

BLUETOOTH CHAT APPLICATION

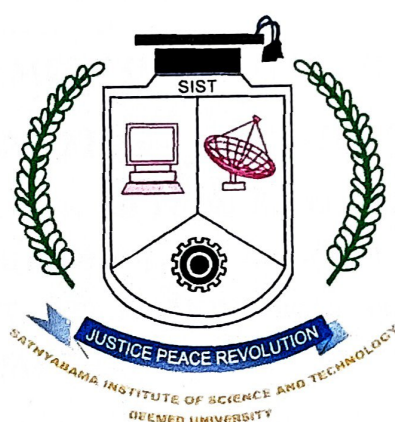
Submitted in partial fulfilment of the required for the award of

Bachelor of Computer Science

by

V.AKSHAYA PRIYA

(REGISTER NO 40290009)



DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF SCIENCE AND HUMANITIES

SATHYABAMA

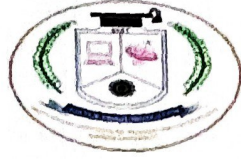
INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE

JEPPIAR NAGAR, RAJIV GANDHI SALAI, CHENNAI 600 199

MAY – 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC I Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of
V.AKSHAYA PRIYA(40290009) who carried out the project entitled
"BLUETOOTH CHAT APPLICATION" under our supervision from
JANUARY 2023 TO MAY 2023.

Shamili.v
Internal Guide

Ms .V. SHAMILI M.Sc.

Rekha
6/5/22
DEAN AND HEAD OF THE DEPARTMENT

Dr. REKHA CHAKRAVARTHI M.E., Ph.D.

Submitted for Viva voce examination held on

8.05.2023

6/5/2023
Internal Examiner

8/5/23
External Examiner

DECLARATION

I am, **V.AKSHAYA PRIYA (40290009)**, here by declare that the Project Report entitled "**BLUETOOTH CHAT APPLICATION**" done by us under the guidance of **MS.V.SHAMILI M.Sc.,(Assistant Professor)** Department of Computer Science at **SATHYABAMA INSTITUTE SCIENCE AND TECHNOLOGY**, Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai-600 119 is submitted in partial fulfillment of the requirements for the award of Bachelor of Science degree in Computer Science.

DATE: 08-5-2023

V. AKSHAYA PRIYA

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. REKHA CHAKRAVARTHI M.E., Ph.D.** **Dean of Arts and Science** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my project guide **MS V.SHAMILI M.Sc.**, Assistant Professor Department of Computer Science for their valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff member of the **Department of Computer Science** who were helpful in many ways for the completion of the project

V.AKSHAYA PRIYA

ABSTRACT

The main aim of this project is to analyse, design, build and test Bluetooth chat software. The software has been developed as an Interactive and collaborative learning aid. Short Messaging Service (SMS) offers the same services as Bluetooth Chat for a free. This project is to analyse Bluetooth chat software and technology. The analysis include market and users need in order to be able to fabricate the Bluetooth chat application and make it available for users as a helpful utility. As this application is depending on Bluetooth, the goal is to attempts to revive the Bluetooth usage again and make it usable continuously and daily as the Internet.

TABLE CONTENT

CHAPTER NO	TITLE	PAGE NO
1	INTRODUCTION	8
1.1	DOMAIN INTRODUCTION	8
1.2	OBJECTIVE OF THE PROJECT	9
1.3	PROJECT INTRODUCTION	9
2	LITERTURE SURVEY	10
2.1	REVIEW ON LITERATURE WORK	10
2.2	EXISTING SYSTEM	12
2.3	PROPOSED SYSTEM	12
3	METHODOLOGY	13
3.1	PROJECT SPECIFICATION	13
3.1.1	REQUIREMENTS	13
3.2	ARCHITECTURE DIAGRAM	14
3.3	DEVELOPMENT PHASES	14
3.3.1	ANALYSIS PHASE	14
3.3.2	DESIGN PHASE	15
3.3.2.1	USE CASE DIAGRAM	15
3.4.3	IMPLEMENTATION PHASE	16
3.4.4	TESTING PHASE	16
4	CONCLUSION AND FUTURE WORK	17
4.1	CONCLUSION	17
4.2	FUTURE WORK	17
5	APPENDICES	18

5.1	SAMPLE CODE	18
5.2	SCREEN SHOTS	30
6	REFERENCES	33

CHAPTER 1

INTRODUCTION

1.1 DOMAIN INTRODUCTION

Bluetooth is a network technology that connects mobile devices wirelessly over a short-range to form a personal area network (PAN). They use short-wavelength, ultra-high frequency (UHF) radio waves within the range 2.400 to 2.485 GHz, for wireless communications.

Bluetooth Usage

Usage of Bluetooth can be broadly categorized into three areas –

Access Points for Data and Voice – Real-time voice and data transmissions are provided by Bluetooth by connecting portable and stationary network devices wirelessly.

Cable replacement – Bluetooth replaces the need for a large number of wires and cables of wired networks. The connections can be made instantly and are retained even when the devices are not within range. The range of the devices is typically 10m. However, the range can be extended by using amplifiers.

Ad hoc networking – Ad hoc networks are formed impromptu by the network devices bypassing the need for a central access point like a router. Bluetooth networks are ad hoc in nature since a Bluetooth enabled device can form an instant connection with another Bluetooth enabled device as soon as it comes into range.

Bluetooth Applications

Some of the common applications of Bluetooth are

- In laptops, notebooks and wireless PCs
- In mobile phones and PDAs (personal digital assistant).
- In printers.
- In wireless headsets.
- In wireless PANs (personal area networks) and even LANs (local area networks)
- To transfer data files, videos, and images and MP3 or MP4.
- In wireless peripheral devices like mouse and keyboards.

1.2 OBJECTIVE OF THE PROJECT

The goal of this project is to create an Bluetooth application in Android. This application would take the advantage of the wide spread of the Android operating system via varieties of devices. It is concerned with solving some problems of communicating freely, securely, silently and within small range.

1.3 PROJECT INTRODUCTION

The main purpose of this project is to deliver a useful Bluetooth chatting application that targets the wide spread Android operating system powered devices. The most important feature to be fulfilled is that application should share or exchange text messages among the Bluetooth chat application users. This app is a Bluetooth chat application. This will allow the users to chat to each other via Bluetooth. The aim of this app is to revive the usage of Bluetooth again. This app will enable users to chat even if you don't have internet service on your phone. This type of app doesn't require you to have any wifi-connection. All it needs is two Bluetooth-compatible Android devices in a range of about 50 feet from each other.

CHAPTER 2

LITERATURE SURVEY

2.1 REVIEW ON LITERATURE WORK

TITLE 1: Bluetooth Smart Support for 6LoBTLE

YEAR: 2015

AUTHOR: Joseph Decuir

ABSTRACT:

The Bluetooth Low-Energy transport, trademarked Bluetooth Smart, is now a market fact [1]. Worldwide sales of Bluetooth Smart devices and Bluetooth Ready mobile devices are measured in several billion units per year. The two most exciting new markets are wearable devices [2], [3] and home networking [4].

TITLE 2: A Bluetooth radio in 0.18-/spl mu/m CMOS

YEAR: 2003

AUTHOR: P. van Zeijl;J.-W.T. Eikenbroek;P.-P. Vervoort;S. Setty;J. Tangenherg;G. Shipton;E. Kooistra;I.C. Keekstra;D. Belot;K. Visser;E. Bosma;S.C. Blaakmeer

ABSTRACT:

This paper describes the results of an implementation of a Bluetooth radio in a 0.18-/spl mu/m CMOS process. A low-IF image-reject conversion architecture is used for the receiver. The transmitter uses direct IQ-upconversion. The VCO runs at 4.8-5.0 GHz, thus facilitating the generation of 0/spl deg/ and 90/spl deg/ signals for both the receiver and transmitter. By using an inductor-less LNA and the extensive use of mismatch simulations, the smallest silicon area for a Bluetooth radio implementation so far can be reached: 5.5 mm/sup 2/. The transceiver consumes 30 mA in receive mode and 35 mA in transmit mode from a 2.5 to 3.0-V power supply.

TITLE 3: Security for Bluetooth enabled devices using BlipTrack Bluetooth detector

YEAR: 2015

AUTHOR: MonuKumar;B. K. Gupta

ABSTRACT:

As the Bluetooth technology is growing there is a need of secure communication and to keep there data secret where Bluetooth is used as primary option. There are many attacks possible on Bluetooth technology and associated devices such as denial of service attack, passive eavesdropping and active eavesdropping, man-in-the-middle attacks, message modification, and resource misappropriation. So it is a challenge to detect the attacker. This paper proposed an idea for those Bluetooth enabled devices which uses Secure Simple Pairing (SSP) association models not having Strong security against passive eavesdropping.

TITLE 4: Bluetooth_Performance Analysis in Personal Area Network (PAN)

YEAR: 2006

AUTHOR: Rozeha A. Rashid;RohaizaYusoff

ABSTRACT:

Bluetooth is a telecommunications industry specification operating in an unlicensed frequency band of 2.4 GHz that describes how mobile phones, computers, and personal digital assistants (PDAs) can be easily interconnected using a short-range wireless connection. In this study, a personal area network (PAN) was set up using three personal computers, two Bluetooth universal serial bus dongles, one serial Bluetooth module and also a Bluetooth mobile phone. A graphical user interface (GUI) using Visual Basic 6.0 was built to display the system accessibility such as file and system properties, transferring text files, chatting program and network detection by Bluetooth's service discovery protocol (SDP).

2.2 EXISTING SYSTEM

In existing Bluetooth Chat Android system, you have to first set up the connection via wires and all and then you can share your details or other information with each other. Existing Bluetooth Chat Android system has many drawbacks such as you have to first carry forward all the user id and other details to connect your device with other and also sometimes connection in between breakdown due to internet speed which also leads to the disabling of sharing of files from one device system to another device system.

DISADVANTAGES:

- Low data security.
- Short range communication.
- Can lose connection very easily.

2.3 PROPOSED SYSTEM

In the proposed Bluetooth Chat Android system, the sending up of all the text and documents can be done via Bluetooth and it does not require internet connection. Proposed system saves a lot of time and efforts for making connections via internet as well Proposed system is far a lot better because in proposed system, there is no chance of losing information, data or and file or document due to losing up of internet connection.

ADVANTAGES:

- No internet connection required.
- Less chance of losing data.
- High transfer rate.

FEATURES OF ANDROID BLUETOOTH CHAT APP

- 1) There are two main screens in this app.
- 2) The first screen lets you turn off or turn on your Bluetooth device.
- 3) This screen also allows you to chat with each other.
- 4) There is an edit text at the bottom which lets you send messages.
- 5) The message will be shown on this screen as well.
- 6) The second screen will show all the paired devices and you can connect to any of them
- 7) Once you are connected, you can start chatting

CHAPTER 3

METHODOLOGY

3.1 PROJECT SPECIFICATION

The Blue Chat application's specifications are not complex. First the software must be installable on any device operated by the Android Operating System and support all screen sizes and resolutions to cover a wide range of users and devices. The application then must meet its main purpose or role of exchanging text messages and files among more than one user; this must be achieved or else the software would be useless or unusable. Then the most important criteria is to satisfy the customers' need by making the software easy to use, the extra features to be entertaining and usable for them and to implement the application with no errors and out of bugs to avoid continuous crashing with a satisfying performance whether.

3.1.1 Requirements

- **Java** - Java is a programming language used to write the logic of our application.
 - **XML** – XML is a markup language used to design the application.
 - **Android Studio** – Android Studio is a platform that provides us with some tools to create our applications.
1. Activity_main.xml is the first file in our dashboard, which contains all the UI code.
 2. In the next file, there is a MainActivity.java. That file will handle the actions of the above file.
 3. As a next step, we need to create different XML files to handle our module's UI. Our java files will contain their code.

- Operating System : Windows 7 , 8, 10, 11(64 bit)
- Software : Java and Android Studio
- Tools : Android Studio

3.2 ARCHITECTURE DIAGRAM

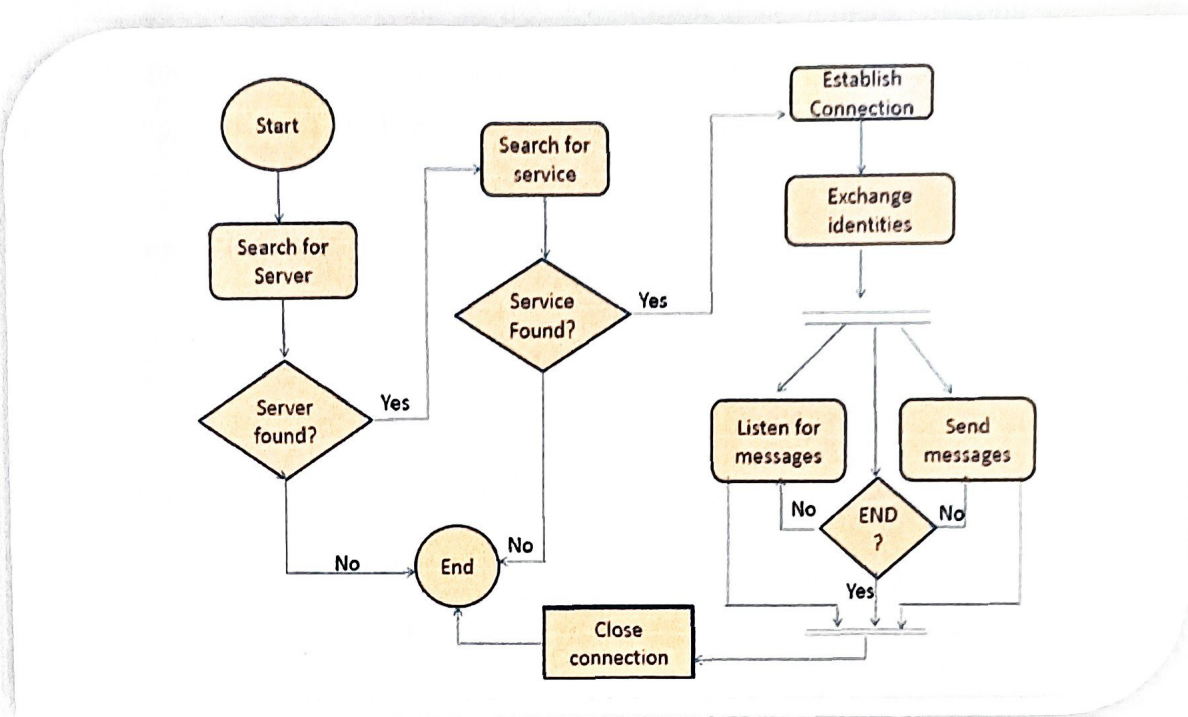


Fig 3.2.1

3.3 DEVELOPMENT PHASES

A well-organized development plan must be followed:

3.3.1 Analysis Phase

The first part of the project is the analysis process. Analysis processes is where research is done to start building the application based on firm, reasonable and logic basis. Also look back if there are past attempts trying to achieve the same idea of the application and get advantage of them. The Bluetooth Chat application can be used by any one unless user cannot or do not know how to interact with Android operating system device itself. Analysis means to get everything just ready for the next step even if it might get a little advance.

3.3.2 Design Phase

The interface design is a milestone in this application development. Starting by simple, fabricating simple buttons, text boxes and warning messages. Next, preparing the protocols needed for the application to use Bluetooth and Client/Server codes.

3.3.2.1 Use Case Diagram

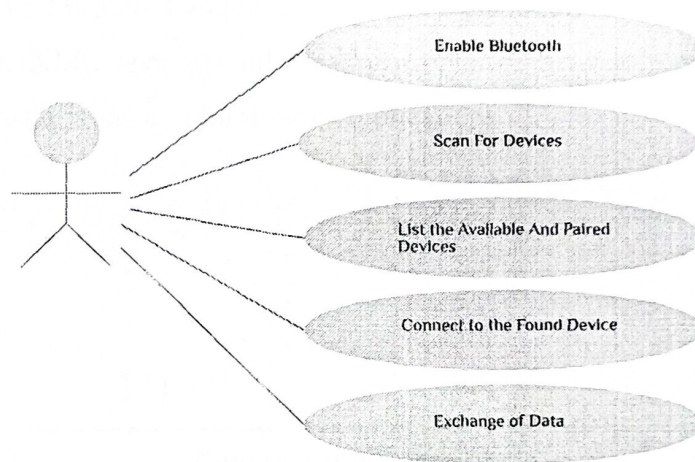


Fig 3.3.2.1.1

1. **Enable Bluetooth** – Request permission for switching on the Bluetooth on your device.
2. **Scan for Devices** – Enable the scanning option for scanning nearby devices.
3. **List the Available and Paired Devices** - After enabling scan devices the list of available and paired devices will be found.
4. **Connect to the Found Device** - He next step is, connect to the found device in the paired and available devices.
5. **Exchange of Data** – After successfully connected to the found device ,the user will start chatting and exchanging text messages with other users.

3.3.3 Implementation Phase

- **Java** - The coding of the application would be worked upon to reach the needed functionalities needed to be implemented in the software. Java is the best suited programming language for developing Android applications. Java is a well spread language with a lot of support and help documentation and resources. All the needed tools are free and easily installed. These tools are recommended by Google and are free
- **XML** - Extensible Markup Language (XML) lets you define and store data in a shareable manner. XML supports information exchange between computer systems such as websites, databases, and third-party applications.

3.3.4 Testing Phase

Use case	Functionality	Working
Enable Bluetooth	Request Permission for Enable Bluetooth	yes
Scan for Devices	Scan for nearby devices	yes
List the Available and Paired Devices	Determine if there are available devices foe chatting	yes
Connect to the Found Device	Accepts incoming chat request	yes
Exchange of Data	Send and exchange text message among users	yes

CHAPTER 4

CONCLUSION AND FUTURE WORK

4.1 CONCLUSION

This application provides all the features that are required when there is no internet connection, can prove helpful in collaborate learning, Easy to communicate, Sending and receiving messages in real time, help to save your internet data, Engaging messaging features. Messaging has become a part of our everyday lives in part due to its convenience for real-time chat communication and simple-to-use functionality. These messaging and real-time chat applications play an important role in how the world interacts today, due to their immediacy and vast capabilities. Hence, after going through the design and architecture of android as well as Bluetooth we can conclude that the implementation of Bluetooth chat application can be done easily which can be prove to be very useful to the android users.

4.2 FUTURE WORK

With the knowledge I have gained by developing this application, I am confident that in the future I can make the application more effectively by adding these services.

- Extending this application by providing Authorization service.
- Creating Database and maintaining users.
- Increasing the effectiveness of the application by providing Voice Chat.
- Extending it to Web Support

CHAPTER 5

APPENDICES

5.1 SAMPLE CODE

Main Activity.java

```
package com.example.bluetoothchatapp;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import android.Manifest;
import android.app.AlertDialog;
import android.bluetooth.BluetoothAdapter;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.Toast;
public class MainActivity extends AppCompatActivity {
    private Context context;
    private BluetoothAdapter bluetoothAdapter;
    private ChatUtils chatUtils;
    private ListView listMainChat;
    private EditText edCreateMessage;
    private Button btnSendMessage;
    private ArrayAdapter<String> adapterMainChat;

    private final int LOCATION_PERMISSION_REQUEST = 101;
    private final int SELECT_DEVICE = 102;
    public static final int MESSAGE_STATE_CHANGED = 0;
```

```

public static final int MESSAGE_READ = 1;
public static final int MESSAGE_WRITE = 2;
public static final int MESSAGE_DEVICE_NAME = 3;
public static final int MESSAGE_TOAST = 4;
public static final String DEVICE_NAME = "deviceName";
public static final String TOAST = "toast";
private String connectedDevice;

private final Handler handler = new Handler(new Handler.Callback() {
    @Override
    public boolean handleMessage(@NonNull Message message) {
        switch (message.what) {
            case MESSAGE_STATE_CHANGED:
                switch (message.arg1) {
                    case ChatUtils.STATE_NONE:
                        setState("Not Connected");
                        break;
                    case ChatUtils.STATE_LISTEN:
                        setState("Not connected");
                        break;
                    case ChatUtils.STATE_CONNECTING:
                        setState("Connecting...");
                        break;
                    case ChatUtils.STATE_CONNECTED:
                        setState("Connected:" + connectedDevice);
                        break;
                }
                break;
            case MESSAGE_WRITE:
                byte[] buffer1 = (byte[]) message.obj;
                String outputBuffer = new String(buffer1);
                adapterMainChat.add("Me: " + outputBuffer);
                break;
            case MESSAGE_READ:
                byte[] buffer = (byte[]) message.obj;
                String inputBuffer = new String(buffer, 0, message.arg1);
                adapterMainChat.add(connectedDevice + ": " + inputBuffer);
                break;
            case MESSAGE_DEVICE_NAME:
                connectedDevice = message.getData().getString(DEVICE_NAME);
                Toast.makeText(context, connectedDevice, Toast.LENGTH_SHORT).show();
                break;
            case MESSAGE_TOAST:
                Toast.makeText(context, message.getData().getString(TOAST),
                    Toast.LENGTH_SHORT).show();
                break;
        }
    }
});

```

```

    }
    return false;
}
});
private void setState(CharSequence subTitle) {
    getSupportActionBar().setSubtitle(subTitle);
}
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    context = this;
    init();
    initBluetooth();
    chatUtils = new ChatUtils(context, handler);
}
private void init() {
    listMainChat = findViewById(R.id.List_conversation);
    edCreateMessage = findViewById(R.id.ed_enter_message);
    btnSendMessage = findViewById(R.id.btn_send_msg);
    adapterMainChat = new ArrayAdapter<String>(context, R.layout.message_layout);
    listMainChat.setAdapter(adapterMainChat);
    btnSendMessage.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            String message = edCreateMessage.getText().toString();
            if (!message.isEmpty()) {
                edCreateMessage.setText("");
                chatUtils.write(message.getBytes());
            }
        }
    });
}
private void initBluetooth() {
    bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
    if (bluetoothAdapter == null) {
        Toast.makeText(context, "No bluetooth found", Toast.LENGTH_SHORT).show();
    }
}
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menu_main_activity, menu);
    return super.onCreateOptionsMenu(menu);
}
@Override
public boolean onOptionsItemSelected(MenuItem item) {

```

```

switch (item.getItemId()) {
    case R.id.menu_search_devices:
        checkPermissions();
        return true;
    case R.id.menu_enable_bluetooth:
        enableBluetooth();
        return true;
    default:
        return super.onOptionsItemSelected(item);
}
}

private void checkPermissions() {
    if (ContextCompat.checkSelfPermission(context,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions(MainActivity.this, new
String[]{Manifest.permission.ACCESS_FINE_LOCATION},
LOCATION_PERMISSION_REQUEST);
    } else {
        Intent intent = new Intent(context, DeviceListActivity.class);
        startActivityForResult(intent, SELECT_DEVICE);
    }
}

@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    if (requestCode == SELECT_DEVICE && resultCode == RESULT_OK) {
        String address = data.getStringExtra("deviceAddress");
        chatUtils.connect(bluetoothAdapter.getRemoteDevice(address));
    }
    super.onActivityResult(requestCode, resultCode, data);
}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
@NonNull int[] grantResults) {
    if (requestCode == LOCATION_PERMISSION_REQUEST) {
        if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
            Intent intent = new Intent(context, DeviceListActivity.class);
            startActivityForResult(intent, SELECT_DEVICE);
        } else {
            new AlertDialog.Builder(context)
                .setCancelable(false)
                .setMessage("Location permission is required.\n Please grant")
                .setPositiveButton("Grant", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialogInterface, int i) {

```

```

        checkPermissions();
    }
})
.setNegativeButton("Deny", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        MainActivity.this.finish();
    }
}).show();
}
} else {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
}
}
private void enableBluetooth() {
    if (!bluetoothAdapter.isEnabled()) {
        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.BLUETOOTH_CONNECT) !=
PackageManager.PERMISSION_GRANTED) {
            return;
        }
        bluetoothAdapter.enable();
    }
    if (bluetoothAdapter.getScanMode() !=
BluetoothAdapter.SCAN_MODE_CONNECTABLE_DISCOVERABLE) {
        Intent discoveryIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_DISCOVERABLE);
discoveryIntent.putExtra(BluetoothAdapter.EXTRA_DISCOVERABLE_DURATION, 300);
startActivity(discoveryIntent);
    }
}
@Override
protected void onDestroy() {
    super.onDestroy();
    if (chatUtils != null) {
        chatUtils.stop();
    }
}
}
}

```

Activity main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <ListView
        android:id="@+id/List_conversation"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginBottom="60dp"
        android:background="#efefef" />

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true">

        <EditText
            android:id="@+id/ed_enter_message"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginEnd="100dp" />

        <Button
            android:id="@+id/btn_send_msg"
            android:layout_width="100dp"
            android:layout_height="wrap_content"
            android:layout_alignParentEnd="true"
            android:text="send" />

    </RelativeLayout>

</RelativeLayout>
```

Chatutils.java

```
package com.example.bluetoothchatapp;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothServerSocket;
import android.bluetooth.BluetoothSocket;
import android.content.Context;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.UUID;

public class ChatUtils {
    private Context context;
    private final Handler handler;
    private BluetoothAdapter bluetoothAdapter;
    private ConnectThread connectThread;
    private AcceptThread acceptThread;
    private ConnectedThread connectedThread;
    private final UUID APP_UUID = UUID.fromString("fa87c0d0-afac-11de-8a39-0800200c9a66");
    private final String APP_NAME = "BluetoothChatApp";
    public static final int STATE_NONE = 0;
    public static final int STATE_LISTEN = 1;
    public static final int STATE_CONNECTING = 2;
    public static final int STATE_CONNECTED = 3;
    private int state;
    public ChatUtils(Context context, Handler handler) {
        this.context = context;
        this.handler = handler;

        state = STATE_NONE;
        bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
    }
    public int getState() {
        return state;
    }
    public synchronized void setState(int state) {
        this.state = state;
    }
}
```

```

        handler.obtainMessage(MainActivity.MESSAGE_STATE_CHANGED, state, -
1).sendToTarget();
    }
    private synchronized void start() {
        if (connectThread != null) {
            connectThread.cancel();
            connectThread = null;
        }
        if (acceptThread == null) {
            acceptThread = new AcceptThread();
            acceptThread.start();
        }
        if (connectedThread != null) {
            connectedThread.cancel();
            connectedThread = null;
        }
        setState(STATE_LISTEN);
    }
    public synchronized void stop() {
        if (connectThread != null) {
            connectThread.cancel();
            connectThread = null;
        }
        if (acceptThread != null) {
            acceptThread.cancel();
            acceptThread = null;
        }
        if (connectedThread != null) {
            connectedThread.cancel();
            connectedThread = null;
        }

        setState(STATE_NONE);
    }
    public void connect(BluetoothDevice device) {
        if (state == STATE_CONNECTING) {
            connectThread.cancel();
            connectThread = null;

            connectThread = new ConnectThread(device);
            connectThread.start();

            if (connectedThread != null) {
                connectedThread.cancel();
                connectedThread = null;
            }
        }
    }

```

```

        setState(STATE_CONNECTING);
    }
    public void write(byte[] buffer) {
        ConnectedThread connThread;
        synchronized (this) {
            if (state != STATE_CONNECTED) {
                return;
            }
            connThread = connectedThread;
        }
        connThread.write(buffer);
    }
    private class AcceptThread extends Thread {
        private BluetoothServerSocket serverSocket;
        public AcceptThread() {
            BluetoothServerSocket tmp = null;
            try {
                tmp = bluetoothAdapter.listenUsingRfcommWithServiceRecord(APP_NAME,
APP_UUID);
            } catch (IOException e) {
                Log.e("Accept->Constructor", e.toString());
            }

            serverSocket = tmp;
        }
        public void run() {
            BluetoothSocket socket = null;
            try {
                socket = serverSocket.accept();
            } catch (IOException e) {
                Log.e("Accept->Run", e.toString());
            }
            try {
                serverSocket.close();
            } catch (IOException e1) {
                Log.e("Accept->Close", e1.toString());
            }
        }
        if (socket != null) {
            switch (state) {
                case STATE_LISTEN:
                case STATE_CONNECTING:
                    connected(socket, socket.getRemoteDevice());
                    break;
                case STATE_NONE:
                case STATE_CONNECTED:
                    try {
                        socket.close();
                    }

```

```

        } catch (IOException e) {
            Log.e("Accept->CloseSocket", e.toString());
        }
        break;
    }
}
}
public void cancel() {
    try {
        serverSocket.close();
    } catch (IOException e) {
        Log.e("Accept->CloseServer", e.toString());
    }
}
}

private class ConnectThread extends Thread {
    private final BluetoothSocket socket;
    private final BluetoothDevice device;
    public ConnectThread(BluetoothDevice device) {
        this.device = device;
        BluetoothSocket tmp = null;
        try {
            tmp = device.createRfcommSocketToServiceRecord(APP_UUID);
        } catch (IOException e) {
            Log.e("Connect->Constructor", e.toString());
        }
        socket = tmp;
    }
    public void run() {
        try {
            socket.connect();
        } catch (IOException e) {
            Log.e("Connect->Run", e.toString());
            try {
                socket.close();
            } catch (IOException e1) {
                Log.e("Connect->CloseSocket", e1.toString());
            }
            connectionFailed();
            return;
        }
    }
    synchronized (ChatUtils.this) {
        connectThread = null;
    }
    connected(socket, device);
}

```

```

    }
    public void cancel() {
        try {
            socket.close();
        } catch (IOException e) {
            Log.e("Connect->Cancel", e.toString());
        }
    }
}

```

```

private class ConnectedThread extends Thread {
    private final BluetoothSocket socket;
    private final InputStream inputStream;
    private final OutputStream outputStream
    public ConnectedThread(BluetoothSocket socket) {
        this.socket = socket
        InputStream tmpIn = null;
        OutputStream tmpOut = null;

        try {
            tmpIn = socket.getInputStream();
            tmpOut = socket.getOutputStream();
        } catch (IOException e) {

        }

        inputStream = tmpIn;
        outputStream = tmpOut;
    }

    public void run() {
        byte[] buffer = new byte[1024];
        int bytes;
        try {
            bytes = inputStream.read(buffer);
            handler.obtainMessage(MainActivity.MESSAGE_READ, bytes, -1,
buffer).sendToTarget();
        } catch (IOException e) {
            connectionLost();
        }
    }

    public void write(byte[] buffer) {
        try {
            outputStream.write(buffer);
            handler.obtainMessage(MainActivity.MESSAGE_WRITE, -1, -1,
buffer).sendToTarget();
        } catch (IOException e) {
        }
    }
}

```

```

public void cancel() {
    try {
        socket.close();
    } catch (IOException e) {}
}

private void connectionLost() {
    Message message = handler.obtainMessage(MainActivity.MESSAGE_TOAST);
    Bundle bundle = new Bundle();
    bundle.putString(MainActivity.TOAST, "Connection Lost");
    message.setData(bundle);
    handler.sendMessage(message);
    ChatUtils.this.start();
}

private synchronized void connectionFailed() {
    Message message = handler.obtainMessage(MainActivity.MESSAGE_TOAST);
    Bundle bundle = new Bundle();
    bundle.putString(MainActivity.TOAST, "Cant connect to the device");
    message.setData(bundle);
    handler.sendMessage(message);
    ChatUtils.this.start();
}

private synchronized void connected(BluetoothSocket socket, BluetoothDevice device) {
    if (connectThread != null) {
        connectThread.cancel();
        connectThread = null;
    }
    if (connectedThread != null) {
        connectedThread.cancel();
        connectedThread = null;
    }
    connectedThread = new ConnectedThread(socket);
    connectedThread.start();
    Message message =
handler.obtainMessage(MainActivity.MESSAGE_DEVICE_NAME);
    Bundle bundle = new Bundle();
    bundle.putString(MainActivity.DEVICE_NAME, device.getName());
    message.setData(bundle);
    handler.sendMessage(message);
    setState(STATE_CONNECTED);
}
}

```

5.2 SCREEN SHOTS

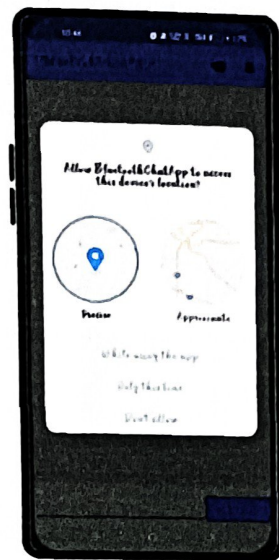


Fig 5.2.1 Permission to access device location

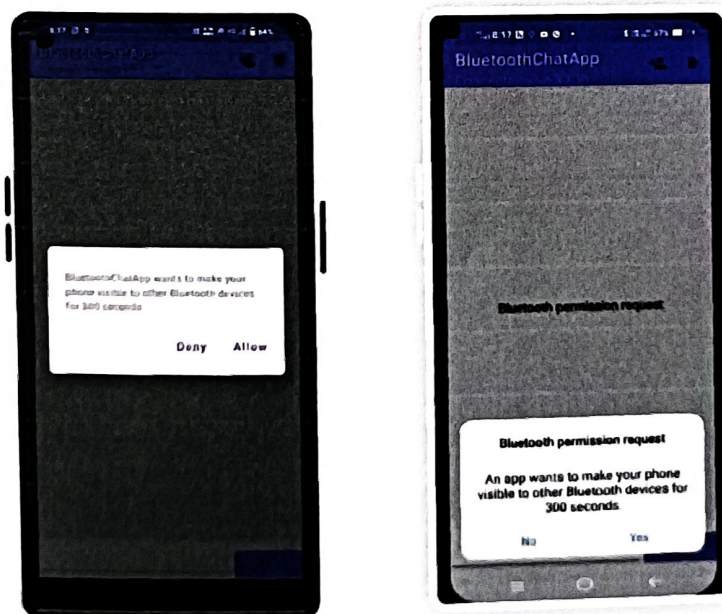


Fig 5.2.2 Bluetooth Permission Request for Pairing

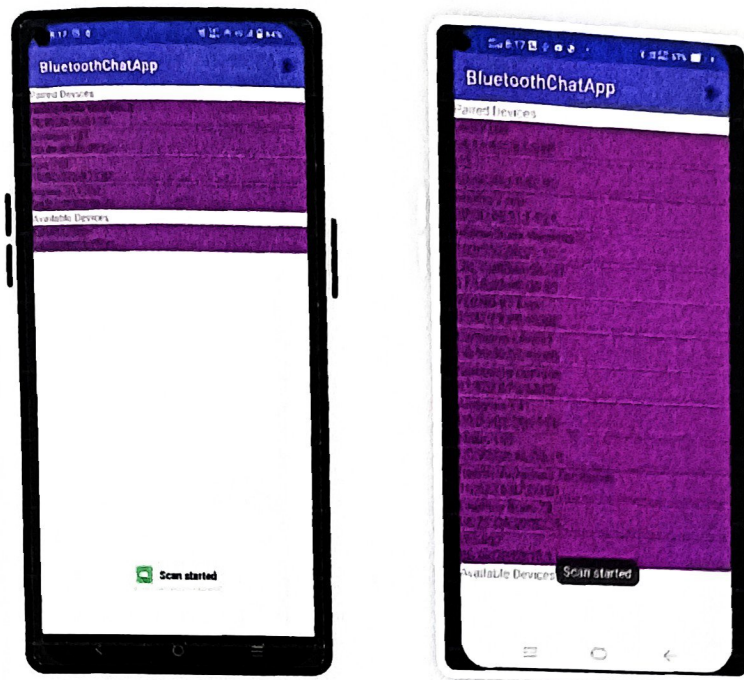


Fig 5.1:31. List Paired Devices
 2. List Available Devices
 3. Scan For Nearby Devices



Fig 5.2:4 Connected with a Device

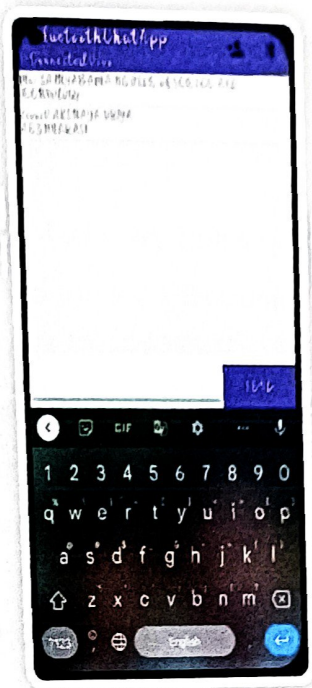
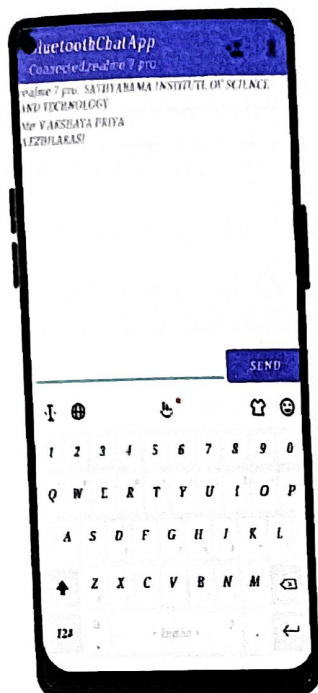


Fig 5.2.5 Send and Receive Text Messages

REFERENCES

1. Andre N Klingsheim. J2ME Bluetooth Programming [D]. Department of Informatics University of bergen,2004.
2. Yang Fengsheng. Android Inside [M]. Machinery Industry Press .2008
3. IEEE International Conference on Consumer Electronics, Communications & Networks, 2012-2013.
4. Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on 21-23 April 2012
5. The Bluetooth Special Interest Group. Bluetooth Specification Core v4.0.(2009-02).[Http://www.bluetooth.org](http://www.bluetooth.org).
6. E2ECloud studio. Google Android [M] Posts & Telecom Press . 2009
7. Hello, Android: Introducing googles Mobile Development platform(Paperback) by Ed Burnett
8. Bluetooth Messenger: an Android Messenger app based on Bluetooth Connectivity 1 Amrita Deb and 2 Swarnabha Sinha, IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661, p- ISSN: 2278-8727 Volume 16, Issue 3, Ver. III (May-Jun. 2014), PP 61-66 www.iosrjournals.org.
9. https://www.tutorialspoint.com/android/android_architecture.htm
10. <https://developer.android.com/guide/topics/connectivity/bluetooth#java>
11. Ramya S, Dr. A. Valarmathi Department of computer application, Anna University, BIT Campus, Tiruchirappalli, TamilNadu, India ,© 2017 IJSRCSEIT | Volume 2 | Issue 2 | ISSN : 2456-3307
12. Design of Chatting Application Based on Android Bluetooth Nikita Mahajan, Garima Verma, Gayatri Erale, Sneha Bonde, Divya Arya, IJCSMC, Vol. 3, Issue. 3, March