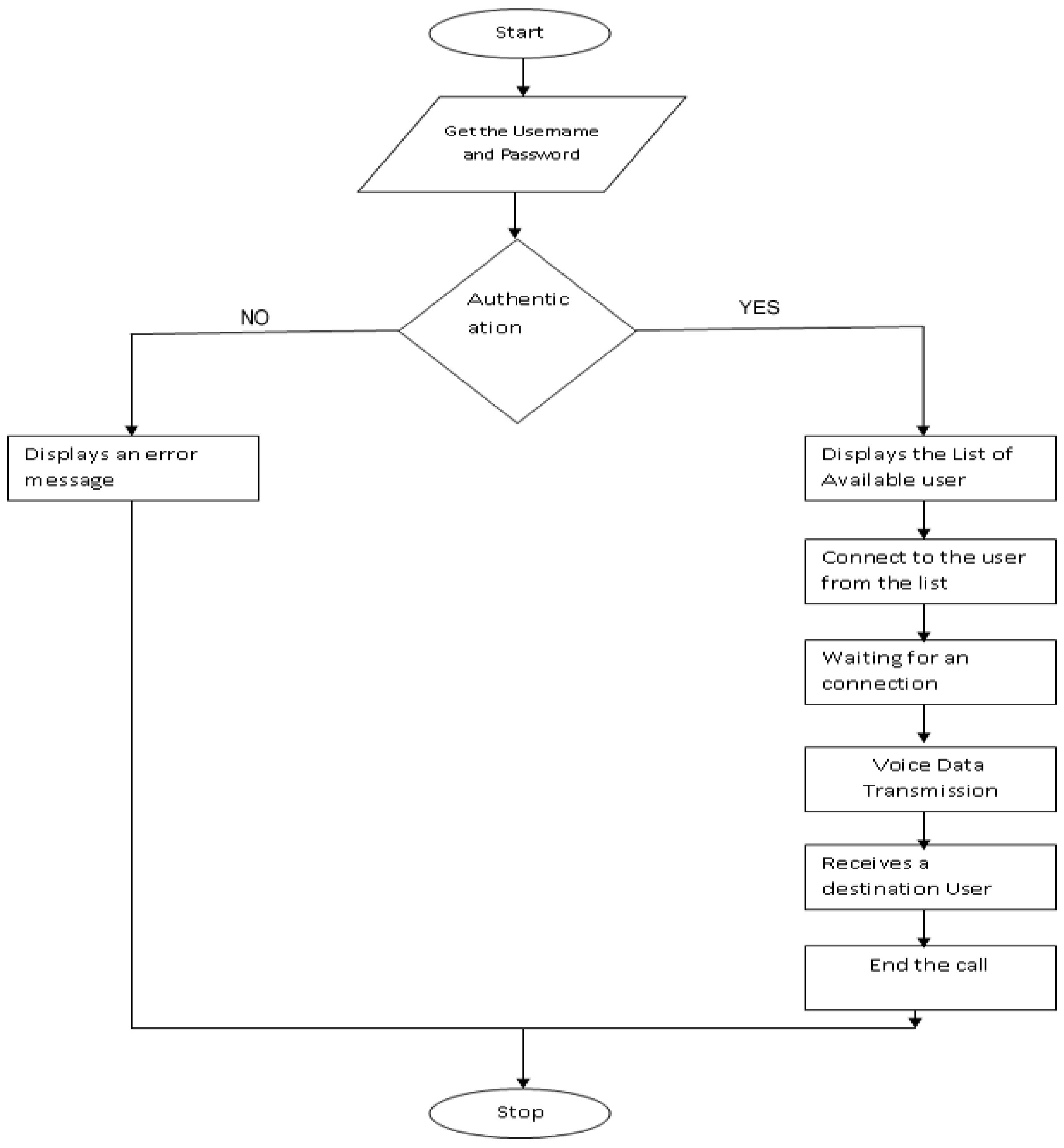
7. Transmission

The user can transmit the voice data from the user list with no cost and time limits. Once the connection is setup, the user clicks a button (Make a Call) to communicate with destination user. The voice data is transmitted in the encrypted format between the users. The quality depends upon the Wi-Fi signal strength.

8. Exit

To close the Wi-Fi call application, the user should select the exit option from the main layout.

4.2 FLOW DIAGRAM



Start
Get the Username and Password
Authentication
NO
YES
Displays an error message
Displays the List of Available user
Connect to the user from the list
Waiting for an connection
Voice Data Transmission
Receives a destination User
End the call
Stop

Fig: 4.2 Flow chart

4.3 UML DIAGRAMS

4.3.1 USE CASE DIAGRAM

Use case describe the interaction between one or more actors and the system itself, represented as a sequence of simple steps that take part in a sequence of activities in a dialog with the system to achieve goal.

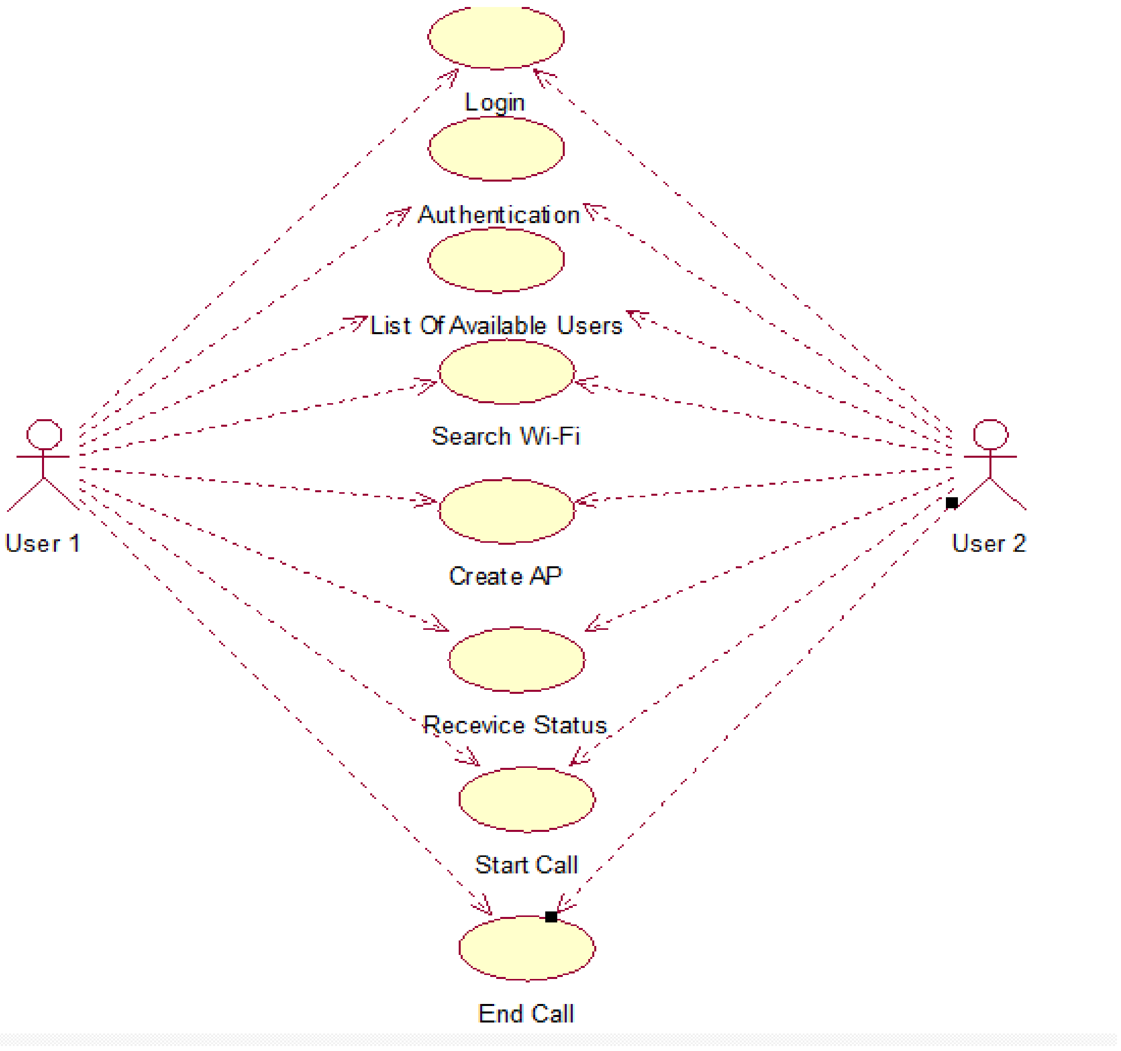


Fig 4.3: Use Case Diagram

4.3.2 SEQUENCE DIAGRAM

A Sequence diagram shows, as parallel vertical lines different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

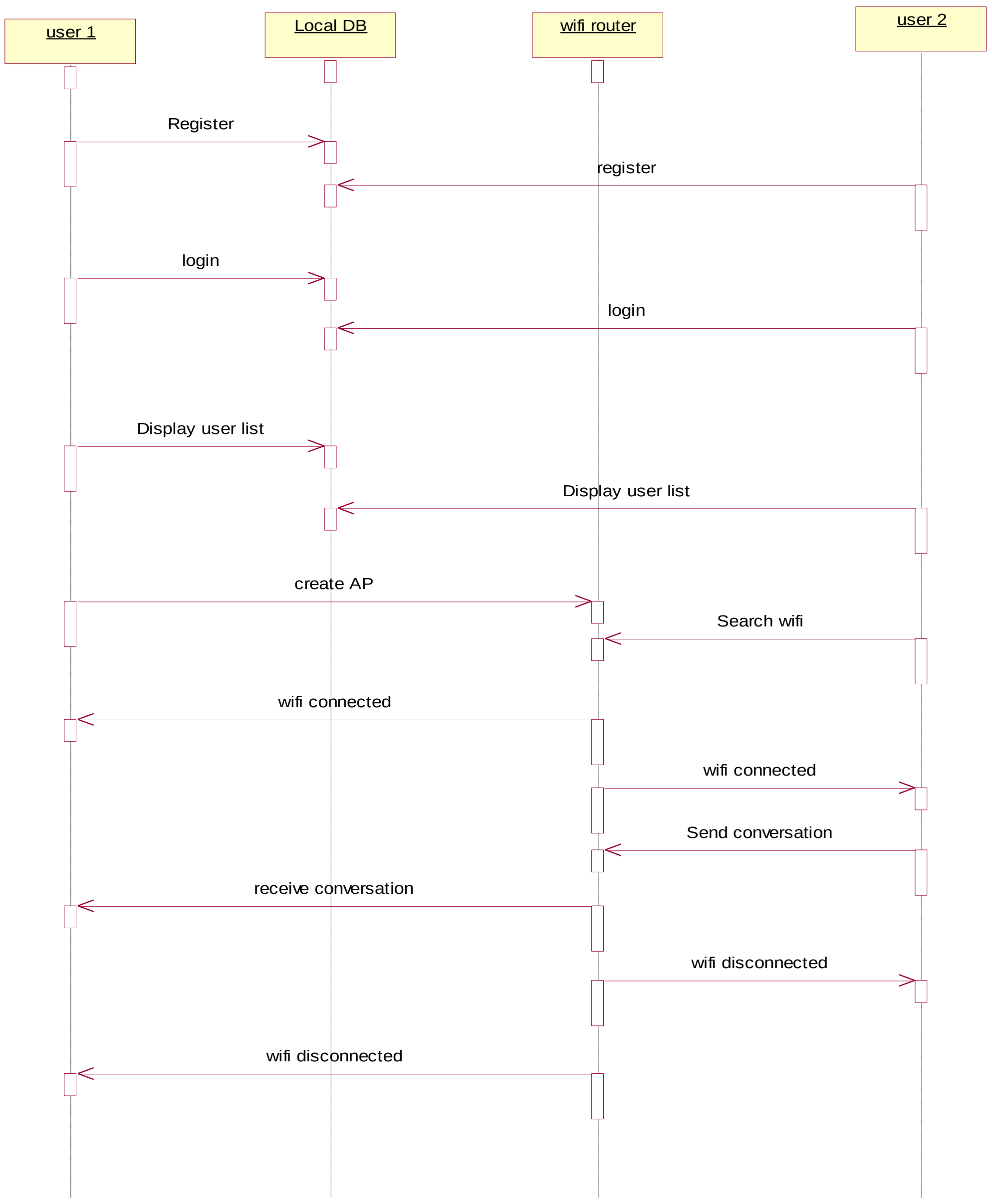


Fig 4.4 Sequence Diagram

4.3.3 ACTIVITY DIAGRAM

Activity diagram are graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

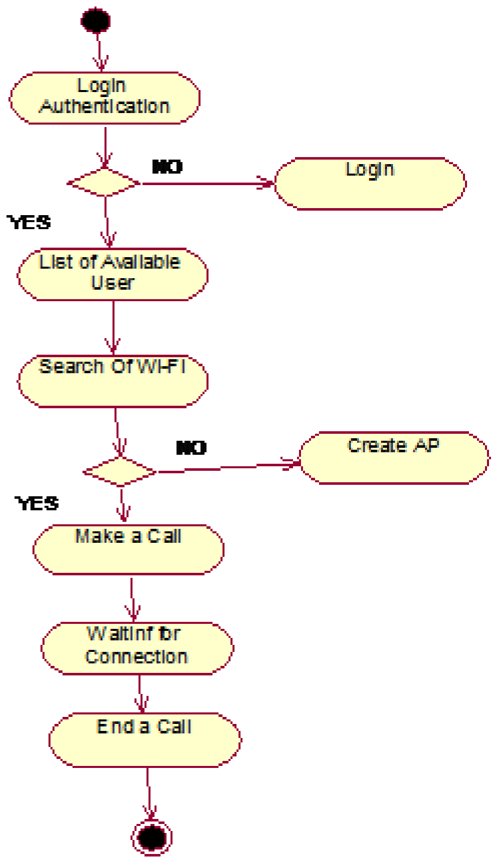


Fig 4.5:Activity Diagram

CHAPTER 5

IMPLEMENTATION AND TESTING

5.1 IMPLEMENTATION

We propose an model for making a free call in android mobiles over Wi-Fi network. Unlicensed Mobile Access (UMA) is a 3rd Generation Partnership Program global specification that provides a standard for service providers to merge mobile networks and wireless LANs into a single access network When making a call within the network, it displays the available user in the Wi-Fi network, and the required user is selected from the list. The application then sends Id in 128 bit encrypted form to the router, requesting a call to be placed. The android application at the destination intimates the user of the incoming call. If the call is accepted, the router connects both mobiles to make transmission of voice data. The result of the voice calls over Wi-Fi application using Wi-Fi enabled mobile phone devices mainly requires android Wi-Fi enabled mobile phone and Wi-Fi router. Both users must install the application in their mobile phones.

New user must register into an application, and the existing user can login with an username and password , it ask for name to be identified in the list of users available in Network. After entering the name it redirect to the user list in the Wi-Fi network, which is stored into Database. The database would be an online entity like cloud storage, from which the data can be shared to other people through the internet. The details stored in the database are retrieved in the form of list for communication. There is a status field which determines what the current activity taking place in the application. Any user in the list can initiate the call to communicate with any user in the list. When the user selects the particular users from the list to communicate, it redirect to the next screen were user can make a call to transmit the voice data. Once the user (sender) clicks the Connect button it search for the connections in the Wi-Fi network. If the network not found it will initiate a access point and wait for an destination user to get connected. When the both users get connected they can start transmitting the voice data over network. The user can transmit the voice data from the user list with no cost and time limits. The voice data is transmitted in the encrypted format between the users. The quality depends upon the Wi-Fi signal strength.

Using this application free calls can be made within the range of Wi-Fi. Android provides a robust, flexible environment for applications running on mobile. It includes flexible user interfaces, robust security built-in network protocols and support for networked and offline applications that can be downloaded dynamically.

INSTALLING WIFICALL APPLICATION

Step 1: Start.

Step 2: Install the application on android mobile.

Step 3: Click to open the application.

Step 4: New user can register in register page.

Step 5: Once the user registered, provide the user name and password in Login page.

Step 6: Once the user validates, it display the available users in Wi-Fi network.

MAKING A CALL

Step 1: Start

Step 2: Click on Connect button.

Step 3: This will search for the Wi-Fi AP if not found then it creates its own AP.

Step 4: Then in another mobile open the same application and select Connect button and now the Wi-Fi AP will be available and it connects to that mobile.

Step 5: Once the connection is setup, then select Make a call button and start speaking this will transmit those message in the encrypted format to the destination user.

Step 6: Then select stop to end a call.

5.2 TESTING

Testing is an important phase that focuses on an empirical investigation in which the results describe the quality of the system. It cannot confirm system functions properly under all conditions but can establish that it fails under specific conditions. The prime purpose of testing is to guarantee that system successfully built and tested in the development phase meets all the requirements and design.

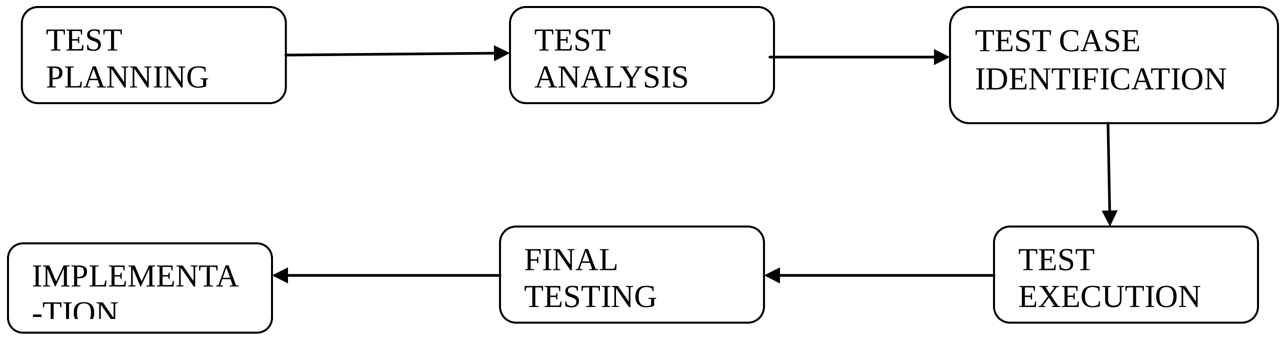


Fig:5.1 Process Of Testing

SIGN UP

No	Test Case	Expected Output	Observed Output	Result
1	Entering Name, Username, Password.	Create profile.	The profile is displayed	Pass
2	Missing data fields.	Error message should be displayed.	Please enter all the details.	Pass

Table 5.2: Sign Up

LOGIN

No	Test Case	Expected Output	Observed Output	Result
1	Entering the username and password	Display the profile.	The profile is displayed	Pass
2	Entering the wrong user name /Password.	Display the error message.	The error message is displayed.	Pass

Table 5.3: Login

MAKING A CALL

NO	**Test Case**	**Expected Output**	**Observed Output**	**Result**
1.	Searching for available users in Wi-Fi network	The list of Users must be displayed.	The available user list is displayed.	PASS
2.	Searching for Wi-Fi Access Point.	The list of access point must be displayed.	The access point list is displayed.	PASS
3.	Creating a new access point.	It should create a new access point.	The access point is created.	PASS
4.	Making a call to the available user from the list.	The call must be received to respective user from the list.	The is received to the user.	PASS
5.	Exit	The application must be closed.	The application closed.	PASS

Table 5.4: Making a Call

5.3 TEST PLAN

The project is tested to verify its correctness and identify the bugs. The test plan includes the various test cases that acts as the set of conditions or variables that determine whether the corresponding feature in the system is working as it originally established to do so. When this test plan is executed, the errors spotted are rectified and the final testing yields following result.

5.4 TEST ANALYSIS

In this phase of testing, the requirements for software testing are analysed and later its feasibility is determined. In the feasibility study the possibility of project development is found through suitable test cases.

5.5 RESULT

The application is tested and found to function as expected with no errors. This application provides an interface for the users to make a call through single Wi-Fi network in an efficient way. Thus the communication is carried out as per user's efficiency over Wi-Fi network.

CHAPTER 6
CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

Communication is a field where it is highly possible for the data to get corrupted, and its high imperative to secure the data to be transmitted. Our proposal provides an efficient mechanism to send voice calls over the Wi-Fi bandwidth by using encryption and decryption mechanism that render secured packet reception This Project provides a cheap, effective and secure means of communication within a specified network. The cost involved is only the initial setup cost and all calls within the network are free. This model will be very useful to solve the communication problems in large organizations, by making free voice calls through Wi-Fi.

6.2 FUTURE ENHANCEMENT

Our application involves making a call through Wi-Fi without the need of SIM card. Our future work includes enabling the conference call so that several user can communicate with one other simultaneously, and the quality of voice can be improved.

APPENDIX-A

SAMPLE SOURCE CODE

MAIN ACTIVITY.java

```
package com.app.mypro;

import android.util.Log;
import android.view.View;
import android.view.Window;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import android.app.Activity;
import android.app.Dialog;
import android.content.Intent;
import android.os.Bundle;

public class MainActivity extends Activity {

    LoginDataBaseAdapter loginDataBaseAdapter;
    Button login;
    Button registerr;
    EditText enterpassword;
    TextView forgetpass;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        login=(Button)findViewById(R.id.login_btn);
        registerr=(Button)findViewById(R.id.register_btn);
        enterpassword=(EditText)findViewById(R.id.password_edt);
        forgetpass=(TextView)findViewById(R.id.textView2);

        loginDataBaseAdapter                    =                    new
LoginDataBaseAdapter(getApplicationContext());
        loginDataBaseAdapter.open();

        registerr.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View v) {
// TODO Auto-generated method stub
                Intent i=new Intent(MainActivity.this,Registration.class);
                startActivity(i);
            }
        });

        login.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View v) {
// TODO Auto-generated method stub
                String Password=enterpassword.getText().toString();

                String
storedPassword=loginDataBaseAdapter.getSinlgeEntry(Password);

                if(Password.equals(storedPassword))
                {
                   Toast.makeText(MainActivity.this,        "Congrats:        Login
Successfully", Toast.LENGTH_LONG).show();
                   Intent ii=new Intent(MainActivity.this,Home.class);
                   startActivity(ii);
                }
                else
                if(Password.equals("")){
                   Toast.makeText(MainActivity.this,     "Please     Enter     Your
Password", Toast.LENGTH_LONG).show();
                }
                else
                {
                   Toast.makeText(MainActivity.this,     "Password     Incorrect",
Toast.LENGTH_LONG).show();
                }
            }
        });
        forgetpass.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View v) {
// TODO Auto-generated method stub

                final Dialog dialog = new Dialog(MainActivity.this);
                dialog.getWindow();
                dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
                dialog.setContentView(R.layout.activity_forget_search);
                dialog.show();

                final                                                    EditText
security=(EditText)dialog.findViewById(R.id.securityhint_edt);
                final                                                    TextView
getpass=(TextView)dialog.findViewById(R.id.textView3);

                Button ok=(Button)dialog.findViewById(R.id.getpassword_btn);
                Button cancel=(Button)dialog.findViewById(R.id.cancel_btn);

                ok.setOnClickListener(new View.OnClickListener() {

                    public void onClick(View v) {

                        String userName=security.getText().toString();
                        if(userName.equals(""))
                        {
```

```
                    Toast.makeText(getApplicationContext(), "Please enter
your securityhint", Toast.LENGTH_SHORT).show();
                  }
                  else
                  {
                    String
storedPassword=loginDataBaseAdapter.getAllTags(userName);
                    if(storedPassword==null)
                    {
                      Toast.makeText(getApplicationContext(), "Please enter
correct securityhint", Toast.LENGTH_SHORT).show();
                    }else{
                      Log.d("GET PASSWORD",storedPassword);
                      getpass.setText(storedPassword);
                    }
                  }
                }
              });
              cancel.setOnClickListener(new OnClickListener() {

                @Override
                public void onClick(View v) {
// TODO Auto-generated method stub
                  dialog.dismiss();
                }
              });
```

```
                dialog.show();
            }
        });
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
// Close The Database
        loginDataBaseAdapter.close();
    }

}
```

NETWORK ADAPTER:

```
             package com.app.mypro;

/**
 * Created by AK Viki on 4/3/2015.
 */
import android.content.Context;
import android.net.ConnectivityManager;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiConfiguration;
import android.net.wifi.WifiManager;
import android.preference.PreferenceManager;
import android.text.format.Formatter;
import android.util.Log;

import java.lang.reflect.Method;

public class NetworkAdapter
{
    private static String OtherName = "---";
    static String OtherClientIP = "0.0.0.0";

    final static String NetworkSSID = "walkie";
    final static String NetworkKey = "walkietalkie";

    final static int StartWifiTimeout = 10000;
    final static int StopWifiTimeout = 10000;
    final static int StartApTimeout = 10000;
    final static int StopApTimeout = 10000;
    final static int BeginScanTimeout = 5000;
    final static int BeginConnectionTimeout = 30000;

    final static int WaitTimeSpan = 250;

    static int ActionCounter = 0;

    private static boolean ConditionalWait(int timeout, CallbackEvent
statusCallback, StopEvent stopNotifier, ConditionalEvent event)
    {
        int ActionId = ActionCounter ++;
        try
        {
            Log.d("Wifi action", "Action # " + ActionId + ": " +
event.getOperationName() + " [ precondition ]");

            if (event.checkCondition())
                return true;

            Log.d("Wifi action", "Action # " + ActionId + ": " +
event.getOperationName() + " [ start ]");

            if (!event.startEvent())
                return false;

            statusCallback.onCallback(event.getOperationName());

            boolean status = false;
            timeout /= WaitTimeSpan;

            Log.d("Wifi action", "Action # " + ActionId + ": " +
event.getOperationName() + "[ loop ]");
            Log.v("Wifi action", "Action # " + ActionId + ": " +
event.getOperationName() + "[ timeout " + (timeout * WaitTimeSpan) + "
]");
```

```
        while ((stopNotifier == null || !stopNotifier.isStopped()) && !(status
= event.checkCondition()) && (timeout-- > 0))
        {
           try
           {
              Thread.sleep(WaitTimeSpan);
           }
           catch (Exception e)
           {
              e.printStackTrace();
           }

           Log.v("Wifi action", "Action # " + ActionId + ": " +
event.getOperationName() + "[ timeout " + (timeout * WaitTimeSpan) + "
]");
        }

        if (stopNotifier != null && stopNotifier.isStopped())
        {
           Log.i("Wifi action", "Action # " + ActionId + ": " +
event.getOperationName() + " [ cancelled ]");

           event.onTimeout();
           return false;
        }

        if (status)
```

```
            {
                Log.i("Wifi action", "Action # " + ActionId + ": " +
event.getOperationName() + " [ success ]");

                return true;
            }
            else
            {
                Log.w("Wifi action", "Action # " + ActionId + ": " +
event.getOperationName() + " [ timeout ]");

                event.onTimeout();
                return false;
            }
        }
        catch (Exception e)
        {
            Log.e("Wifi action", "Action # " + ActionId + ": " +
event.getOperationName() + " [ exception ]");

            try
            {
                event.onTimeout();
            }
            catch (Exception e2)
            {
                e2.printStackTrace();
```

```
          }

          e.printStackTrace();
          return false;
        }
    }

    public static boolean StartWifi(Context context, CallbackEvent
statusCallback, StopEvent stopNotifier)
    {
        final WifiManager wifiManager =
(WifiManager)context.getSystemService(Context.WIFI_SERVICE);
        final Context contextHandler = context;
        final CallbackEvent statusCallbackHandler = statusCallback;
        final StopEvent stopNotifierHandler = stopNotifier;

        return ConditionalWait(StartWifiTimeout, statusCallback, stopNotifier,
new ConditionalEvent()
        {
            public boolean startEvent() throws Exception
            {
                if (!StopAccessPoint(contextHandler, statusCallbackHandler,
stopNotifierHandler))
                    return false;

                return wifiManager.setWifiEnabled(true);
            }
            public boolean checkCondition() throws Exception
            {
                return (wifiManager.getWifiState() ==
WifiManager.WIFI_STATE_ENABLED);
            }

            public void onTimeout() throws Exception
            {
                StopWifi(contextHandler, statusCallbackHandler, null);
            }

            public String getOperationName()
            {
                return "Turning WiFi on...";
            }
        });
    }

    public static boolean StopWifi(Context context, CallbackEvent
statusCallback, StopEvent stopNotifier)
    {
        final WifiManager wifiManager =
(WifiManager)context.getSystemService(Context.WIFI_SERVICE);

        return ConditionalWait(StopWifiTimeout, statusCallback, stopNotifier,
new ConditionalEvent()
```

```
        {
            public boolean startEvent() throws Exception
            {
                return wifiManager.setWifiEnabled(false);
            }

            public boolean checkCondition() throws Exception
            {
                return (wifiManager.getWifiState() ==
WifiManager.WIFI_STATE_DISABLED);
            }

            public void onTimeout() throws Exception
            {
            }

            public String getOperationName()
            {
                return "Turning WiFi off...";
            }
        });
    }

    public static boolean StartAccessPoint(Context context, CallbackEvent
statusCallback, StopEvent stopNotifier)
    {
        final WifiManager wifiManager =
(WifiManager)context.getSystemService(Context.WIFI_SERVICE);
        final Context contextHandler = context;
        final CallbackEvent statusCallbackHandler = statusCallback;
        final StopEvent stopNotifierHandler = stopNotifier;

        return ConditionalWait(StartApTimeout, statusCallback, stopNotifier,
new ConditionalEvent()
        {
            public boolean startEvent() throws Exception
            {
                if (!StopWifi(contextHandler, statusCallbackHandler,
stopNotifierHandler))
                    return false;

                Method SetWifiApEnabled =
wifiManager.getClass().getMethod("setWifiApEnabled",
WifiConfiguration.class, boolean.class);

                WifiConfiguration netConfig = new WifiConfiguration();
                netConfig.SSID = NetworkSSID;
                netConfig.preSharedKey = NetworkKey;

netConfig.allowedAuthAlgorithms.set(WifiConfiguration.AuthAlgorithm.S
HARED);
                netConfig.allowedProtocols.set(WifiConfiguration.Protocol.RSN);
netConfig.allowedProtocols.set(WifiConfiguration.Protocol.WPA);

netConfig.allowedKeyManagement.set(WifiConfiguration.KeyMgmt.WPA_
PSK);

netConfig.allowedPairwiseCiphers.set(WifiConfiguration.PairwiseCipher.C
CMP);

netConfig.allowedPairwiseCiphers.set(WifiConfiguration.PairwiseCipher.T
KIP);

netConfig.allowedGroupCiphers.set(WifiConfiguration.GroupCipher.CCMP
);

netConfig.allowedGroupCiphers.set(WifiConfiguration.GroupCipher.TKIP);

            return (Boolean)SetWifiApEnabled.invoke(wifiManager,
netConfig, true);
        }

        public boolean checkCondition() throws Exception
        {
            Method IsWifiApEnabled =
wifiManager.getClass().getMethod("isWifiApEnabled");

            return ((Boolean)IsWifiApEnabled.invoke(wifiManager));
```

```
          }

          public void onTimeout() throws Exception
          {
             StopAccessPoint(contextHandler, statusCallbackHandler, null);
          }

          public String getOperationName()
          {
             return "Turning AP on...";
          }
       });
    }

    public static boolean StopAccessPoint(Context context, CallbackEvent
statusCallback, StopEvent stopNotifier)
    {
       final WifiManager wifiManager =
(WifiManager)context.getSystemService(Context.WIFI_SERVICE);

       return ConditionalWait(StopApTimeout, statusCallback, stopNotifier,
new ConditionalEvent()
       {
          public boolean startEvent() throws Exception
          {
                Method SetWifiApEnabled =
wifiManager.getClass().getMethod("setWifiApEnabled",
WifiConfiguration.class, boolean.class);

                return (Boolean)SetWifiApEnabled.invoke(wifiManager, null,
false);
            }

            public boolean checkCondition() throws Exception
            {
                Method IsWifiApEnabled =
wifiManager.getClass().getMethod("isWifiApEnabled");

                return (!(Boolean)IsWifiApEnabled.invoke(wifiManager));
            }

            public void onTimeout() throws Exception
            {
            }

            public String getOperationName()
            {
                return "Turning AP off...";
            }
        });
    }
```

```
  public static boolean BeginNetworkScan(Context context, CallbackEvent
statusCallback, StopEvent stopNotifier)
  {
    final WifiManager wifiManager =
(WifiManager)context.getSystemService(Context.WIFI_SERVICE);
    final Context contextHandler = context;
    final CallbackEvent statusCallbackHandler = statusCallback;
    final StopEvent stopNotifierHandler = stopNotifier;

    return ConditionalWait(BeginScanTimeout, statusCallback,
stopNotifier, new ConditionalEvent()
    {
      public boolean startEvent() throws Exception
      {
        if (!StartWifi(contextHandler, statusCallbackHandler,
stopNotifierHandler))
          return false;

        return wifiManager.startScan();
      }

      public boolean checkCondition() throws Exception
      {
        if (wifiManager.getScanResults() != null)
        {
          for (ScanResult result : wifiManager.getScanResults())
          {
```

```
                    if (result.SSID.equals(NetworkSSID) || result.SSID.equals(""
+ NetworkSSID + ""))
                        return true;
                    }
                }

                return false;
            }

            public void onTimeout() throws Exception
            {
                StopWifi(contextHandler, statusCallbackHandler, null);
            }

            public String getOperationName()
            {
                return "Scanning for network...";
            }
        });
    }

    public static boolean BeginNetworkConnection(Context context,
CallbackEvent statusCallback, StopEvent stopNotifier)
    {
        final WifiManager wifiManager =
(WifiManager)context.getSystemService(Context.WIFI_SERVICE);
```

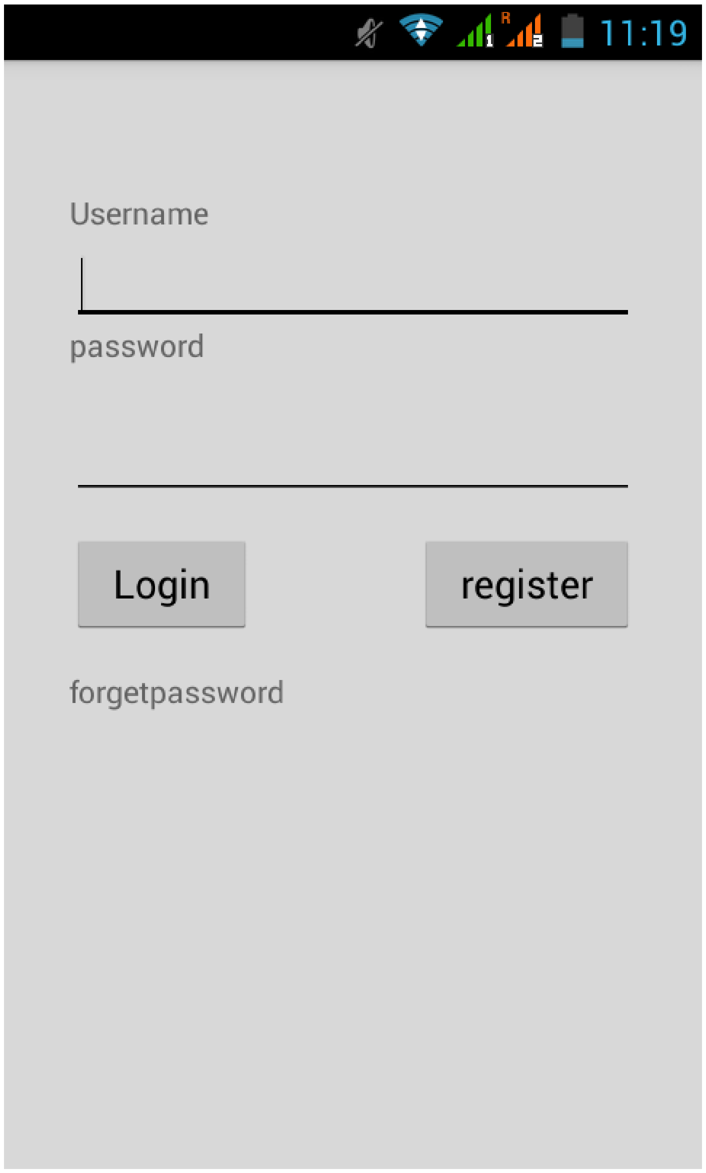
```
        final ConnectivityManager connectivityManager =
(ConnectivityManager)context.getSystemService(Context.CONNECTIVIT
Y_SERVICE);
        final Context contextHandler = context;
        final CallbackEvent statusCallbackHandler = statusCallback;
        final StopEvent stopNotifierHandler = stopNotifier;

        return ConditionalWait(BeginConnectionTimeout, statusCallback,
stopNotifier, new ConditionalEvent()
        {
            public boolean startEvent() throws Exception
            {
                if (!BeginNetworkScan(contextHandler, statusCallbackHandler,
stopNotifierHandler))
                    return false;

                if (wifiManager.getConfiguredNetworks() != null)
                {
                    for (WifiConfiguration Network :
wifiManager.getConfiguredNetworks())
                    {
                        if (Network.SSID.equals(NetworkSSID) ||
Network.SSID.equals("" + NetworkSSID + ""))
                        {
                            wifiManager.removeNetwork(Network.networkId);
                            break;
                        }
                    }
                }

                WifiConfiguration netConfig = new WifiConfiguration();
                netConfig.SSID = "" + NetworkSSID + "";
                netConfig.preSharedKey = "" + NetworkKey + "";

                int networkId = wifiManager.addNetwork(netConfig);

                if (networkId == -1)
                    return false;

                return wifiManager.enableNetwork(networkId, true);
            }

            public boolean checkCondition() throws Exception
            {
                if (wifiManager.getConnectionInfo() == null)
                    return false;

                if (wifiManager.getConnectionInfo().getSSID() == null)
                    return false;

                if
(!wifiManager.getConnectionInfo().getSSID().equals(NetworkSSID) &&
                        !wifiManager.getConnectionInfo().getSSID().equals("" +
NetworkSSID + ""))
```

```
                return false;

            if (connectivityManager.getActiveNetworkInfo() == null)
                return false;

            if (!connectivityManager.getActiveNetworkInfo().isConnected())
                return false;

            return true;
        }

        public void onTimeout() throws Exception
        {
            StopWifi(contextHandler, statusCallbackHandler, null);
        }

        public String getOperationName()
        {
            return "Connecting to network...";
        }
    });
}

public static String GetOtherClientIP()
{
    return OtherClientIP;
}
```

```
    public static void SetOtherClientIP(String address)
    {
        OtherClientIP = address;
    }

    public static String GetUserName(Context context)
    {
        if (PreferenceManager.getDefaultSharedPreferences(context) == null)
            return "User";

        return
PreferenceManager.getDefaultSharedPreferences(context).getString("nickNa
mePref", "User");
    }

    public static String GetOtherName()
    {
        return OtherName;
    }

    public static void SetOtherName(String name)
    {
        OtherName = name;
    }

    public static String GetServerIP(Context context)
    {
        final WifiManager wifiManager =
(WifiManager)context.getSystemService(Context.WIFI_SERVICE);

        if (wifiManager.getDhcpInfo() == null)
            return "0.0.0.0";

        return
Formatter.formatIpAddress(wifiManager.getDhcpInfo().gateway);
    }
}
```

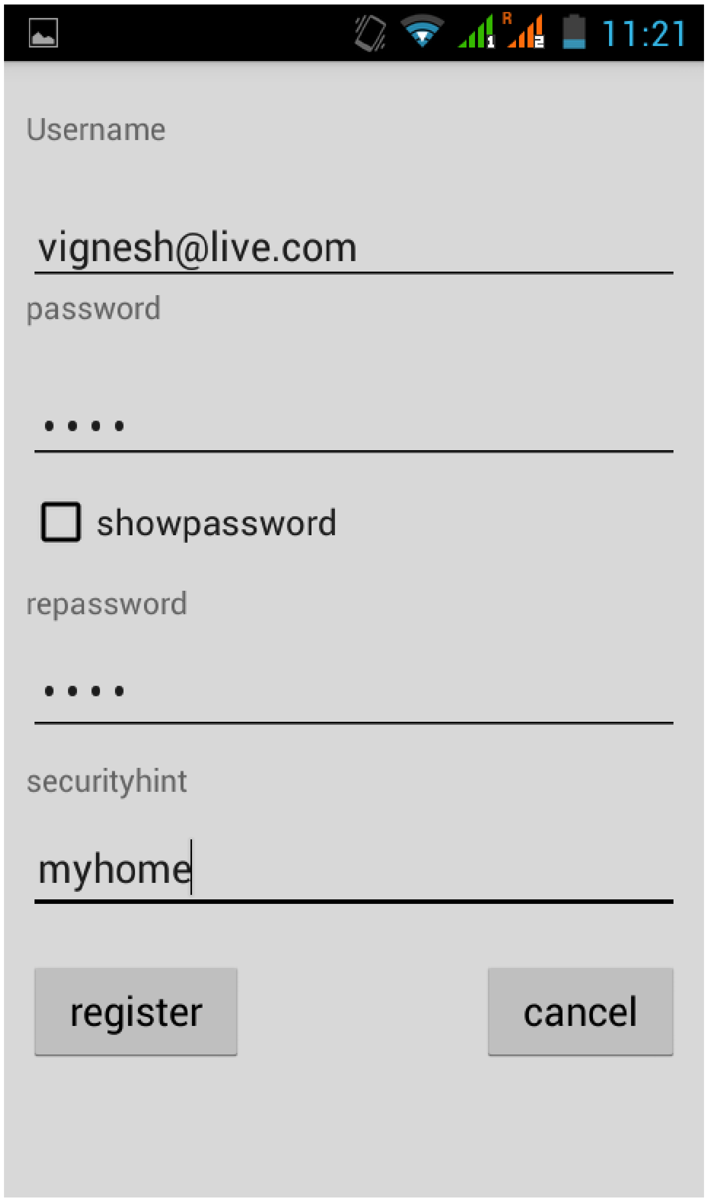
SAMPLE SCREEN SHOTS

LOGIN



When the user enters into the application, the first page that appears is the login page.

REGISTER PAGE



11:21

Username

vignesh@live.com

password

••••

☐ showpassword

repassword

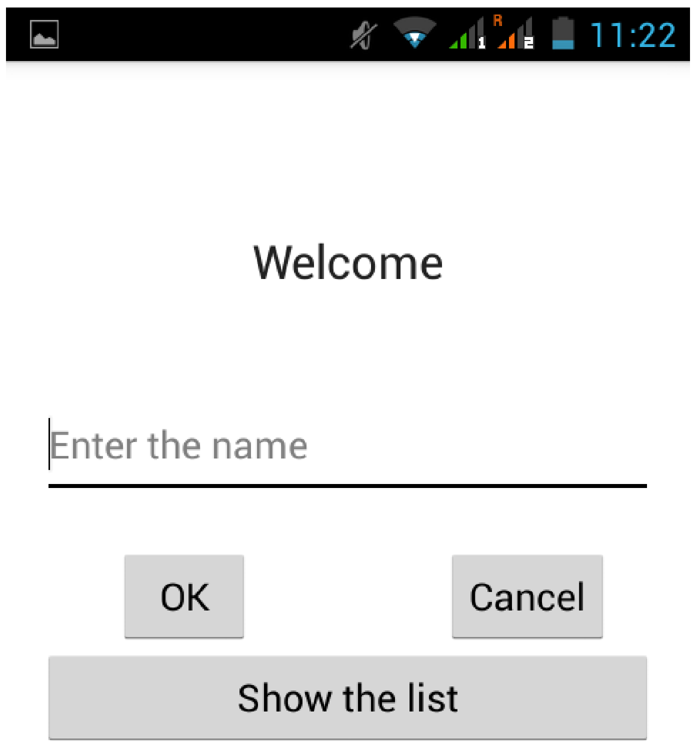
••••

securityhint

myhome

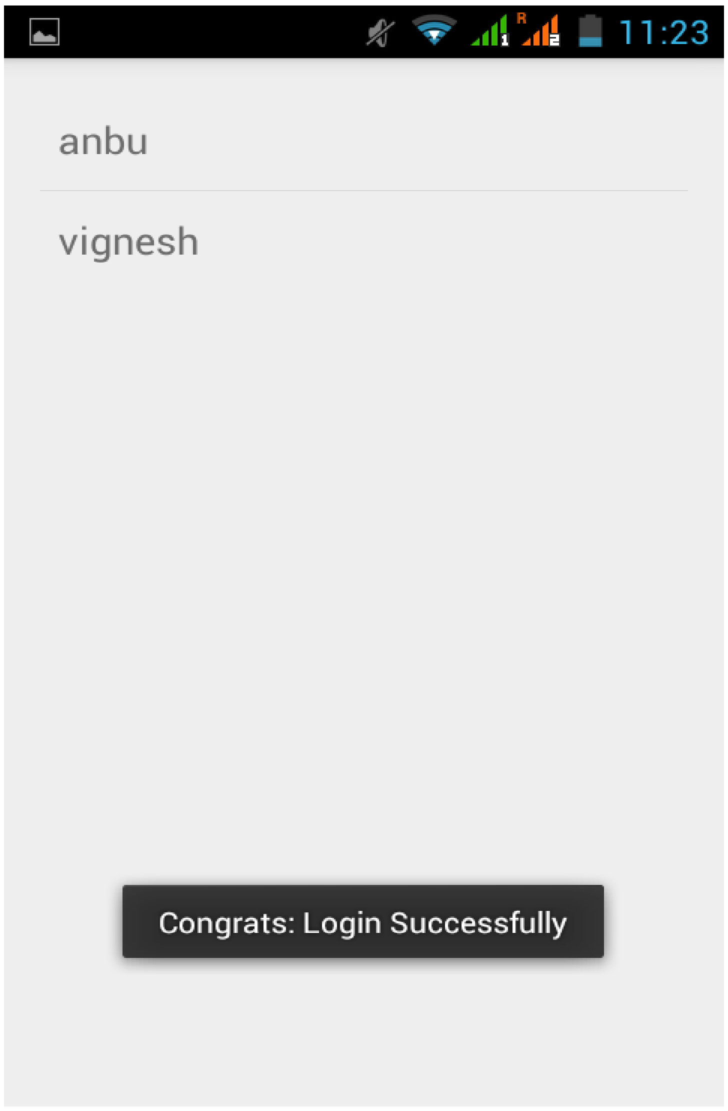
The registration page consists of team id, password and security hint.

ENTERING THE NAME



The user enters the name to be identify in the Wi-Fi network.

LIST OF AVAILABLE USERS


11:23

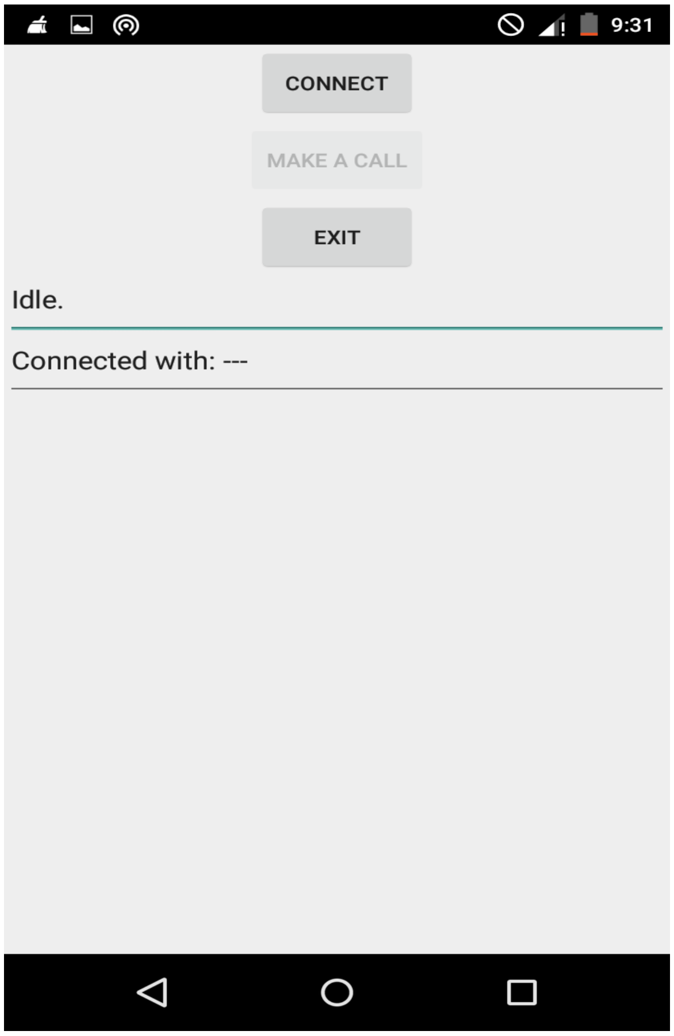
anbu

vignesh

Congrats: Login Successfully

It displays the available users in the network.

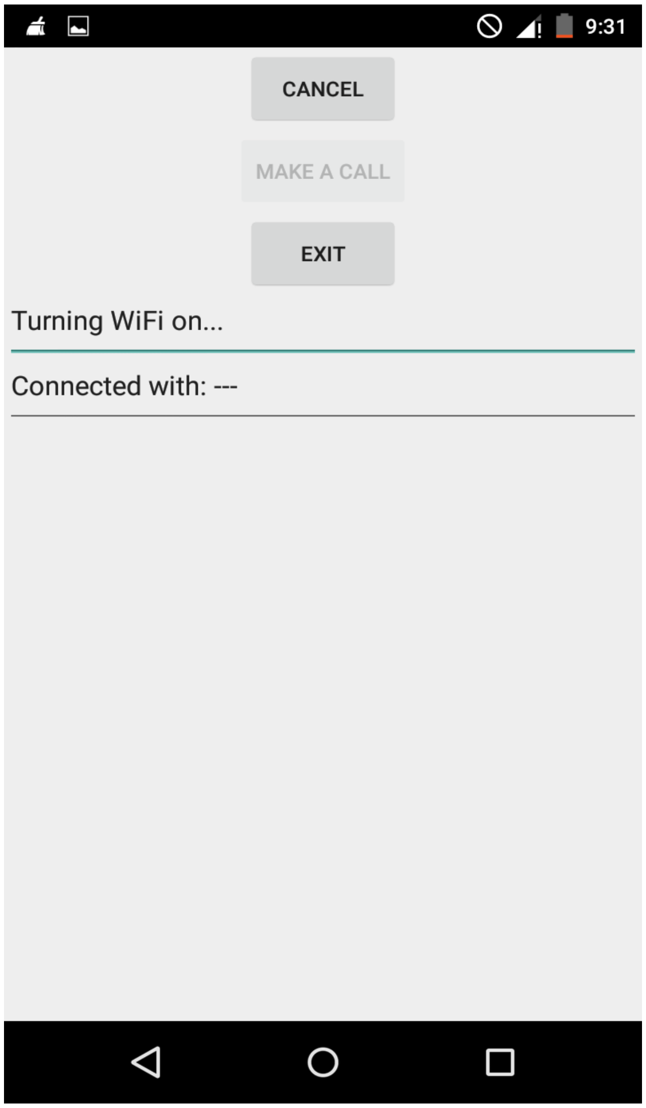
MAIN PAGE



Idle.

Connected with: ---

CONNECTION

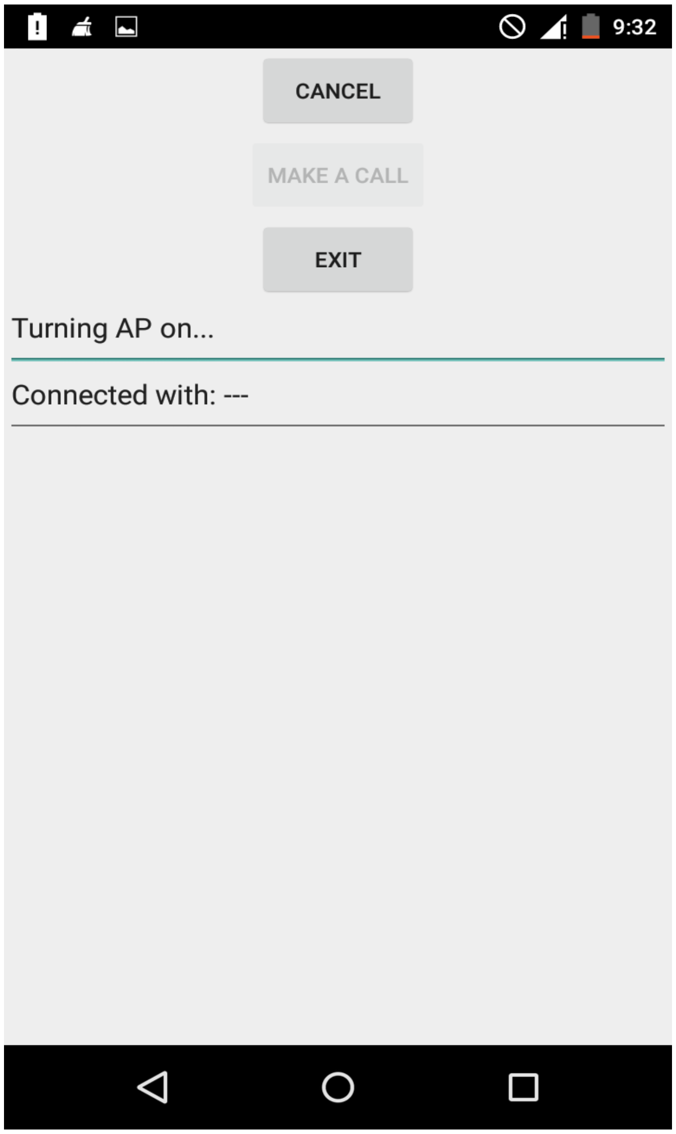


Turning WiFi on...

Connected with: ---

Connect button is clicked make to turn Wi-Fi on and search for the Wi-Fi AP.

TURNING AP ON

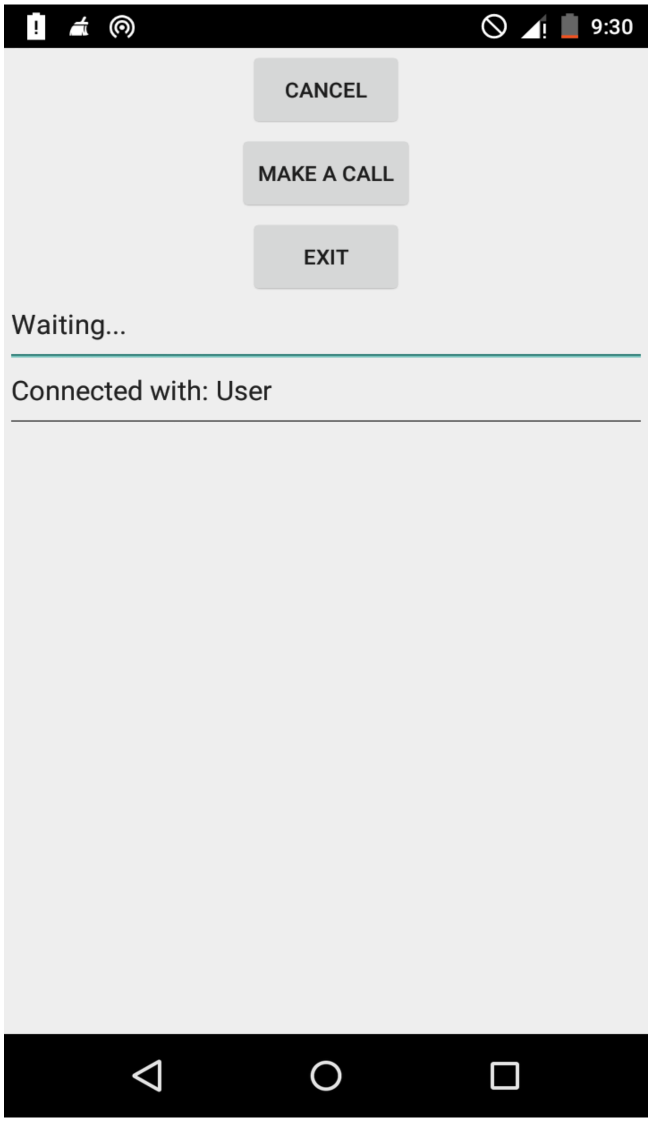


Turning AP on...

Connected with: ---

The status message turning AP on start the Wi-Fi tethering hotspot which act as AP.

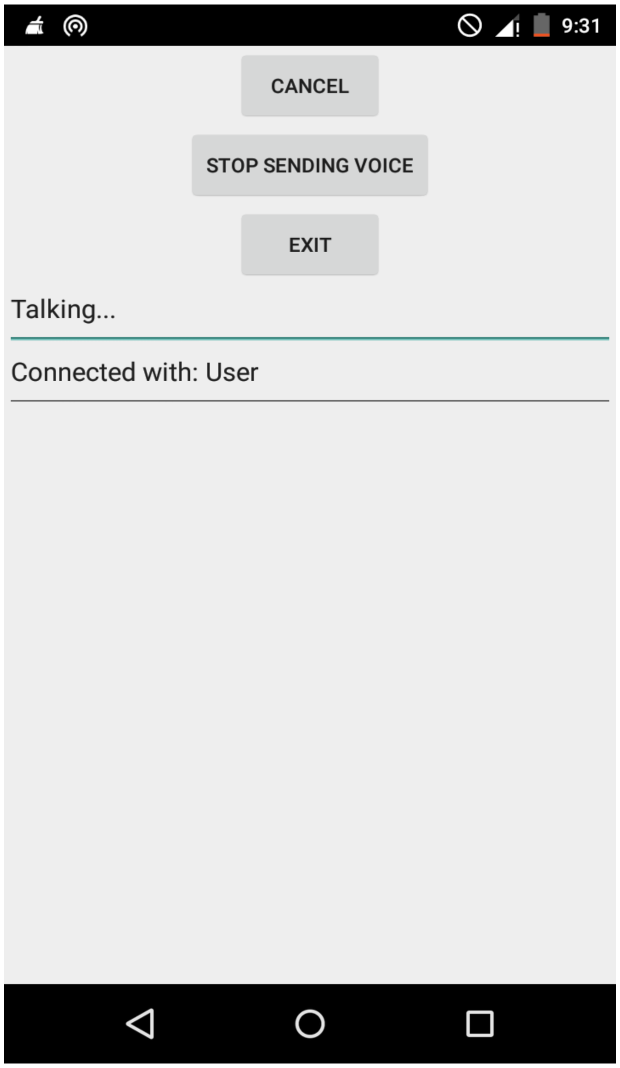
MAKE A CALL



Waiting...

Connected with: User

LISTENING TO VOICE



Talking...

Connected with: User

REFERENCES

- Implementing 802.11,802.16,802.20 Wireless Networks : Planning, Troubleshooting, and Operations, Ron Olexa.

- Real 802.11 security : Wi-Fi Protected Access and 802.11i, Jon Edney, William A.Arabaugh.

- Cao Peng and Yang Xuejum, "Performance analysis of SIP-based push- to-talk service for GPRS/cdma2000 network," 2005 2nd International Conference on Mobile Technology, Applications and Systems.

- GSM, http://www.gsmworld.com/technologvigsm/index.htill

- WIFI IEEE 802.11, http://www.wifinotes.com/IEEE-802.11.html.273.

- Intranet Based Messaging Service on Android Smart phones and Tablets International Journal of Advanced Research in Computer Science and Software Engineering.

- Engineering (Volume 3, Issue 7, July 2013).[2] Voice Calls between Wireless (Android) Phones and a Cooperative Application for Sending Sms over Wi-Fi Networks By Mr. Sandip Rane, Miss. Jaya Suradkar, Volume 12 Issue 4 Version 1.0 February 2012.