ASSIGNMENT- 2

1. Write a program that identifies the Bluetooth devices in the wireless range.

Discover_Bluetooth.java

```
package com.example.madpractical;

import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AppCompatActivity;

import android.Manifest;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.Build;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;

import java.util.ArrayList;

public class Discover_Bluetooth extends AppCompatActivity {
    private static final String TAG = "MainActivity";
    BluetoothAdapter bluetoothAdapter;
    public DeviceListAdapter deviceListAdapter;
    ListView listView;
    ArrayList<BluetoothDevice> deviceArrayList;

    /**
     * Broadcast Receiver for changes made to bluetooth states such as:
     * 1) Discoverability mode on/off or expire.
     */
    final BroadcastReceiver btReceiver = new BroadcastReceiver() {
        @Override
        public void onReceive(Context context, Intent intent) {
            String action = intent.getAction();
            if (action.equals(bluetoothAdapter.ACTION_STATE_CHANGED)) {
                final int state =
intent.getIntExtra(BluetoothAdapter.EXTRA_STATE, bluetoothAdapter.ERROR);

                switch (state) {
                    case BluetoothAdapter.STATE_OFF:
                        Log.d(TAG, "btReceiver: STATE OFF");
                        break;
                    case BluetoothAdapter.STATE_TURNING_OFF:
                        Log.d(TAG, "btReceiver: STATE TURNING OFF");
                        break;
                    case BluetoothAdapter.STATE_ON:
                        Log.d(TAG, "btReceiver: STATE ON");
                        break;
                    case BluetoothAdapter.STATE_TURNING_ON:
                        Log.d(TAG, "btReceiver: STATE TURNING ON");
                        break;
                }
            }
            //if(BluetoothDevice.ACTION_FOUND.equals(action)){
            //BluetoothDevice device =
intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
            // add the name to the list
            //BTArrayAdapter.add(device.getName() + "\n" +
device.getAddress());
            //mBTArrayAdapter.notifyDataSetChanged();
            //}
        }
    };

    /**
     * Broadcast Receiver for listing devices that are not yet paired
     * -Executed by btnDiscover() method.
     */
    final BroadcastReceiver enable_discovery_receiver = new
BroadcastReceiver() {
        @Override
        public void onReceive(Context context, Intent intent) {
            String action = intent.getAction();
            if (action.equals(bluetoothAdapter.ACTION_SCAN_MODE_CHANGED)) {
                final int state =
intent.getIntExtra(BluetoothAdapter.EXTRA_STATE, bluetoothAdapter.ERROR);

                switch (state) {
                    case BluetoothAdapter.SCAN_MODE_CONNECTABLE_DISCOVERABLE:
                        Log.d(TAG, "enable_discovery_receiver:
Discoverability Enabled");
                        break;
                    case BluetoothAdapter.SCAN_MODE_CONNECTABLE:
                        Log.d(TAG, "enable_discovery_receiver:
Discoverability Enabled. Able to receive Connect");
                        break;
                    case BluetoothAdapter.SCAN_MODE_NONE:
                        Log.d(TAG, "enable_discovery_receiver:
Discoverability Disabled. Not able to receive Connect");
                        break;
                    case BluetoothAdapter.STATE_CONNECTING:
                        Log.d(TAG, "enable_discovery_receiver:
Connecting...");
                        break;
                    case BluetoothAdapter.STATE_CONNECTED:
                        Log.d(TAG, "enable_discovery_receiver: Connected.");
                        break;
                }
            }
            //if(BluetoothDevice.ACTION_FOUND.equals(action)){
            //BluetoothDevice device =
intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
```

```
            // add the name to the list
            //BTArrayAdapter.add(device.getName() + "\n" + 
device.getAddress());
            //mBTArrayAdapter.notifyDataSetChanged();
            //}
        }
    };

    /**
     * Broadcast Receiver for listing devices that are not yet paired
     * -Executed by btnDiscover() method.
     */
    private BroadcastReceiver mBroadcastReceiver3 = new BroadcastReceiver() {
        @Override
        public void onReceive(Context context, Intent intent) {
            final String action = intent.getAction();
            Log.d(TAG, "onReceive: ACTION FOUND.");

            if (action.equals(BluetoothDevice.ACTION_FOUND)) {
                BluetoothDevice device = 
intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
                Log.d(TAG, deviceArrayList.size() + "");

                Log.d(TAG, "onReceive: " + device.getName() + ": " + 
device.getAddress());
                if (!deviceArrayList.contains(device)) 
deviceArrayList.add(device);
                Log.d(TAG, deviceArrayList.size() + "");
                deviceListAdapter.notifyDataSetChanged();
            }
        }
    };

    @Override
    protected void onDestroy() {
        super.onDestroy();
        Log.d(TAG, "onDestroy called");
        unregisterReceiver(btReceiver);
        unregisterReceiver(enable_discovery_receiver);
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_discover_bluetooth);
        Button on_off = findViewById(R.id.on_off);
        Button enable_discover = findViewById(R.id.enable_discover);
        listView = findViewById(R.id.device_list);
        deviceArrayList = new ArrayList<>();
        deviceListAdapter = new DeviceListAdapter(Discover_Bluetooth.this, 
R.layout.device_adapter_list, deviceArrayList);
        listView.setAdapter(deviceListAdapter);
        enable_discover.setOnClickListener(v -> {
            btnEnableDisable_Discoverable(v);
        });
        bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
        on_off.setOnClickListener(v -> {
            enableBluetooth();
        });
        Button discover = findViewById(R.id.discover_devices);
        discover.setOnClickListener(v -> btnDiscover(v));
    }

    public void enableBluetooth() {
        if (bluetoothAdapter == null) {
            Log.d("enableBlueTooth", "enabledisableBT: your device doesn't
have BT");
        }
        if (!bluetoothAdapter.isEnabled()) {
            Intent intent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivity(intent);

            IntentFilter BTintent = new
IntentFilter(BluetoothAdapter.ACTION_STATE_CHANGED);
            registerReceiver(btReceiver, BTintent);
        }
        if (bluetoothAdapter.isEnabled()) {
            bluetoothAdapter.disable();

            IntentFilter BTintent = new
IntentFilter(BluetoothAdapter.ACTION_STATE_CHANGED);
            registerReceiver(btReceiver, BTintent);
        }
    }

    public void btnEnableDisable_Discoverable(View view) {
        Log.d(TAG, "btnEnableDisable_Discoverable: Making device discoverable
for 300 seconds");

        Intent intent = new
Intent(BluetoothAdapter.ACTION_REQUEST_DISCOVERABLE);
        intent.putExtra(BluetoothAdapter.EXTRA_DISCOVERABLE_DURATION, 300);
        startActivity(intent);

        IntentFilter intentFilter = new
IntentFilter(bluetoothAdapter.ACTION_SCAN_MODE_CHANGED);
        registerReceiver(enable_discovery_receiver, intentFilter);
    }

    public void btnDiscover(View view) {
        Log.d(TAG, "btnDiscover: Looking for unpaired devices.");

        if (bluetoothAdapter.isDiscovering()) {
            bluetoothAdapter.cancelDiscovery();
            Log.d(TAG, "btnDiscover: Canceling discovery.");

            //check BT permissions in manifest
            checkBTPermissions();

            bluetoothAdapter.startDiscovery();
            IntentFilter discoverDevicesIntent = new
IntentFilter(BluetoothDevice.ACTION_FOUND);
            registerReceiver(mBroadcastReceiver3, discoverDevicesIntent);
        }
        if (!bluetoothAdapter.isDiscovering()) {

            //check BT permissions in manifest
            checkBTPermissions();

            bluetoothAdapter.startDiscovery();
            IntentFilter discoverDevicesIntent = new
IntentFilter(BluetoothDevice.ACTION_FOUND);
            registerReceiver(mBroadcastReceiver3, discoverDevicesIntent);
        }
    }

    /**
     * This method is required for all devices running API23+
     * Android must programmatically check the permissions for bluetooth.
Putting the proper permissions
     * in the manifest is not enough.
     * <p>
     * NOTE: This will only execute on versions > LOLLIPOP because it is not
needed otherwise.
     */
    private void checkBTPermissions() {
        if (Build.VERSION.SDK_INT > Build.VERSION_CODES.LOLLIPOP) {
            int permissionCheck =
this.checkSelfPermission("Manifest.permission.ACCESS_FINE_LOCATION");
            permissionCheck +=
this.checkSelfPermission("Manifest.permission.ACCESS_COARSE_LOCATION");
            if (permissionCheck != 0) {

                this.requestPermissions(new
String[]{Manifest.permission.ACCESS_FINE_LOCATION,
Manifest.permission.ACCESS_COARSE_LOCATION}, 1001); //Any number
            }
        } else {
            Log.d(TAG, "checkBTPermissions: No need to check permissions. SDK
version < LOLLIPOP.");
        }
    }

}
```

DevicelistAdapter.java

```
package com.example.madpractical;

import android.bluetooth.BluetoothDevice;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;

import java.util.ArrayList;
```

```
public class DeviceListAdapter extends ArrayAdapter<BluetoothDevice> {

    private LayoutInflater mLayoutInflater;
    private int mViewResourceId;

    public DeviceListAdapter(Context context, int tvResourceId,
ArrayList<BluetoothDevice> devices) {
        super(context, tvResourceId, devices);
        mLayoutInflater = LayoutInflater.from(context);
        mViewResourceId = tvResourceId;
    }

    public View getView(int position, View convertView, ViewGroup parent) {
        convertView = mLayoutInflater.inflate(mViewResourceId, parent,
false);

        BluetoothDevice device = getItem(position);
        TextView deviceName = convertView.findViewById(R.id.tvDeviceName);
        TextView deviceAdress =
convertView.findViewById(R.id.tvDeviceAddress);
        deviceName.setText(device.getName());
        deviceAdress.setText(device.getAddress());

        return convertView;
    }

}
```

device_adapter_list.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:id="@+id/tvDeviceName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:text="00:00"
        android:textColor="@color/white"
        android:textSize="20sp" />

    <TextView
        android:id="@+id/tvDeviceAddress"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginBottom="20dp"
        android:text="00:00"
        android:textColor="@color/white"
        android:textSize="20sp" />
```

```
</LinearLayout>
```

activity_discover_bluetooth.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/gradient"
    tools:context=".Discover_Bluetooth">

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <Button
                android:id="@+id/on_off"
                android:layout_width="250dp"
                android:layout_height="wrap_content"
                android:layout_gravity="center"
                android:layout_marginTop="20dp"
                android:text="ON/OFF"
                android:textSize="24sp" />

            <Button
                android:id="@+id/enable_discover"
                android:layout_width="250dp"
                android:layout_height="wrap_content"
                android:layout_gravity="center"
                android:layout_marginTop="20dp"
                android:text="Enable discover"
                android:textSize="24sp" />

            <Button
                android:id="@+id/discover_devices"
                android:layout_width="250dp"
                android:layout_height="wrap_content"
                android:layout_gravity="center"
                android:layout_marginTop="20dp"
                android:text="discover devices"
                android:textSize="24sp" />

            <ListView
```

```
                android:id="@+id/device_list"
                android:layout_width="match_parent"
                android:layout_height="400dp"
                android:layout_gravity="center_horizontal"
                android:layout_marginTop="20dp"
                android:foregroundGravity="center_horizontal" />

        </LinearLayout>
    </ScrollView>

</androidx.constraintlayout.widget.ConstraintLayout>
```

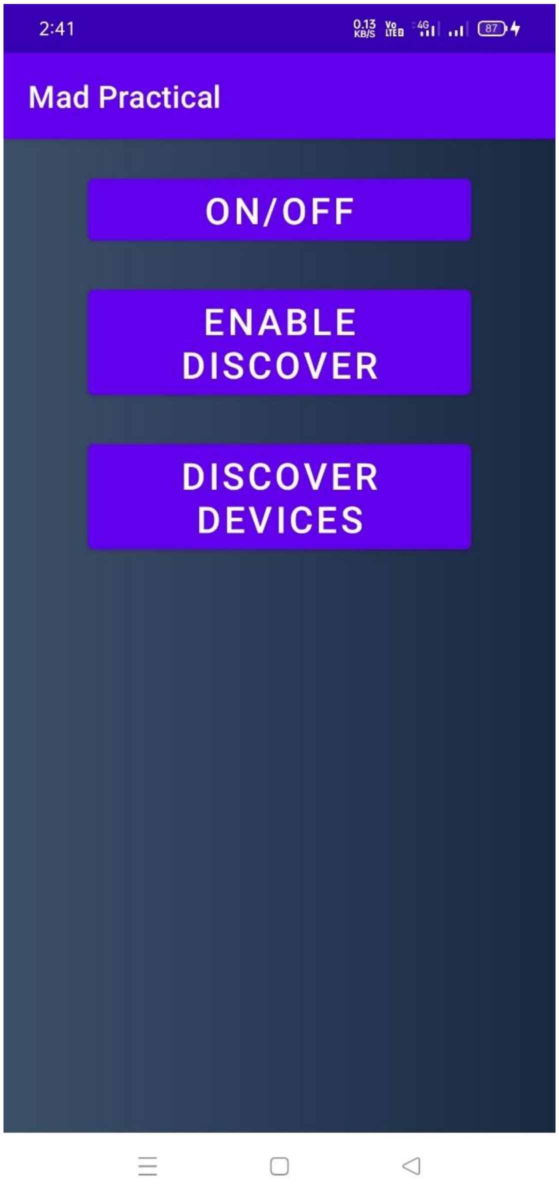
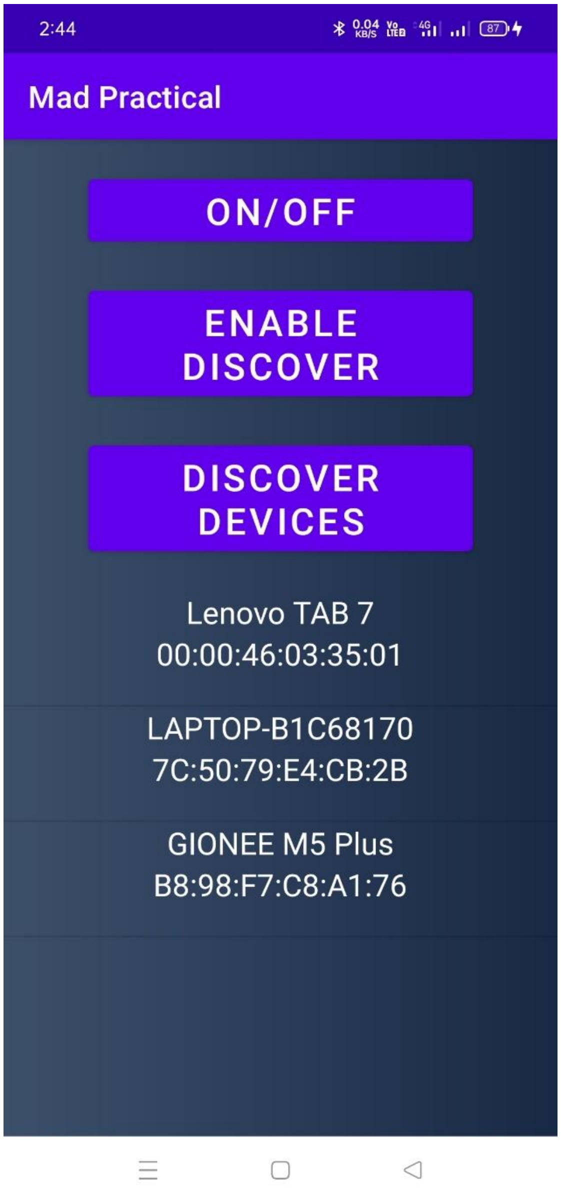


Fig 1 – Bluetooth options

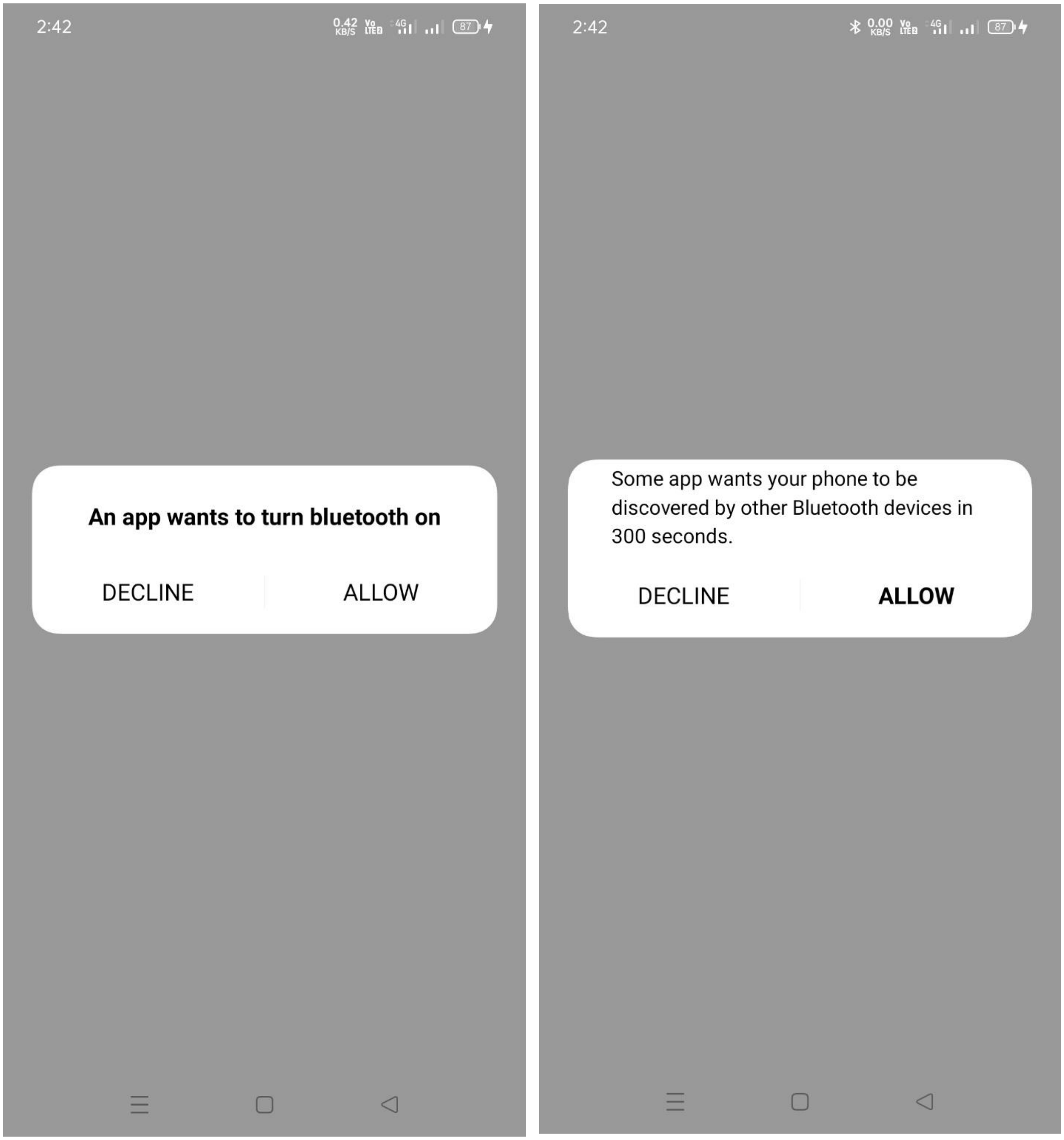


Lenovo TAB 7
00:00:46:03:35:01

LAPTOP-B1C68170
7C:50:79:E4:CB:2B

GIONEE M5 Plus
B8:98:F7:C8:A1:76

Fig 2 – Devices Discovery



An app wants to turn bluetooth on

Some app wants your phone to be discovered by other Bluetooth devices in 300 seconds.

Fig 3 & 4 Take Permission

2. Write a program that prints the signal strength of WiFi connection of the given computer/mobile.

Wifi_Signal.java

```
package com.example.madpractical;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;

import android.Manifest;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.pm.PackageManager;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiManager;
import android.os.Bundle;
import android.util.Log;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.List;

public class Wifi_Signal extends AppCompatActivity {

    private WifiManager wifiManager;
    private WifiReceiver wifiReceiver;
    private List<ScanResult> result = new ArrayList<>();
    ScanResultsAdapter scanResultsAdapter;
    ListView deviceList;
    StringBuilder sb = new StringBuilder();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_wifi_signal);
        ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.ACCESS_WIFI_STATE,
                Manifest.permission.CHANGE_WIFI_STATE,
Manifest.permission.ACCESS_FINE_LOCATION,
Manifest.permission.ACCESS_COARSE_LOCATION},
PackageManager.PERMISSION_GRANTED);

        wifiManager = (WifiManager)
getApplicationContext().getSystemService(Context.WIFI_SERVICE);
        wifiReceiver = new WifiReceiver();
        registerReceiver(wifiReceiver, new
IntentFilter(WifiManager.SCAN_RESULTS_AVAILABLE_ACTION));
        Log.d("MAinActivity", "Starting scan");
        wifiManager.startScan();
        Button on_off = findViewById(R.id.on_off_wifi);
        on_off.setOnClickListener(v -> {
            if (!wifiManager.isWifiEnabled()) {
                wifiManager.setWifiEnabled(true);
                Log.d("Wifi state", "wifi enable");
            } else {
                wifiManager.setWifiEnabled(false);
                Log.d("Wifi state", "wifi disable");
            }
        });

        Button scan = findViewById(R.id.enable_discover_wifi);
        scan.setOnClickListener(v -> {
            wifiManager.setWifiEnabled(true);
            Boolean success = wifiManager.startScan();

            if (success) Log.d("startscan", "started");
            else Log.d("startscan", "not started");

            Log.d("Buttton", "Scanning...");
        });
    }

    private void setadapter() {
        deviceList = findViewById(R.id.device_list);
        scanResultsAdapter = new ScanResultsAdapter(Wifi_Signal.this,
R.layout.network_list_row, result, wifiManager);
        deviceList.setAdapter(scanResultsAdapter);
    }

    protected void onResume() {
        super.onResume();
        registerReceiver(wifiReceiver, new IntentFilter(
                WifiManager.SCAN_RESULTS_AVAILABLE_ACTION));
    }

    protected void onPause() {
        super.onPause();
        unregisterReceiver(wifiReceiver);
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        unregisterReceiver(wifiReceiver);
    }

    public class WifiReceiver extends BroadcastReceiver {
        @Override
        public void onReceive(Context context, Intent intent) {
            sb = new StringBuilder();
            result = wifiManager.getScanResults();
            Log.d("WifiReceiver", "Received Something");
            // prepare text for display and CSV table
            sb.append("Number of APs Detected: ");
            sb.append((Integer.valueOf(result.size())).toString());
            sb.append("\n\n");
            for (int i = 0; i < result.size(); i++) {
                // sb.append((Integer.valueOf(i + 1)).toString() + ".");
                // SSID
                sb.append("SSID:").append((result.get(i)).SSID);
                sb.append("\n");
                // BSSID
                sb.append("BSSID:").append((result.get(i)).BSSID);
                sb.append("\n");
                // capabilities
                sb.append("Capabilities:").append(
                        (result.get(0)).capabilities);
                sb.append("\n");
                // frequency
                sb.append("Frequency:").append((result.get(i)).frequency);
                sb.append("\n");

                // level
                sb.append("Level:").append((result.get(i)).level);
                sb.append("\n\n");
            }
            Log.d("WifiReceiver", sb.toString());
            Log.d("WifiReceiver", "setting adapter");
            setadapter();
            // notify that Wi-Fi scan has finished
        }
    }


}
```

ScanResultAdapter.java

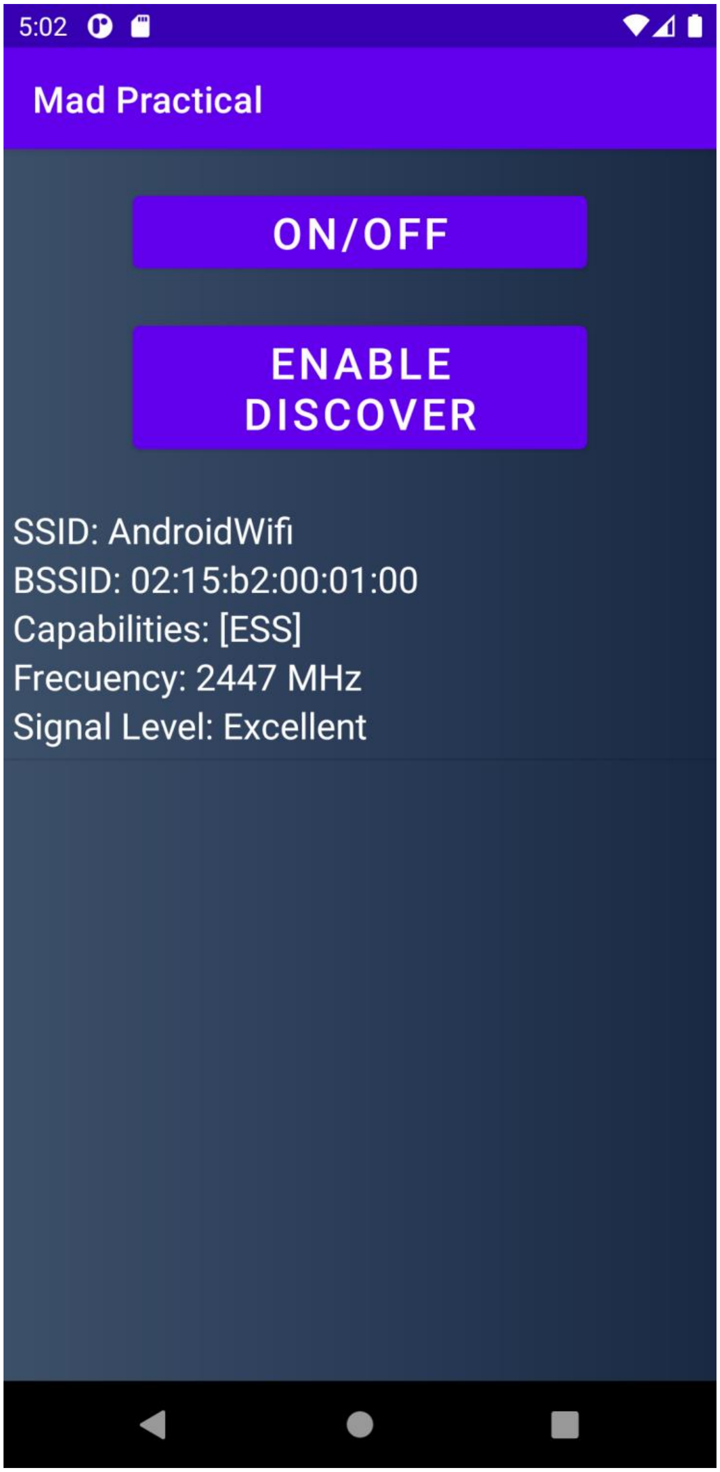
```
package com.example.madpractical;

import java.util.List;

import android.content.Context;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiManager;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;

public class ScanResultsAdapter extends ArrayAdapter<ScanResult> {

    private final Context context;
    private final List<ScanResult> results;
    private int mViewResourceId;
    private LayoutInflater mLayoutInflater;
    WifiManager wifiManager;
    private final String[] signalStrength = {"Weak", "Fair", "Good",
"Excellent"};

    public ScanResultsAdapter(Context context, int tvResourceId,
List<ScanResult> results, WifiManager wifiManager) {
        super(context, tvResourceId, results);
        this.context = context;
        this.results = results;
        mViewResourceId = tvResourceId;
        mLayoutInflater = LayoutInflater.from(context);
        this.wifiManager = wifiManager;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        convertView = mLayoutInflater.inflate(mViewResourceId, parent,
false);
        if (results.size() > 0) {
            ScanResult result = getItem(position);
            // Get textview fields

            TextView txtSSID = convertView.findViewById(R.id.txtSSID);
            TextView txtBSSID = convertView.findViewById(R.id.txtBSSID);
            TextView txtCapabilities =
convertView.findViewById(R.id.txtCapabilities);
            TextView txtFrecuency =
convertView.findViewById(R.id.txtFrecuency);
            TextView txtLevel = convertView.findViewById(R.id.txtLevel);

            int level = wifiManager.calculateSignalLevel(result.level);
            // Set fields values

txtSSID.setText(convertView.getContext().getString(R.string.ssid_msg,
result.SSID));

txtBSSID.setText(convertView.getContext().getString(R.string.bssid_msg,
result.BSSID));

txtCapabilities.setText(convertView.getContext().getString(R.string.capabilit
ies_msg, result.capabilities));

txtFrecuency.setText(convertView.getContext().getString(R.string.frecuency_ms
g, Integer.toString(result.frequency)));

txtLevel.setText(convertView.getContext().getString(R.string.signal_level_msg
, signalStrength[level - 1]));
        }
        return convertView;
    }


}
```

activity_wifi_signal.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/gradient"
    tools:context=".Wifi_Signal">

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <Button
                android:id="@+id/on_off_wifi"
                android:layout_width="250dp"
                android:layout_height="wrap_content"
                android:layout_gravity="center"
                android:layout_marginTop="20dp"
                android:text="ON/OFF"
                android:textSize="24sp" />

            <Button
                android:id="@+id/enable_discover_wifi"
                android:layout_width="250dp"
                android:layout_height="wrap_content"
                android:layout_gravity="center"
                android:layout_marginTop="20dp"
                android:text="Enable discover"
                android:textSize="24sp" />

            <ListView
                android:id="@+id/device_list"
                android:layout_width="match_parent"
                android:layout_height="400dp"
                android:layout_gravity="center_horizontal"
                android:layout_marginTop="20dp"
                android:foregroundGravity="center_horizontal" >


            </ListView>

        </LinearLayout>
    </ScrollView>

</androidx.constraintlayout.widget.ConstraintLayout>
```

network_list_row.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/network"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:gravity="left|center"
    android:orientation="vertical"
    android:paddingLeft="5dp"
    android:paddingTop="5dp"
    android:paddingBottom="5dp">

    <TextView
        android:id="@+id/txtSSID"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/ssid_msg"
        android:textColor="@color/white"
        android:textSize="20sp" />

    <TextView
        android:id="@+id/txtBSSID"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="@color/white"
        android:textSize="20sp" />

    <TextView
        android:id="@+id/txtCapabilities"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="@color/white"
        android:textSize="20sp" />

    <TextView
        android:id="@+id/txtFrecuency"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="@color/white"
        android:textSize="20sp" />

    <TextView
        android:id="@+id/txtLevel"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="@color/white"
        android:textSize="20sp" />
</LinearLayout>
```



SSID: AndroidWifi
BSSID: 02:15:b2:00:01:00
Capabilities: [ESS]
Frecuency: 2447 MHz
Signal Level: Excellent

Fig 5 – Wi-Fi Scanner shows Signal strength with extra details