

# CHAPTER 1

## SOFTWARE REQUIREMENTS SPECIFICATION

---

### **INDEX:**

#### **1 INTRODUCTION**

1.1	DOCUMENT PURPOSE .....	2
1.2	PRODUCT SCOPE .....	2
1.3	INTENDED AUDIENCE AND DOCUMENT OVERVIEW .....	2
1.4	DEFINITIONS, ACRONYMS AND ABBREVIATIONS .....	2

#### **2 OVERALL DESCRIPTION**

2.1	PRODUCT PERSPECTIVE .....	3
2.2	PRODUCT FUNCTIONALITY .....	3
2.3	USERS AND CHARACTERISTICS .....	3
2.4	OPERATING ENVIRONMENT .....	4
2.5	USER DOCUMENTATION .....	4
2.6	ASSUMPTIONS AND DEPENDENCIES .....	4

#### **3 SPECIFIC REQUIREMENTS**

3.1	EXTERNAL INTERFACE REQUIREMENTS .....	5
3.2	FUNCTIONAL REQUIREMENTS .....	5

#### **4 OTHER NON-FUNCTIONAL REQUIREMENTS**

4.1	PERFORMANCE REQUIREMENTS .....	6
-----	--------------------------------	---

# 1. INTRODUCTION

The document aims at defining the overall software requirements for “WiFiCall” and Server side application. Efforts are been made to define the requirement exhaustively and accurately. The final product will be having only features/functionalities mentioned in this document and assumption for any additional functionality/feature should not be made by any of the parties involved in developing /testing/implementing the products. In case it is required to have some additional features, a formal change request will need to be raised and subsequently a new release of this document and/or product will be produced.

## Document Purpose

This specification document describes the capabilities that will be provided by the android applications “**WiFiCall**” and “**Server**”. It also states the various required constraints by which the system will abide. The intended audience for this document is the development team, testing team and the users of the product.

### 1.1 Product Scope

The software “**WiFiCall**” is an application that will be used by general users to make voice calls using an Aakash tablet . To make a call both the users need to be registered at the server. This communication is via wifi and will incur no expenditure to the end users. Communication between any two users will be enabled as long as both are logged-in at the server.

An application “Server” is required to register the users and maintain the information regarding the MAC and current IP addresses of all the logged-in users. The clients may be connected to same/different WiFi given both the routers are registered at the server. “Server” may be connected to either WiFi or LAN.

### 1.2 Intended Audience and Document Overview

The intended audience for this document is the development team, testing team and the users of the product.

### 1.3 Definitions, Acronyms and Abbreviations

The following abbreviations have been used throughout:

IEEE - Institute of Electrical and Electronics Engineers

SRS - Software Requirement Specification

### 1.4 Document Conventions

This document follows the IEEE formatting requirements:

- Arial font size 11, or 12 throughout the document for text
- italics for comments
- document text is single space.

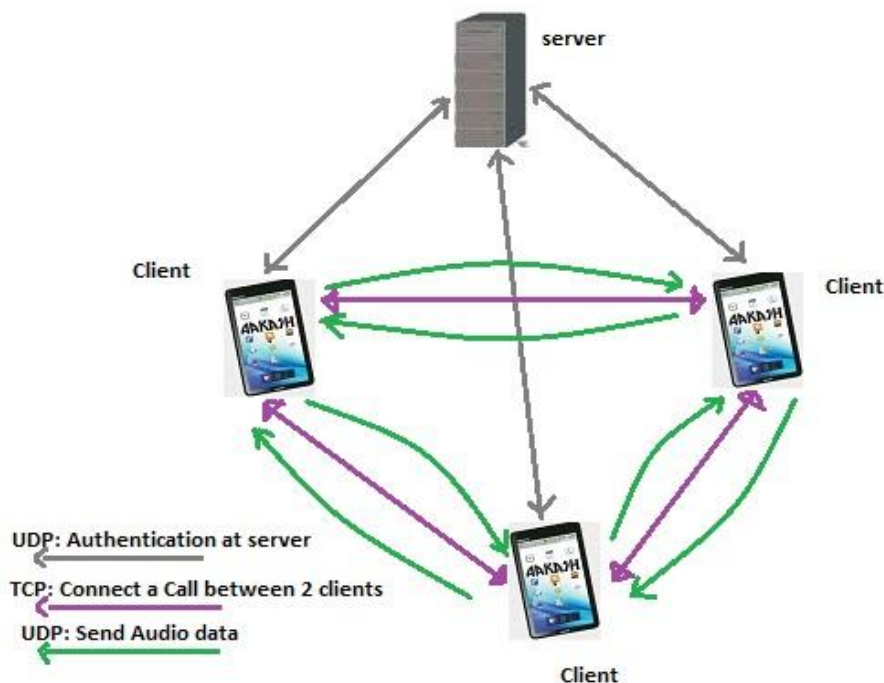
## 2. OVERALL DESCRIPTION

### 2.1 Product Perspective

In the present day where communication plays such an important role in day-to-day activity, this product was proposed with the aim to provide free communication over smaller distance eg. within an organisation campus. This product is a part of Aakash tablet.

WiFiCall application can be implemented on any android based device.

The “Server” program is available in two versions. One version can be implemented on the tablet itself while the other is suitable for desktop computers.



### 2.2 Product Functionality

Major functions of the WiFiCall application:

- Connect and Login user to the desired server.
- Initiate a call to any available user.
- Receive a call from available user

Major functions of the Server software:

- Register the users who wish to use this product and allocate their tablet a unique roll no.
- Login and logout the users when they start and close their application respectively.

### 2.3 Users and Characteristics

The users of the “WiFiCall” software are the general users:

- The users are assumed to have basic knowledge of tablets (or android devices) and wifi connectivity.
- The users need to be registered at the server, where they are allocated unique roll no before they can use this software.

The users of the “Server” software can be either general user or system administrator at the corresponding organization:

- The users are assumed to have basic knowledge of computers or android based devices and wifi or LAN connectivity.
- The user must be an authorized one to handle client registrations.

## **2.4 Operating Environment**

The application “WiFiCall” will be developed to operate on any android based devices, with android versions 2.2 and later, like mobile phones and tablets. It requires a wifi connection in the device. For voice communication a speaker and an internal/external microphone is required.

Server software has two versions so that it can either be implemented on desktop based computers (windows/linux based) or on the tablet (android based, 2.2 and later) itself. It requires a WiFi or LAN connection to connect to all the users.

## **2.5 User Documentation**

A detailed user manual and possibly on-line help will be delivered along with the software.

## **2.6 Assumptions and Dependencies**

- The deadline must be met.
- The product must be reliable.
- The architecture must be open so that additional functionality may be added later.
- The product must be user-friendly.
- Tools we are going to use
  - JDK 1.6 or later
  - JRE 1.6 or later
  - Eclipse
  - Android SDK
  - ADT Plugins.
- From the very start of this project we are aware of time constraints so the main emphasis is on extensibility and parallel development. We shall try our best to ensure that project deadlines are met.

## **3. SPECIFIC REQUIREMENTS**

### **3.1 External Interface Requirements**

#### **3.1.1 User Interfaces**

For the “WiFiCall” application, the user interface will provide buttons to- login, change settings, start call and end call. The application will have its own contact list to save roll no and name of other users. The user will be notified with proper error messages in case of connection errors. The “Server” software has a simpler user interface. It provides facility to enter a user’s information and register it by the administrator. It also shows the list of all registered users and indicate whether they are online/offline. Request from any user is processed automatically by the server software as long as it is running, without any involvement of manual help.

#### **3.1.2 Hardware Interfaces**

- The computer or device should be connected to a wifi access point.
- To use the “WiFiCall” application a speaker and an internal/external microphone is required with the device.

#### **3.1.3 Software Interfaces**

Software interface for “WiFiCall” application:

- Any device based on android operating system versions 2.2 and later.
- The device should provide facility for lightweight Database Management System – SQLite.

Software interface for “Server” software:

- Any computer with windows or linux based operating system (Database Management System used is MySQL).
- Any device based on android operating system versions 2.2 and later (Database Management system used is SQLite).

#### **3.1.4 Communication Interfaces**

- The product uses UDP protocol for communication (login/logout) between a client and the server.
- TCP protocol is used to establish a call between two clients.
- UDP protocol is used for transmitting the audio data between two users during the call.

### **3.2 Functional Requirements**

Major functions of the WiFiCall application:

- Login user to the desired server.
- Initiate a call to a user.
- Receive a call from another user.

- A contact list is provided where the user may save contact no. and name of other users.
- User may set the frequency for audio communication depending on the device on which the application is running.

Major functions of the Server software:

- Three text boxes are provided to enter the details of a new user for its registration and a unique roll no is allocated to their tablet.
- Keeps track of all registered users and mark them as available or offline when they start and close their application respectively.

## **4. OTHER NON-FUNCTIONAL REQUIREMENTS**

### **4.1 Performance Requirements**

- Any transaction between a client and the server will not take more than 5 seconds.
- Establishing a call connection between two clients will not take more than 20 seconds.
- The lag in the audio data being transferred between the two users during call will be at-most 3-4 seconds.
- Any no. of users may be logged-in at the server at any time.
- The two users need to be connected to same/different access points of the same network.
- The “WiFiCall” application should be light to minimise the power consumption of the device.

# CHAPTER 2

## DESIGN DOCUMENTS

---

### **INDEX:**

#### **1 INTRODUCTION**

1.1	BACKGROUND .....	8
1.2	DESIGN GOALS .....	8

#### **2 PROJECT PLAN**

2.1	TITLE AND SCOPE OF PROJECT .....	8
2.2	RESOURCE REQUIREMENTS .....	8
2.3	MODEL USED (ITERATIVE) .....	9

#### **3 DESIGN AND IMPLEMENTATION**

3.1	HIGH LEVEL DESIGN DOCUMENT .....	10
3.2	LOW LEVEL DESIGN DOCUMENT .....	18

#### **4 CHALLENGES AND THEIR SOLUTIONS.....21**

#### **5 SUMMARY AND CONCLUSION**

5.1	SUMMARY .....	22
5.2	FURTHER ENHANCEMENTS .....	22
5.3	REFERENCES AND BIBLIOGRAPHY .....	22

---

## 1. INTRODUCTION

### 1.1 Background

Aakash, the low cost Indian tablet, can be used in a variety of ways to spread literacy and education in the country. It is an excellent device which can be used by Indian students and teachers to cater their needs. There are many efforts going on in IIT Bombay to make the device more powerful and enhance the utility of the tablet.

In this project we have tried to make voice calls between Aakash tablets through WiFi connectivity. Though the device has GSM calling facility but there is cost involved in GSM calling. Our objective was to provide free voice calling between the tablets by taking help of it's WiFi connectivity.

### 1.2 Design Goals

High Quality Voice Transmission was one of our design goals. It is very important to make the sound quality good in this kind of voice communication application.

Minimum lag in the voice transmission was also one of the design goals.

Reducing the server overhead was necessary, as when many tablets would be connected to the server there would be sufficient load on the server. The server is very important and it would be running continuously.

Making the server efficient such that it can handle a large number of tablet connections simultaneously without crashing.

## 2. PROJECT PLAN

### 2.1 Title and Scope of the Project

Title: Peer to Peer Voice Communication System

Scope:

The software “**WiFiCall**” is an application that will be used by general users to make voice calls using an Aakash tablet . To make a call both the users need to be registered at the server. This communication is via wifi and will incur no expenditure to the end users. Communication between any two users will be enabled as long as both are logged-in at the server.

An application “Server” is required to register the users and maintain the information regarding the MAC and current IP addresses of all the logged-in users. The clients may be connected to same/different WiFi given both the routers are registered at the server. “Server” may be connected to either WiFi or LAN.

### 2.2 Resource Requirement

#### 2.2.1 H/W Requirement

- The Server (Desktop/tablet) and Client (tablet) should have WiFi connectivity..
- To use the “WiFiCall” application a speaker and an internal/external microphone is required with the device



### 2.2.2 S/W Requirement

Software interface for “WiFiCall” application:

- Any device based on android operating system versions 2.2 and higher.
- Support for lightweight Database Management System –SQLite.

Software interface for “Server” software:

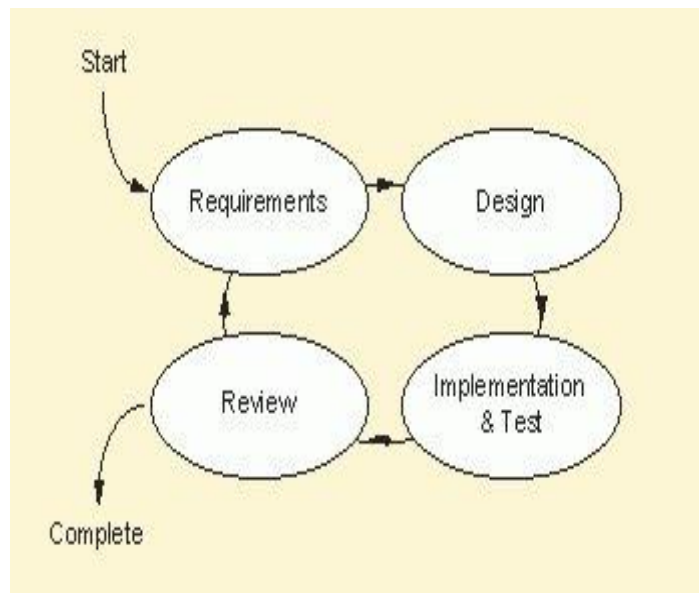
- Any computer with windows or linux based operating system (Database Management System used is MySQL).

OR

- Any device based on android operating system versions 2.2 and higher (Database Management system used is SQLite).

### 2.3 Model Used (Iterative Model)

The iterative lifecycle model does not attempt to start with a full specification of requirements. Instead, development begins by specifying and implementing just part of the software, which can then be reviewed in order to identify further requirements. This process is then repeated, producing a new version of the software for each cycle of the model , until the product is accepted as shown below:



A **Requirements** phase- in which the requirements for the software are gathered and analyzed. Iteration should eventually result in a requirements phase that produces a complete and final specification of requirements. A **Design** phase- in which a software solution to meet the requirements is designed. This may be a new design, or an extension of an earlier design. An **Implementation and Test** phase- when the software is coded, integrated and tested. A **Review** phase- in which the software is evaluated,

the current requirements are reviewed, and changes and additions to requirements proposed.

## 2.4 Task List

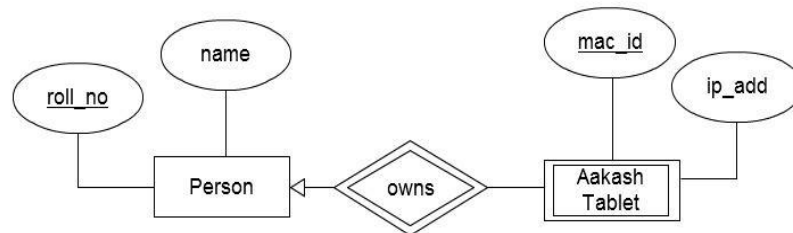
1. Establish Connection between Server and Client using WiFi
2. Audio Call Testing
3. Sending messages from Client to Server
4. Database Management and Database Connectivity
5. Parsing the Messages in the server and updating the Database
6. Establishing Peer to Peer Connection
7. Recording Audio in Tablet
8. Transmitting the live audio
9. Setting up of Calling Functionality between Clients
10. Audio Call Testing

## 3. DESIGNS AND IMPLEMENTATIONS

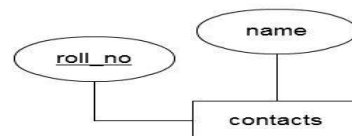
### 3.1 High Level Design Document

#### 3.1.1 E-R Diagram

Server side ER Diagram



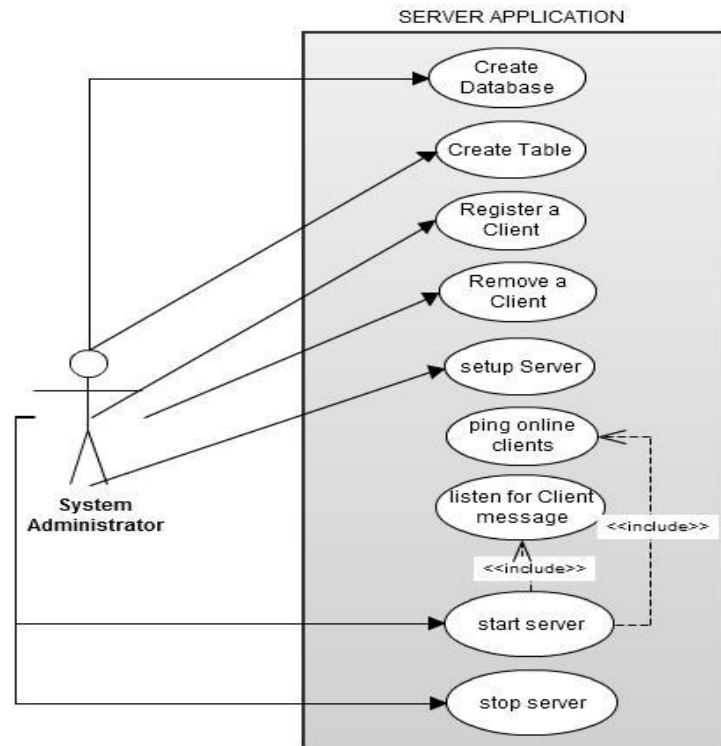
Client Side ER Diagram



### 3.1.2 Use Case Diagram

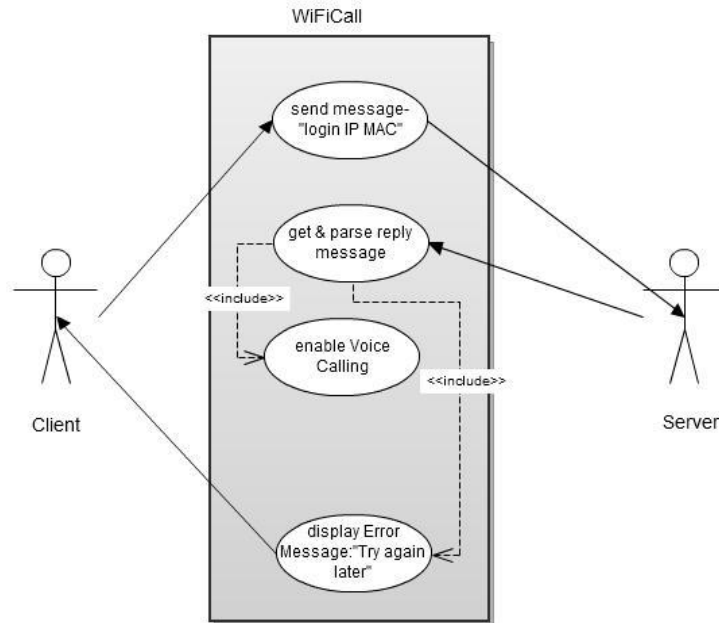
#### a) Server Application:

The system administrator takes the responsibility of setting-up the server at a location, like in the campus of an organization, and registering all the users that will be using the application via that server. Setting-up a server may involve creating the database and tables to be used by the server application. System administrator may also need to stop/restart the application in case of technical problems.



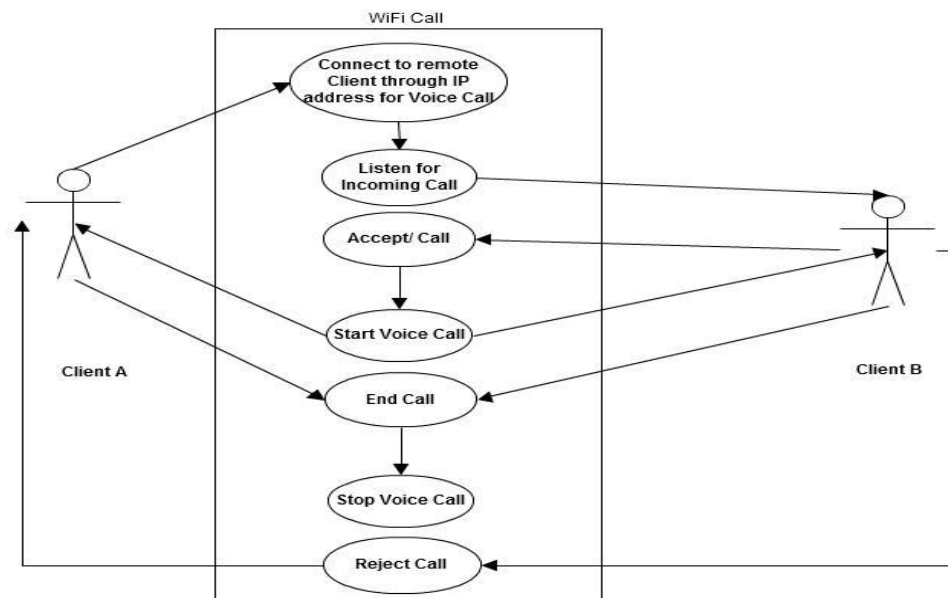
#### b) Client-Server Login:

- When a user starts the application he/she needs to first log-in at the server.
- If the server is running, the user will be logged-in and receive a confirmation from the server; or the user will be indicated that it is not registered.
- Else, an error message will be displayed indicating the connection problem. User may need to try again.



c) Successful Call between two users:

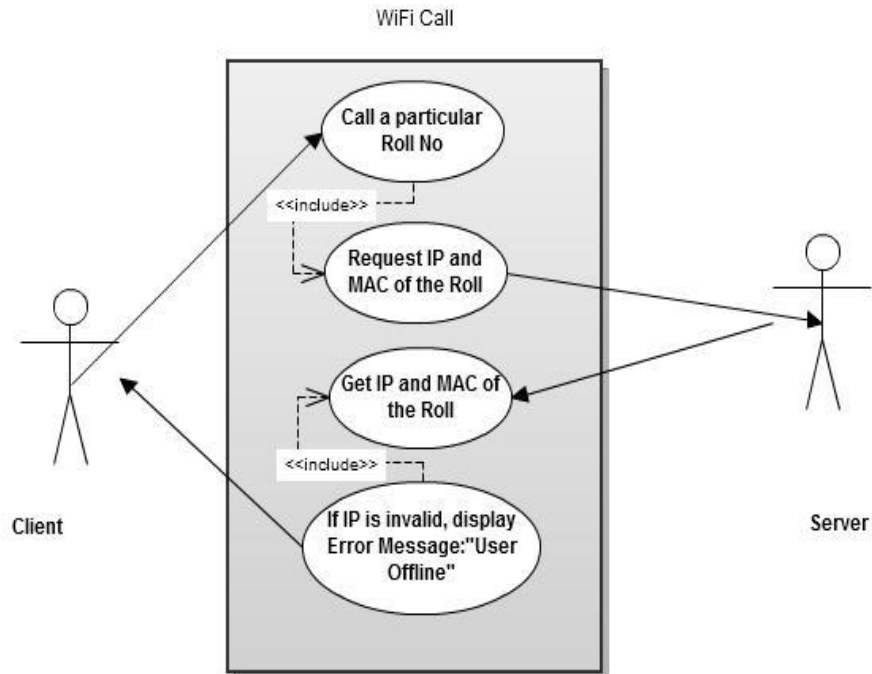
- The caller tries to establish a TCP connection with the remote user. The remote user receives an incoming call request.
- If the remote user accepts the call, voice call starts at both the ends. When either of them ends the call, the voice call stops at both the ends and also, the connection between them is terminated.
- If the remote user rejects the call, the TCP connection between them is terminated indicating to the caller that remote user has rejected the call..



d) Unsuccessful Call between two users:

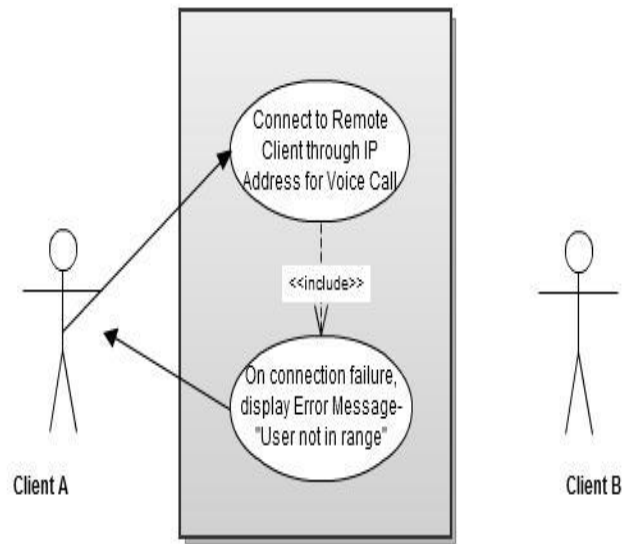
i) Remote User Offline

- If the IP address of the requested Roll No as received from the server is null (/invalid), it indicates that the remote client is currently not logged-in at the server. A message is displayed at the client side indicating this situation.



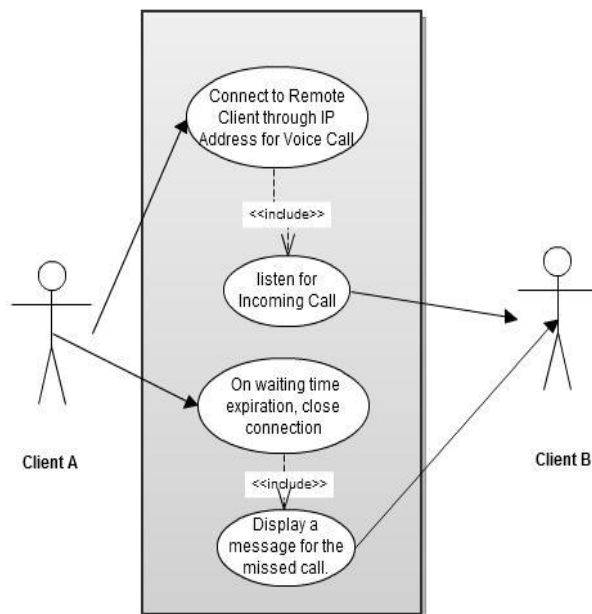
ii) Remote user not in range

- If the remote user is logged-in at server, but is not within the wifi range at that moment, the TCP connection attempt will fail. The caller will be notified by an error message: "User not in range".



iii) Remote user: Not Answering/Busy

- After establishing a connection with the remote user, if the caller does not receive any response to its call request from the other side, it may close the connection. A 'missed call' message will be displayed at the remote user's side.

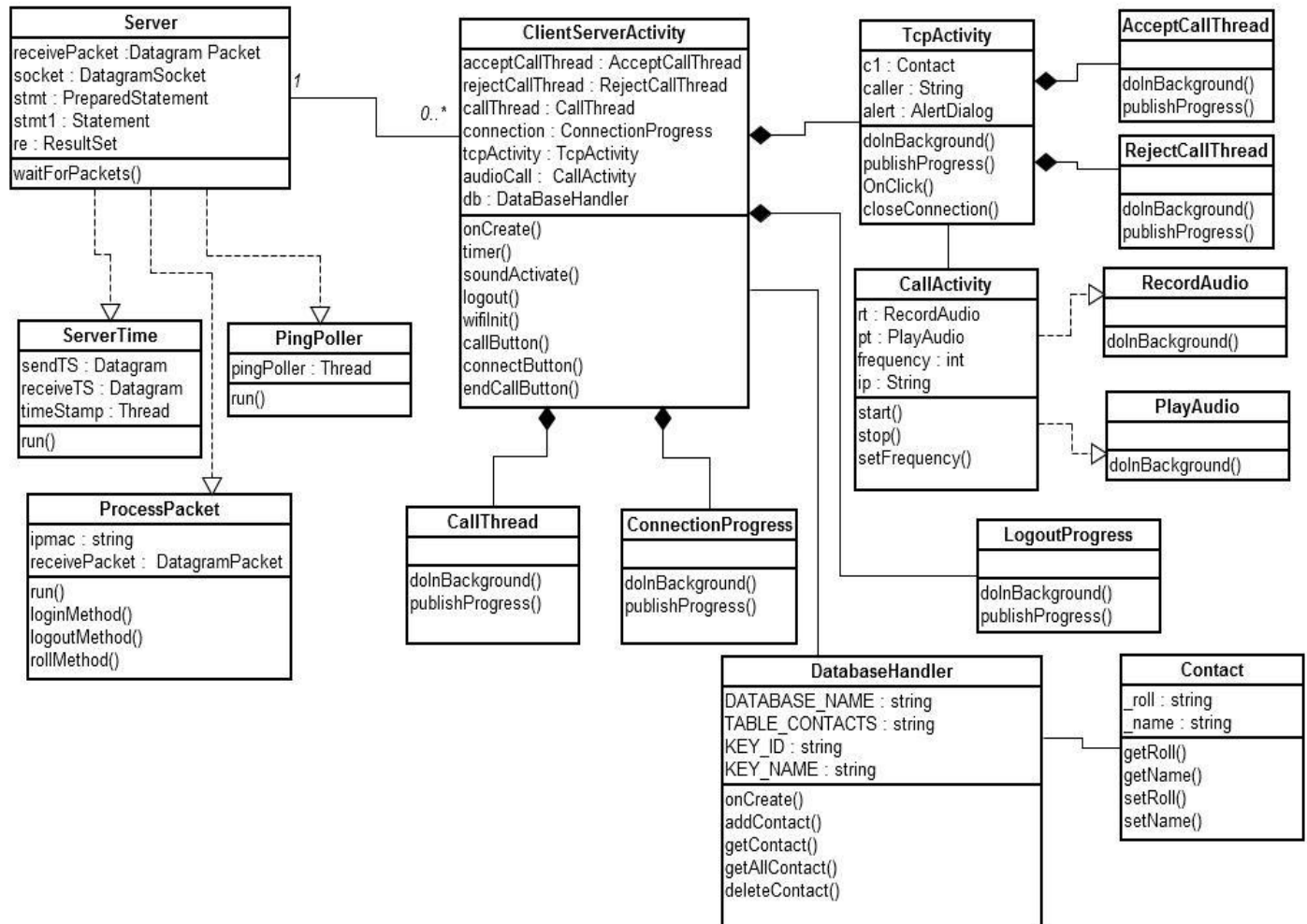


### 3.1.3 Class Diagram

The following diagram shows all the user defined classes functional in our applications:

#### PeerToPeer Voice

#### Communiaction Class Diagram



**Server:** It initiates the PingPoller and ServerTime threads. It continuously listens at port 6000 for all client requests. It creates a new ProcessPacket thread to process each received packet from a client.

**PingPoller:** This class pings each online client every 5 minutes. If it does not receive a reply it updates the database to mark the client as offline.

**ServerTime:** Send the current time of the server to the requesting clients.

ProcessPacket: It processes each received packet from the clients and responds to the corresponding client accordingly.

ClientServerActivity: It creates and starts an activity at the client side application. It opens a port to communicate with the server. It listens for clicks from the user at all the buttons and accordingly performs activity. For example: when client clicks “login” button, it creates an object of ConnectionProgress class. Similarly, on click on other buttons, the activity sends corresponding messages to the server or a remote client.

TcpActivity: It opens a ServerSocket at port 6000 which continuously listens for a TCP connection from a remote client. It receives and processes the various messages exchanged between two clients and opens and closes a TCP connection accordingly. This class also holds the responsibility to start the voice call at the caller side.

AcceptCallThread: This class accepts a TCP connection request from a remote client, starts its voice call and sends back an appropriate message to indicate acceptance to the remote user.

RejectCallThread: This class rejects a TCP connection request from another client and sends back an appropriate message.

CallActivity: It opens two ports to enable audio data transmission between two users. It also creates and executes object of classes RecordAudio and PlayAudio.

RecordAudio: It continuously records audio data from the user side and sends packets to the remote user during voice call.

PlayAudio: It continuously receives audio data packets from the remote user and plays the data at the user’s side during voice call.

CallThread: Gets the current time of the server, sends the “roll” message to the server, establishes a TCP connection with the remote client and waits for a response from other side.

ConnectionProgress: Get the current time of the server, send “login” message to the server and on successful login creates and executes an object of TcpActivity.

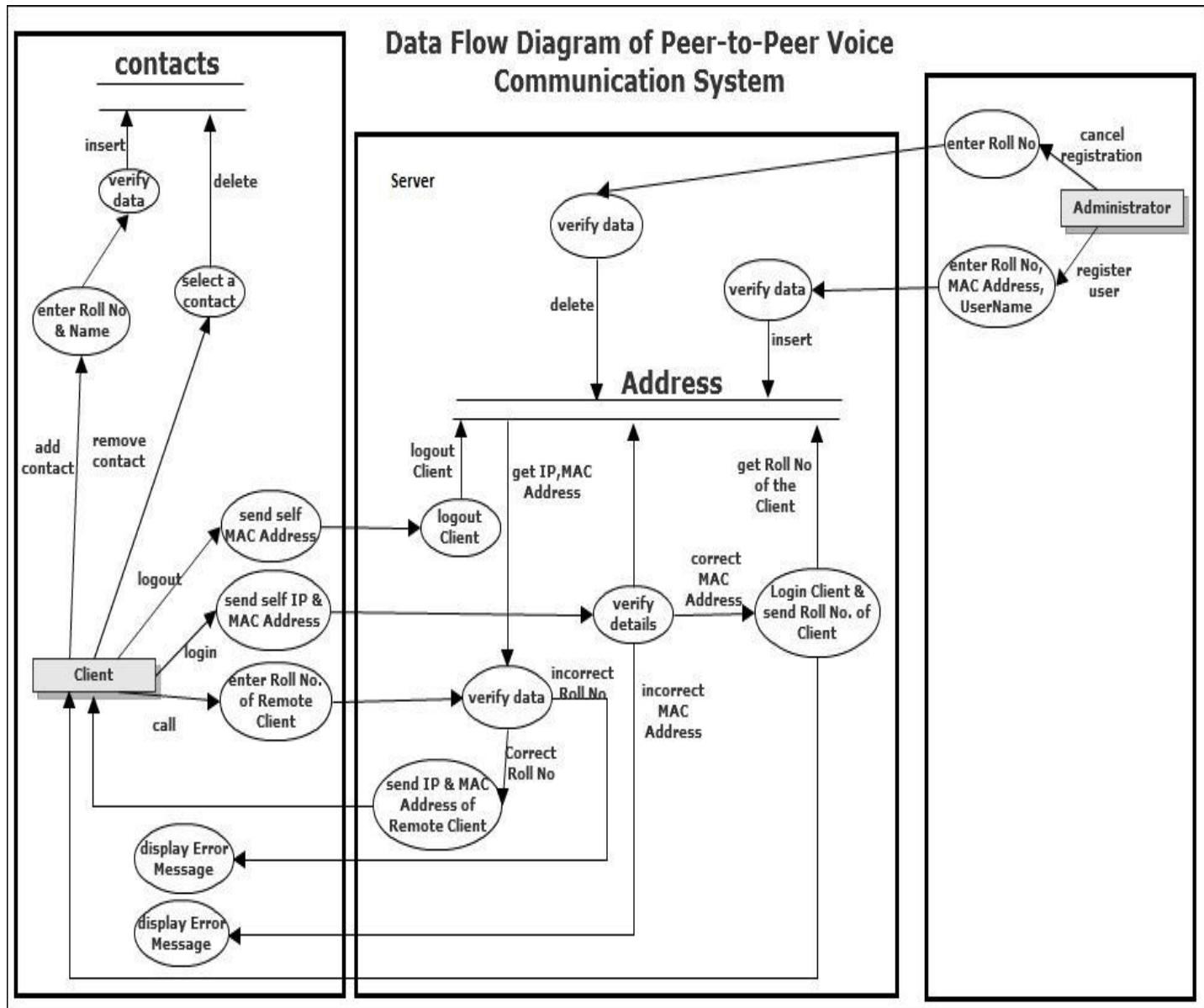
LogoutProgress: Get the current time of the server, send “logout” message to the server. It displays appropriate messages to indicate to the user about the status changes.

DatabaseHandler: This class updates the database which stores the various contacts in the client application.

Contact: This class helps client to update contact list. A contact list is a feature which stores Roll No and Names of other clients registered on same server.



### 3.1.4 Data Flow Diagram



## 3.2 Low Level Design Document

### 3.2.1 Algorithm & Flowchart:

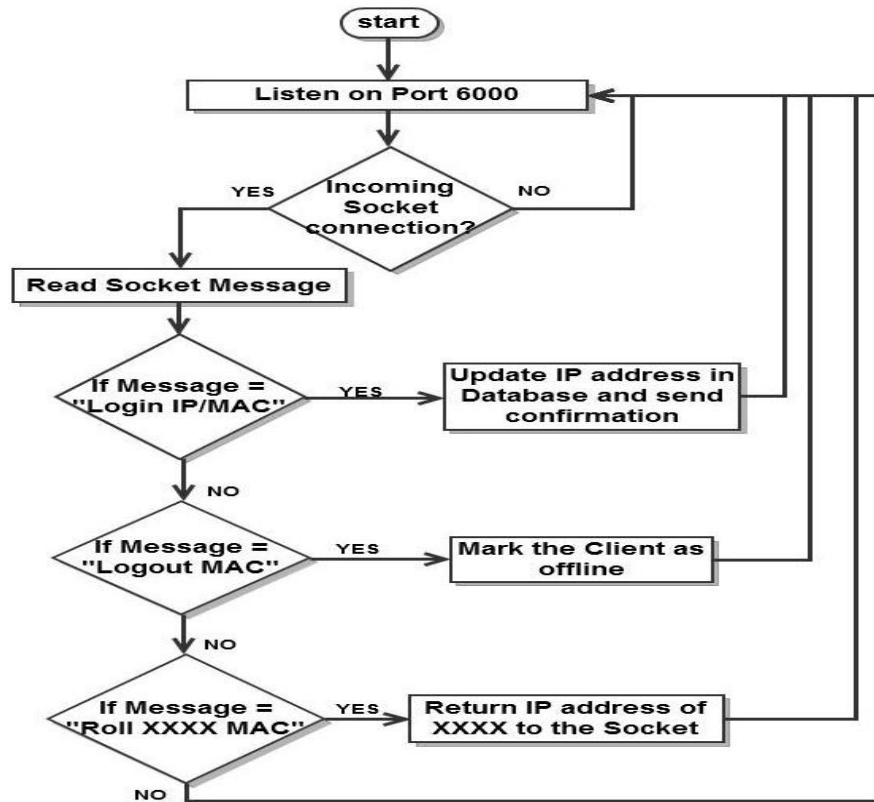
#### 1) Server Software:

1. Connect to the database which stores the IP and MAC address information of all registered clients.
2. Create a UDP socket to receive (login |logout |roll) messages from the clients.
3. Create a new thread. In this thread- (Server Time)
  - 3.1 create another UDP socket to receive timestamp requests from the clients.
  - 3.2 receive the "time" message.
  - 3.3 send back the current time of the server.
  - 3.4 go to 3.2
4. Create another thread. In this thread – (PingPoller)
  - 4.1 sleep for 5 minutes.
  - 4.2 get the IP addresses of all online users from the database.
  - 4.3 For all the IP addresses obtained:
    - Ping each IP address.
    - If reply is received, do nothing.
    - Else, update the database to mark the user as offline.
  - 4.4 go to 4.1
5. In the main thread:
  - 5.1 Receive the incoming message, MESSAGE
  - 5.2 If (timestamp of the MESSAGE) is **BEFORE** (server's current time) go to 5.1
  - 5.3 if MESSAGE="login I/P MAC", update the database to mark the user as online and send back client's unique roll no.
  - 5.4 else if MESSAGE="logout MAC", update the database to mark the user as offline.
  - 5.5 else if MESSAGE="roll <roll> MAC", if the user with received MAC address is online, send back the MAC and current IP address of the requested user.
  - 5.6 go to 5.1

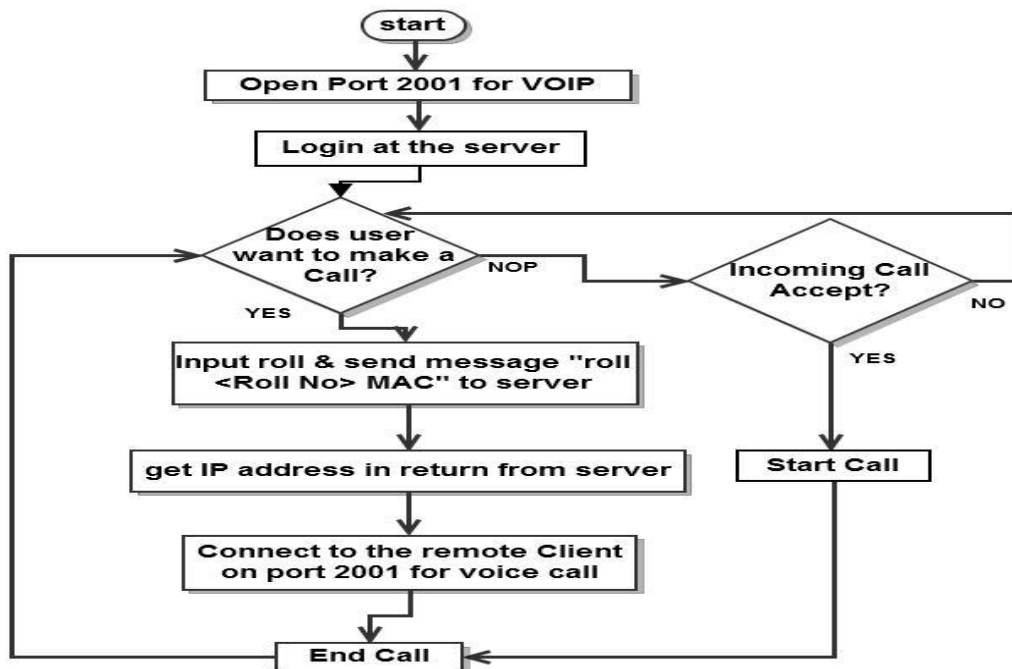
## 2) Client Application:

1. Create a UDP socket to communicate with the server.
2. Get the current time of server. Send message-“login I/P MAC” to login at the server.
3. If reply not received go to step 2.
4. Else, create 2 TCP sockets (to communicate with another online client)
5. If the user wants to make a call:
  - a. Send message-“roll <roll> MAC” to the server.
  - b. Receive the incoming message- MESSAGE
  - c. If MESSAGE=” IP MAC” and IP is valid
    - i. connect to the remote user
    - ii. Start call.
    - iii. End call
  - d. Go to 5
6. Else, an Incoming Call-
  - a. If “accept”,
    - i. Start call
    - ii. End call
  - b. Else if “reject”, reject call.
7. If done, go to 8  
Else go to 5.
8. Send message -“logout MAC” to the server.
9. Exit.

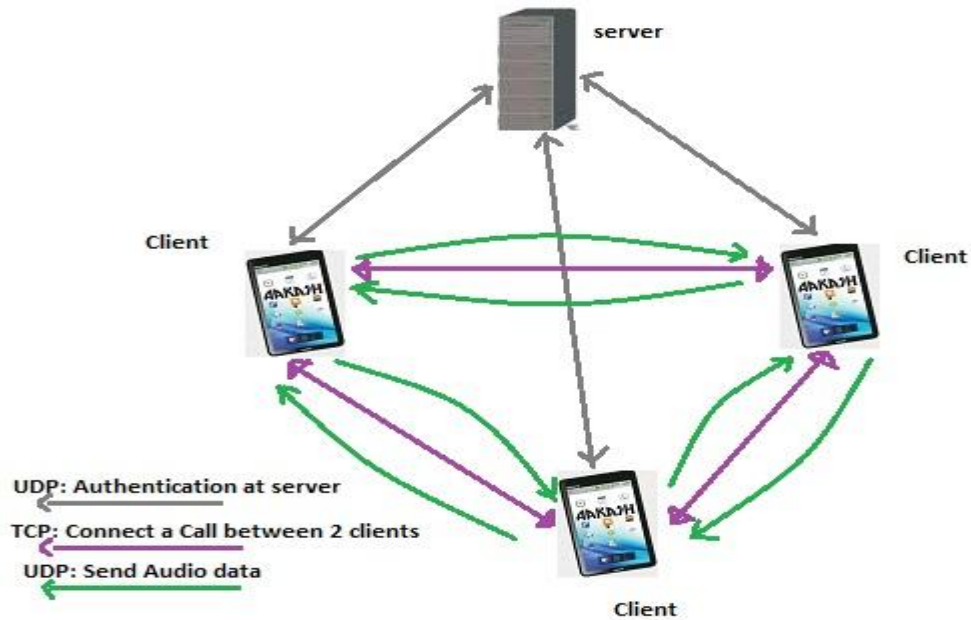
### 3) Server Flow-Chart:



### 4) Client Flow-Chart:



### 3.2.2 Interface design:



## 4. Challenges and their Solutions

- Voice Lag and Distortion
  - This was a major issue at first. When we recorded and sent the raw audio packets over the network, the other tablet which was receiving the packet and played the audio, the voice received was distorted and there was a considerable amount of lag.
  - Then after a lot of experimenting when we broke the audio packet in packet size of 512Bytes the problem was solved. As before the packet size was large so there was overhead of breaking the packets, so this may be a cause to the problem.
- Frequency of Sampling
  - At first we developed the audio module (record/play) in the HCL Tablet. The sampling rate was 44100Hz. Then when we tried this in Aakash Tablet the Audio Record object was not being initialized. This was a major problem.
  - Then after a lot of experimenting we saw that recording in Aakash tablet was to be done at 11025Hz. After this this problem was also solved.
- Aakash Audio Jack Issue
  - This was a major problem. No proper compatible ear piece is being found for the Aakash tablet. Due to this there is problem in the audio transmission though our application is working fine on other Android Devices.
  - This problem is yet to be solved.

## 5. SUMMARY & CONCLUSION

### 5.1 Summary

First we wrote the server side program and tested it with a tester program. After that we concentrated on recording and playing raw audio in PCM format in the Aakash Tablet. After this audio calling module was over we distributed the tasks and started working. The timestamp was included in the packet send to the server. The initiation of the call was done through TCP communication between the two tablets. The address book was also incorporated in the application, in which the user can store the names and roll number of his friends. Missed Call functionality was included and when the application is in background the user will be notified when there is an incoming call. The Server IP Address which will be set will be saved on the hardware, so that the user will not have to change it unless the servers address changes. It can work at different frequency

### 5.2 Further Enhancements

#### 5.2.1 Integrating video with audio

- transmitting live video data along with audio data between the two clients to enable video calls.

#### 4.2.2 Encoding Audio Packets before transmission

- In the current product raw audio data is transmitted between the communicating peers. Thus audio packets will be encoded to reduce the amount of data to be streamed, while maintaining the voice quality.

#### 4.2.3 Audio Conferencing

- This feature will be added to the product to enable more than 2 users to communicate simultaneously with each other.

#### 4.2.4 File transfer and sharing

- Adding this feature to the current product will allow a user to share with or send to/receive files from other users.

### 5.3 References and Bibliography

[1]Shawn Van Every, Pro Android Media, Apress 2009;167-172

[2] Deitel, Java How To Program, 4Th Edition, Prentice Hall

[3] Android Developer Website, <http://developer.android.com>

[4]SQLite Tutorial Website, <http://www.androidhive.info/2011/11/android-sqlite-database-tutorial/>

[5]Custom List View Website, <http://www.codeproject.com/Articles/183608/Android-Lists-ListActivity-and-ListView-II-Custom>

## CHAPTER 3

# USER MANUAL

---

### ***INDEX:***

<b>1</b>	<b>SERVER (DESKTOP).....</b>	<b>24</b>
<b>2</b>	<b>SERVER (ANDROID) .....</b>	<b>29</b>
<b>3</b>	<b>CLIENT (WIFI CALL) .....</b>	<b>32</b>

## ***A)Peer To Peer – SERVER (Desktop)***

### **1 Introduction**

“Peer To Peer – Server” authenticates the android devices using WiFi call application. You have to login into the server before making a call. At the same time you can logout from server whenever you wish to. Server will also exchange your device’s IP address with called person’s device.

### **2 Open Peer To Peer Server**

- Open Peer to Peer Server (Jar File). You will see a window as shown in Figure (1).

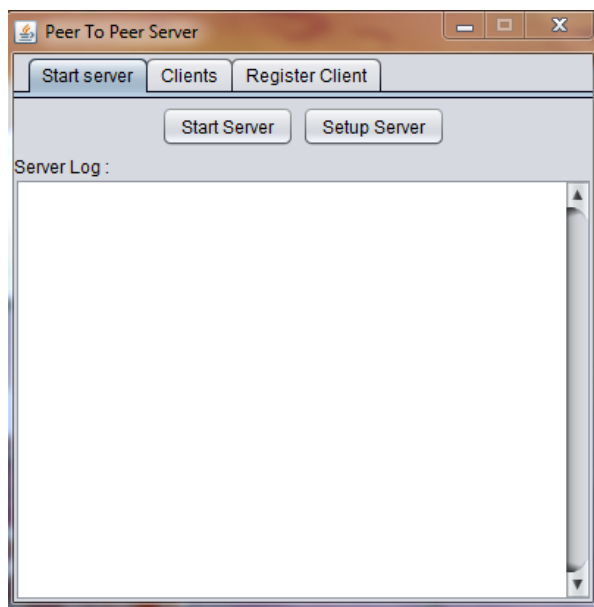


Figure (1)

- Click on “Setup Server” button as shown in Figure (2) to set host name, database name, username and password.

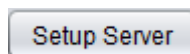


Figure (2)

- You will see another window as shown in Figure (3).

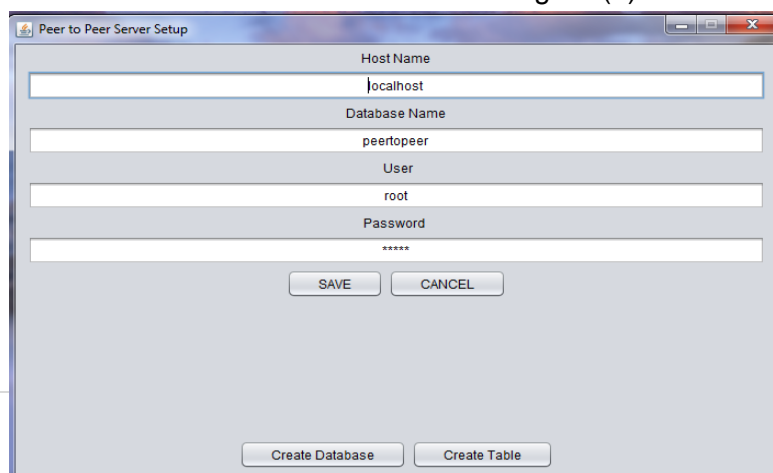




Figure (3)

- You can create a new database with “Create Database” button as shown in Figure(3). This will take the name from “Database Name” field to create the database. If such a database already exists it will show an error in “Server Log” otherwise a new database will be created if you have permissions to do so.
- At the same time you can create a new table in your database with “Create Table” button in Figure (3). This will create a new table named “address” in your database. If table already exists, it will give you an error otherwise it will be created if you have permissions to do so.

This window shows some default entries.

- i) Set the host name as “local host” if database is on your machine itself. If database is on a remote computer and you wish to access the database from that computer set the host name as its IP address.
- ii) This database has a table named “address” which has following columns in order : roll\_no, ip\_add, mac\_add, name. This table stores the information of all the registered clients. You can create a this database with “Create Database” button.

Alternatively you can also create your own database as explained below :

```
create database peertopeer;  
create table address  
(roll_no varchar(30),  
ip_add varchar(30),  
mac_add varchar(30),  
name varchar(30),  
primary key (roll_no, mac_add));
```

- iii) Set the username of your MySQL Client.
  - iv) Set the password of your MySQL Client.
- You can edit the entries according to your machine and click on “SAVE” button as shown in Figure (4).

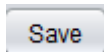
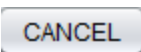


Figure (4)

- Click on “CANCEL” button as shown in Figure (5) to hide the setup window.



Figure(5)

### 3 **Starting/Stopping the Server**

- Click on “Start Server” button as shown in Figure (6) to start the server. Server uses two port addresses : 5000 and 6000. If “Server Log” shows “Address Already in Use” error try to free above two port addresses.

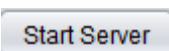


Figure (6)

- Once server is started, registered clients can login to the server. You can see all the login, roll, logout and error messages in “Server Log” as shown in Figure (7).

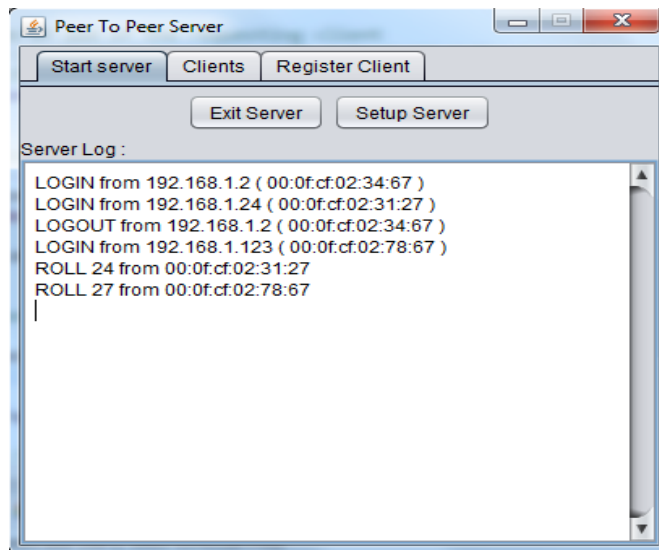
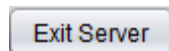


Figure (7)

- To stop and exit the server, click on “Exit Server” button as shown in Figure (8).



Figure(8)

#### 4 **Registered Clients**

- You can see all the online and offline user under the “Clients” tab as shown in Figure(9).
- The Online users are shown in green and the Offline are shown in gray.
- This tab is updated every 5 minutes.

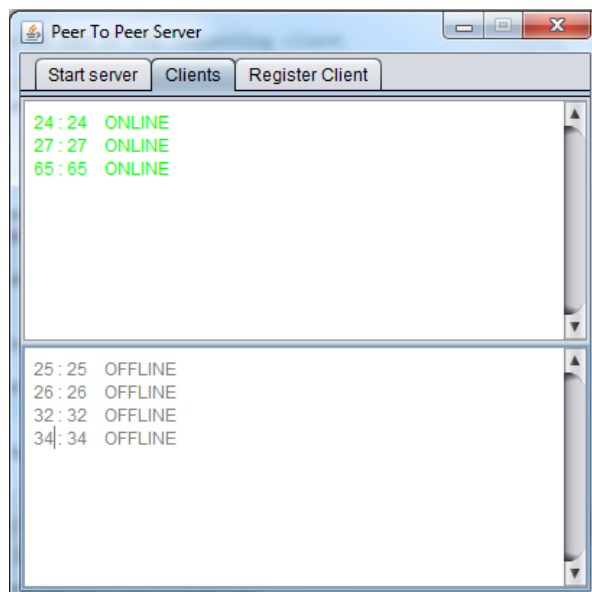
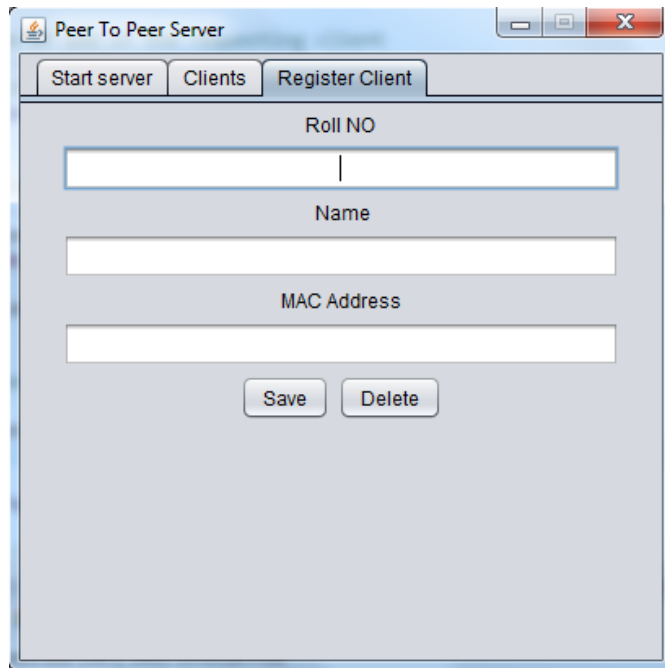


Figure (9)

## 5 Registering Users

- You can register new clients under “Register Clients” tab as shown in Figure (10).



The screenshot shows a window titled "Peer To Peer Server" with three tabs: "Start server", "Clients", and "Register Client". The "Register Client" tab is active. It contains three text input fields labeled "Roll NO", "Name", and "MAC Address". Below these fields are two buttons: "Save" and "Delete".

Figure (10)

- To Register a client into the database, click on “Save” button as shown in Figure (11).

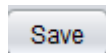
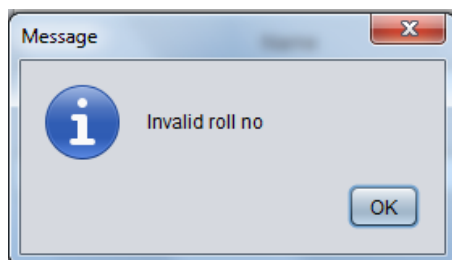


Figure (11)

- Roll no should have atleast one character and can include alphabets, numbers and underscore.
- Name should have atleast one character.
- MAC address should be in standard 6 bytes format with only hexadecimal numbers allowed.
- If above input format is not followed, you will be shown message dialog shown in Figure (12).



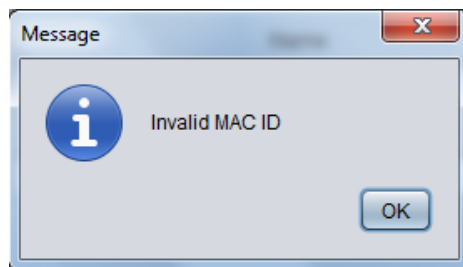
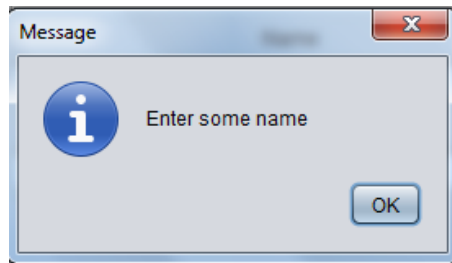


Figure (12)

- To remove a client from database, click on “Delete” button as shown in Figure (13). You have to enter only client’s roll number to remove it from database.



Figure (13)

- If user is not registered in database and you try to delete, you will see a message dialog as shown in Figure (14).

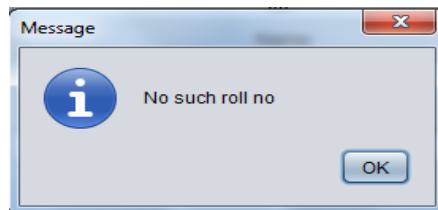


Figure (14)

# *Peer To Peer – SERVER (Android)*

## **1 Introduction**

“Peer To Peer – Server” authenticates the android devices using WiFi call application. You have to login into this server before making a call. At the same time you can logout from server whenever you wish to. Server will also exchange your device’s IP address with called person’s device.

## **2 Open Peer To Peer Server**

- When you switch ON the tablet you will see on the screen, the display similar as shown in the Figure(1) below.



Figure (1)

- Click on the launcher icon provided on the right hand side of the screen of the tablet as shown in the figure (2) below.

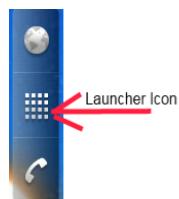


Figure (2)

- Then you will see on the screen all the application present in the tablet.
- Click on **WiFi Call Server** icon present in the tablet as shown in the figure (3) below.



Figure (3) WiFi Call Server

- The home screen will look like Figure(4)

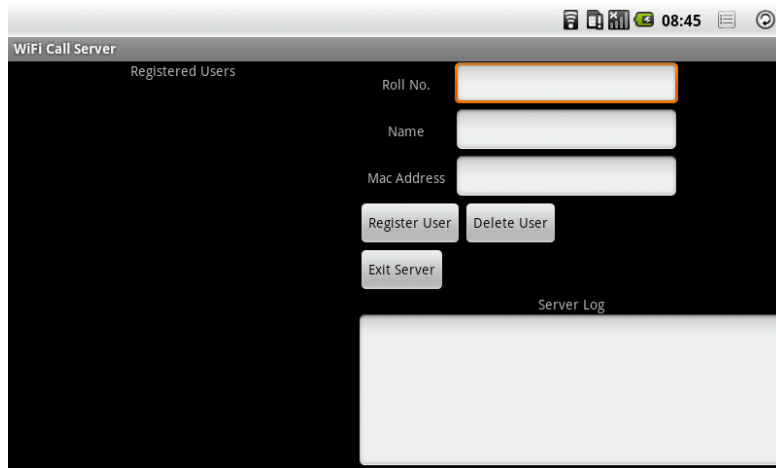


Figure (4)

### 3 Starting/Stopping the Server

- Click on “Start Server” button as shown in Figure (5) to start the server. Server uses two port addresses: 5000 and 6000. If server shows “Address Already in Use” error in Server Log, then try to free above two port addresses.

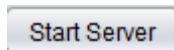


Figure (5)

- Once server is started, registered clients can login to the server. You can see all the login, roll, logout and error messages in Server Log as shown in Figure (6).

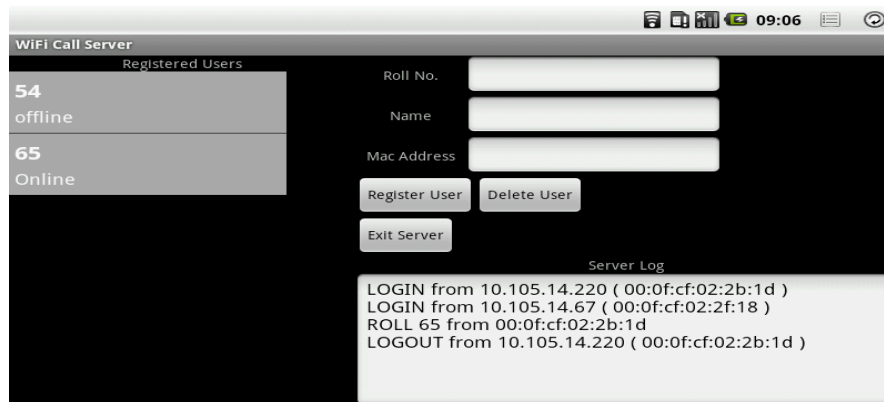
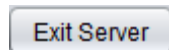


Figure (6)

- To stop and exit the server, click on “Exit Server” button as shown in Figure (7).



Figure(7)

### 4 Registered Clients

- You can see all the online and offline user under the Registered Users list as shown in Figure (6).
- This list is updated every 5 minutes.

### 5 Registering Users

- You can register new clients as shown in Figure (7).

Figure (7) shows a registration form with the following fields and values:

Field	Value
Roll No.	54
Name	54
Mac Address	45:76:ad:54:ae:44

Below the fields are two buttons: "Register User" and "Delete User".

Figure (7)

- To register a client into the database, click on "Register User" button as shown in Figure (7).
- Roll no should have at least one character and can include alphabets, numbers and underscore.
- Name should have at least one character.
- MAC address should be in standard 6 bytes format with only hexadecimal numbers allowed.
- If above input format is not followed, you will be shown message dialog shown in Figure (8).

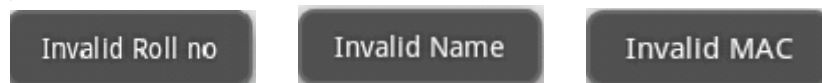


Figure (8)

- To remove a client from database, click on "Delete User" button as shown in Figure (9). You have to enter only client's roll number to remove it from database.

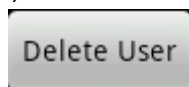


Figure (9)

- If user is not registered in database and you try to delete, you will see a message dialog as shown in Figure (10).

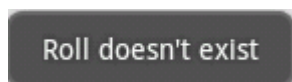


Figure (10)

## *Peer To Peer - Client*

### **1 Introduction**

- Through “Peer to Peer Voice Communication” audio call can be done between two tablets connected to some access points.

### **2 Open WiFi Call**

- When you switch ON the tablet you will see on the screen, the display similar as shown in the Figure(1) below.

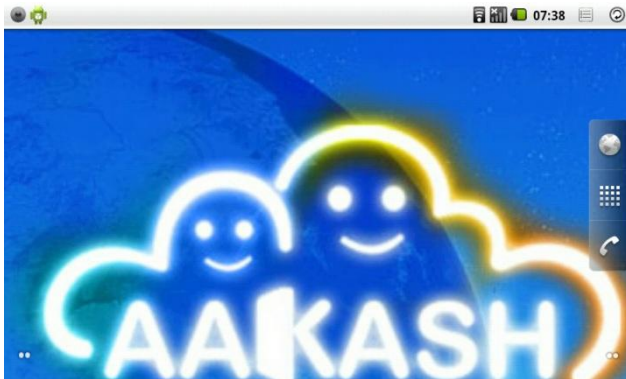


Figure (1)

- Click on the launcher icon provided on the right hand side of the screen of the tablet as shown in the figure (2) below.



Figure (2)

- Then you will see on the screen all the application present in the tablet.
- Click on **WiFi Call** icon present in the tablet as shown in the figure (3) below.



Figure (3): WiFi Call

### **3 Setting Up the Application**



### 3.1 Set the Server IP Address

- Click on the Settings Icon as in Figure (4).



Figure (4)

- You will see the following dialog as in Figure (5). It will have a default IP Address, if the application is run for the first time or the previous IP Address which was set.

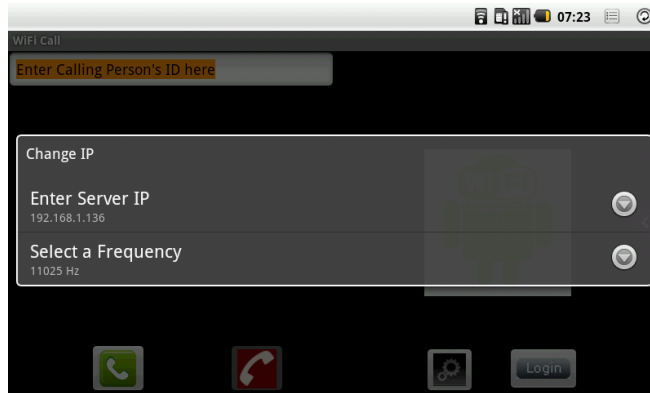


Figure (5)

- If you have to change the Server IP Address click on the Server IP Address. Next you will see the text box to enter the Server IP Address as in Figure (6).

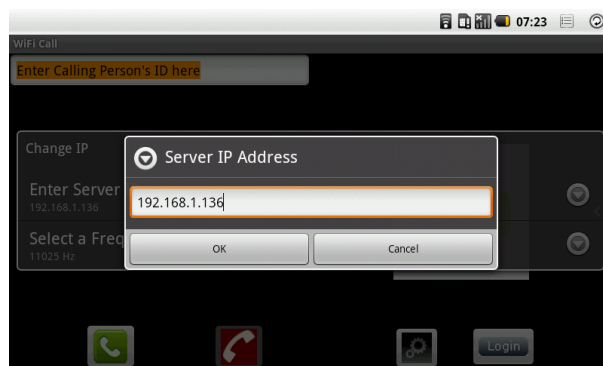


Figure (6)

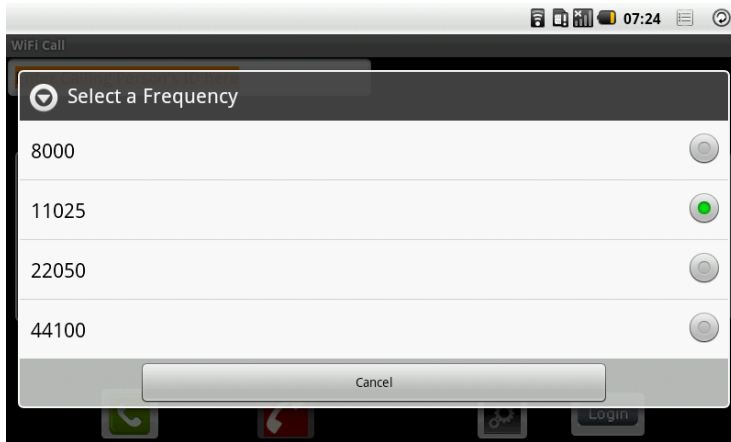
- Enter the desired Server IP Address as provided by the system administrator and click OK.

**NOTE** – Server is used for user authentication.

### 3.2 Set the Frequency of Audio Call

- Click on the Settings Icon as in Figure (4).

- You will see the following dialog as in Figure (5).It will have a default Frequency, if the application is run for the first time or the frequency which was entered the last time.
- If you have to change the frequency then click on the Frequency and you can choose from the given range of frequencies as in Figure (7).



Figure(7)

- NOTE : Choose 11025 Hz for all versions of Aakash tablet.

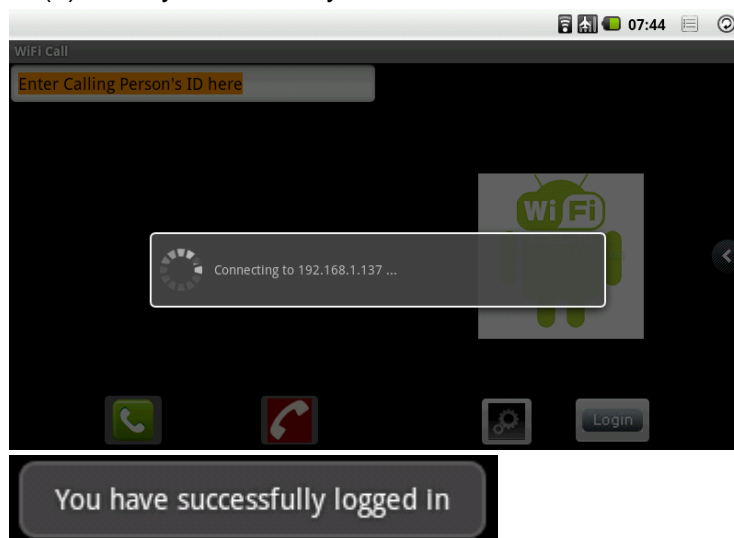
### 3.3 Connecting with the Server

- After setting proper parameters , click on the Login button as in Figure (8).



Figure (8)

- If server is running on your network you will get connected with it. On successful login, the login button will be replaced with a "logout" button and you will be notified as in Figure (9). Now you are ready to receive and make calls.



Figure(9)

- If something is wrong, wait for 10seconds and you will be notified of server status as in Figure (10).

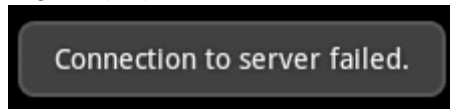


Figure (10)

### 3.4 Logging out at the Server

- To logout click on the logout button.

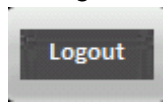
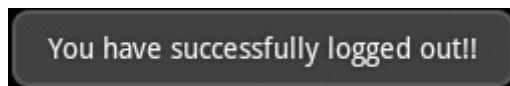


Figure (11)

- On successful logout, the button will be replaced with a “login” button and you will be notified as below:

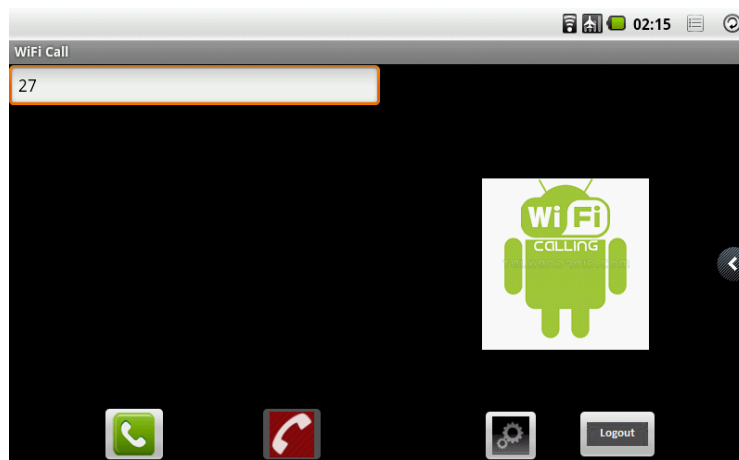


You will need to login again to receive and make calls.

## 4 Calling and Receiving

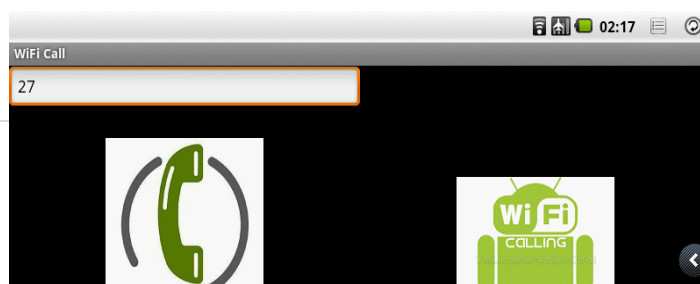
### 4.1 Making a Call

- Enter the Roll Number of the user in the text box as shown in Figure (12).



Figure(12)

Then press the call button and your call will start and the screen will be as in Figure (12).



Figure(13)

- When called person accepts your call your screen will be as in Figure (14).

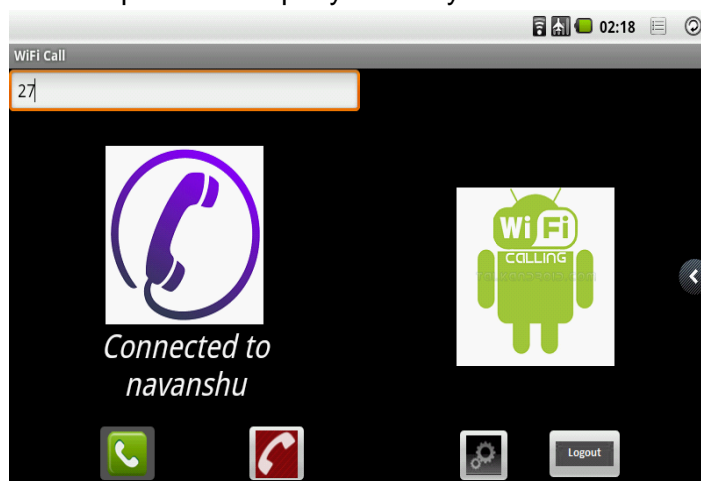
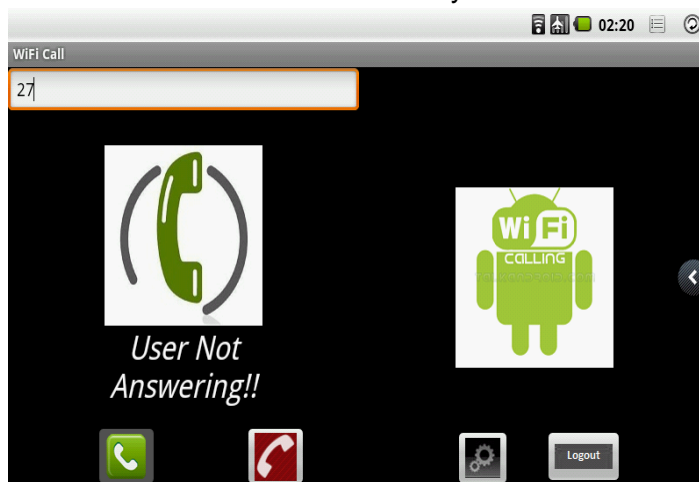


Figure (14)

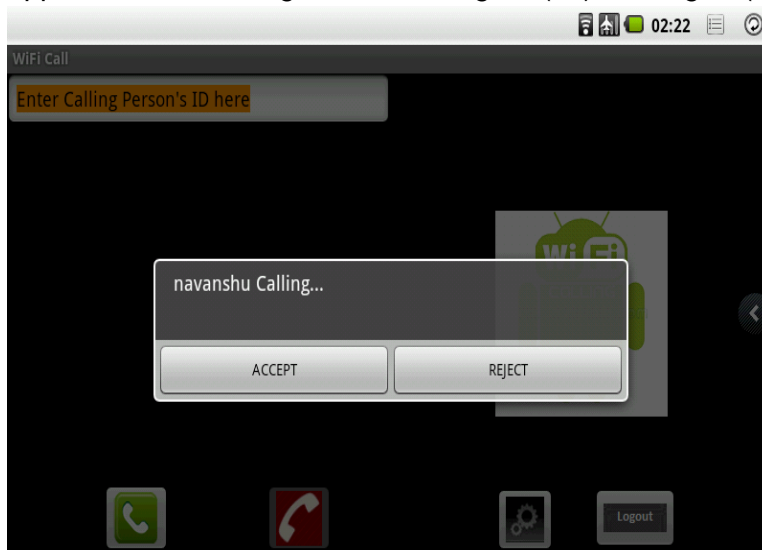
- End the call by pressing the end call button.
- If the user will not receive the call you will be notified as in Figure(15).



Figure(15)

## 4.2 Receiving a Call

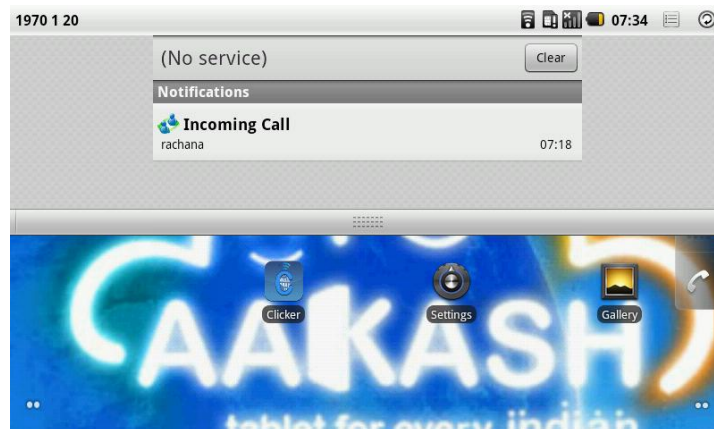
- When there is an incoming call, you will be notified as in Figure (16) (even if the application is in background as in Figure (17) and Figure(18))



Figure(16)

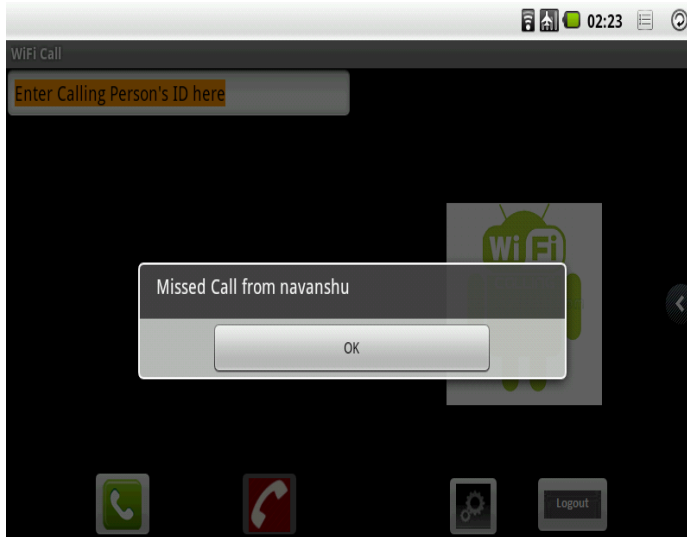


Figure(17)



Figure(18)

- To accept the call press the accept button as in Figure.
- To reject the call press the reject button.
- If you are unable to attend the call , you will be notified with a missed call as in Figure(19).

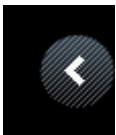


Figure(19)

## 5 Managing the Contact Book

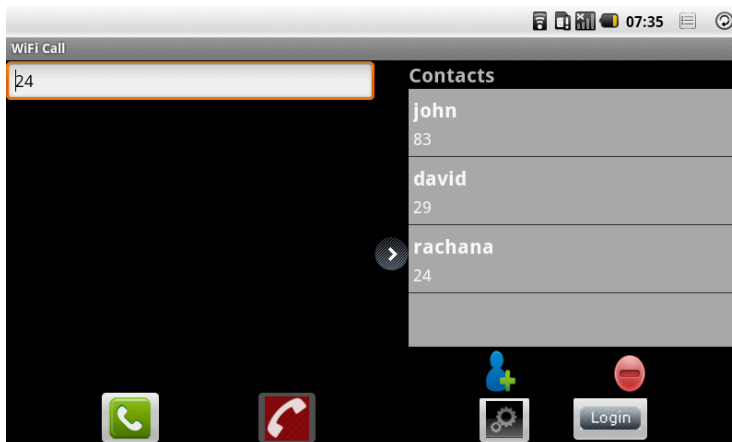
### 5.1 Opening and Closing the Contact book

- Click on the Slide Button to open the Contact Book as in Figure(20).



Figure(20)

- Click on the Slide Button to close the Contact Book back as in Figure(21).



Figure(21)

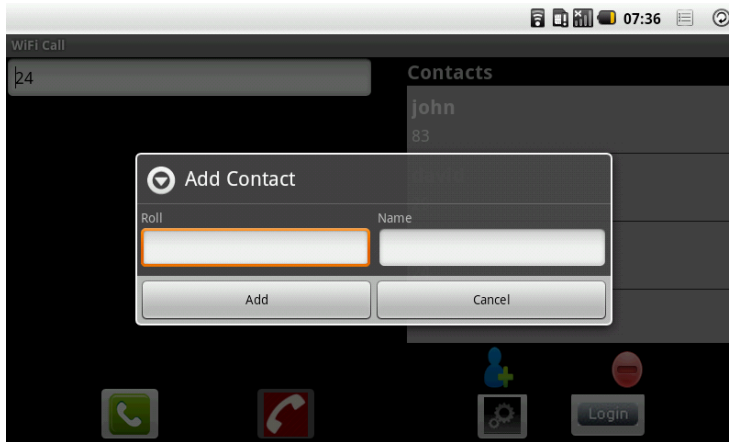
### 5.2 Adding Contacts

- Click on the Add Contact Button as in Figure(22).



Figure(22)

- Now enter the Roll and Name in the Dialog Box and press OK as in Figure(23).

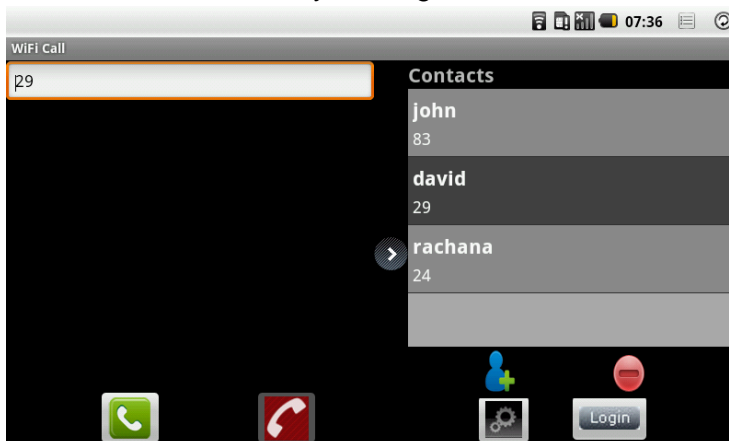


Figure(23)

- The Contact will be shown in the list.

### 5.3 Removing Contacts

- First Select a Contact by clicking over it in the list as in the Figure(24).



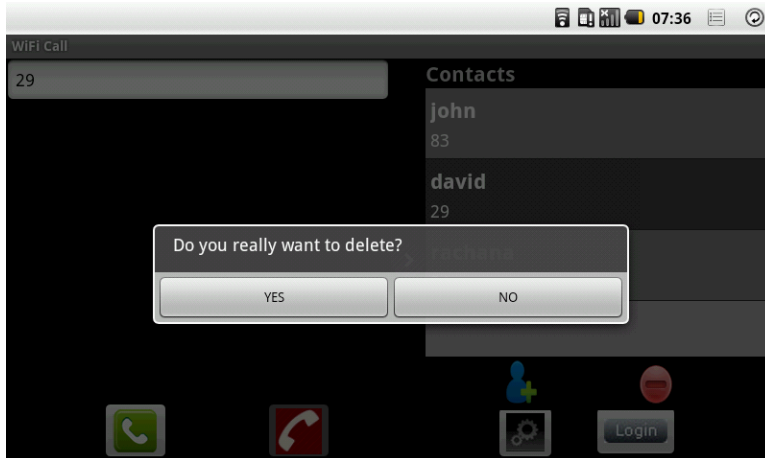
Figure(24)

- Then press the remove button as shown in Figure(25).



Figure(25)

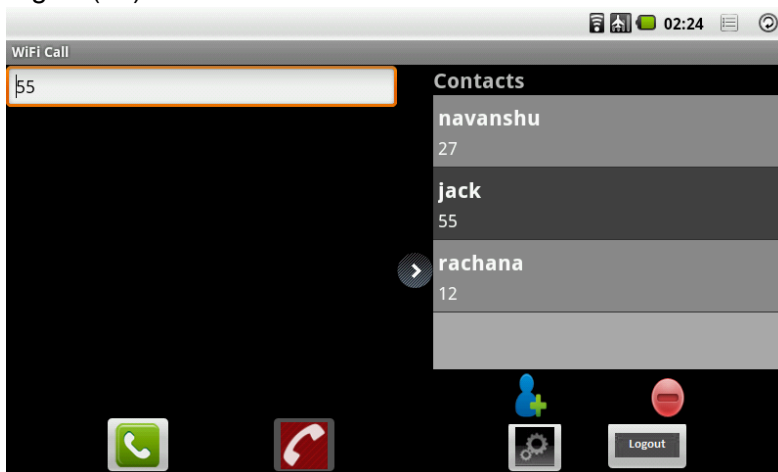
- Then press OK button if you want to remove the contact as in Figure(26).



Figure(26)

## 5.4 Making Calls from Contacts

- Select the User to whom you want to call by clicking over it on the contact list as in Figure(27).



Figure(27)

- Then press the call button .