Codice Client Android

Classe Main

```
import android.Manifest;
import android.app.Dialog;
import android.content.ComponentName;
import android.content.Context;
import android.content.Intent;
import android.content.ServiceConnection;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.os.IBinder;
import android.os.Message;
import android.provider.ContactsContract;
import android.support.design.widget.NavigationView;
import android.support.v4.view.GravityCompat;
import android.support.v4.widget.DrawerLayout;
import android.support.v7.app.ActionBarDrawerToggle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.MenuItem;
import android.view.View;
import android.view.Window;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;

public class Main extends AppCompatActivity
       implements NavigationView.OnNavigationItemSelectedListener {

   private String name, surname;
   private ConnectorService connectorService;
   private boolean isBound = false;
   private SynchronizedQueue<Object> synchronizedQueue = new
SynchronizedQueue<>();
   private SecurePreferences securePreferences;
   private Handler handler;
   private static final int PERMISSIONS_REQUEST_READ_CONTACTS =
100;
   private ContactListAdapter contactAdapter;
```

```
    private EditText etTel;
    private Dialog dialog;

    @Override
    protected void onStart() {
        super.onStart();
        bindService(new Intent(this, ConnectorService.class),
serviceConnection, Context.BIND_AUTO_CREATE);

    }

    @Override
    protected void onDestroy() {
        super.onDestroy();


connectorService.comunicate(Utils.delIP(securePreferences.getStrin
g("tel")), this);
        if (connectorService != null)
            unbindService(serviceConnection);
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        securePreferences = new SecurePreferences(this);
        //securePreferences.putBoolean("firstLog", true);
        DrawerLayout drawer = (DrawerLayout)
findViewById(R.id.drawer_layout);
        ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
                this, drawer, toolbar,
R.string.navigation_drawer_open,
R.string.navigation_drawer_close);
        drawer.setDrawerListener(toggle);
        toggle.syncState();

        NavigationView navigationView = (NavigationView)
findViewById(R.id.nav_view);
        navigationView.setNavigationItemSelectedListener(this);
        View hView = navigationView.getHeaderView(0);
        TextView tvHeadName = (TextView)
hView.findViewById(R.id.tvHeadName);
        Intent intent = getIntent();
        name = intent.getStringExtra("name");
        surname = intent.getStringExtra("surname");
        tvHeadName.setText(name + " " + surname);

        ArrayList<Contact> contacts = new ArrayList<>();
        contactAdapter = new ContactListAdapter(this, contacts);
        final ListView lvChat = (ListView)
findViewById(R.id.lvChat);
        lvChat.setAdapter(contactAdapter);
        lvChat.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
            public void onItemClick(AdapterView<?> parent, View
view,
                                    int position, long id) {

                Object o = lvChat.getItemAtPosition(position);
                Contact c = (Contact) o;
                System.out.println(c.getTel());
                startActivity(new Intent(Main.this,
Chat.class).putExtra("name", c.getName())
                        .putExtra("surname", c.getSurname())
                        .putExtra("tel", c.getTel()));
            }
        });

        if (securePreferences.getString("tel") == null)
            showTelDialog();


    }

    @Override
    public void onBackPressed() {
        DrawerLayout drawer = (DrawerLayout)
findViewById(R.id.drawer_layout);
        if (drawer.isDrawerOpen(GravityCompat.START)) {
            drawer.closeDrawer(GravityCompat.START);
        } else {
            super.onBackPressed();
        }
    }

    @SuppressWarnings("StatementWithEmptyBody")
    @Override
    public boolean onNavigationItemSelected(MenuItem item) {
        // Handle navigation view item clicks here.
        int id = item.getItemId();

        DrawerLayout drawer = (DrawerLayout)
```

```
findViewById(R.id.drawer_layout);
        drawer.closeDrawer(GravityCompat.START);
        return true;
    }

    public void showTelDialog() {
        dialog = new Dialog(this);

        dialog.getWindow();
        dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
        dialog.setContentView(R.layout.custom_dialog1);
        dialog.setCancelable(true);


        etTel = (EditText) dialog.findViewById(R.id.etTel);
        Button btOk = (Button) dialog.findViewById(R.id.btOk);
        Button btAnnulla = (Button)
dialog.findViewById(R.id.btAnnulla);

        btOk.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (!etTel.getText().toString().isEmpty()) {
                    securePreferences.putString("tel",
etTel.getText().toString());

connectorService.comunicate(Utils.logIP(etTel.getText().toString()
), Main.this);
                    dialog.dismiss();
                } else {
                    Utils.errRegisterToast(Main.this);
                }
            }
        });
        btAnnulla.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                dialog.dismiss();
            }
        });
        dialog.show();
    }

    public void addUsersToList() {
        ArrayList<Contact> newContacts = new
SQLiteManager(this).getUsers();
        contactAdapter.clear();
        if (newContacts != null && !newContacts.isEmpty())
```

```
            contactAdapter.addAll(newContacts);
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, String[]
permissions,
                                            int[] grantResults) {
        if (requestCode == PERMISSIONS_REQUEST_READ_CONTACTS) {
            if (grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                // Permission is granted
                getUsers();
            } else {
                Toast.makeText(this, "Finchè non acconsenti non
potrati vedere i contatti nella lista",
Toast.LENGTH_SHORT).show();
            }
        }
    }

    public void getUsers() {
        try {
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M &&
checkSelfPermission(Manifest.permission.READ_CONTACTS) !=
PackageManager.PERMISSION_GRANTED) {
                requestPermissions(new
String[]{Manifest.permission.READ_CONTACTS},
PERMISSIONS_REQUEST_READ_CONTACTS);
                return;
            }
            Cursor phones = getContentResolver()

.query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI, null,
null, null, null);
            ArrayList<Object> telNumbers = new ArrayList<>();

            while (phones.moveToNext()) {
                String tmp = phones.getString(

phones.getColumnIndex(ContactsContract.CommonDataKinds.Phone.NUMBE
R)).replaceAll("\\s+", "");
                String phoneNumber;
                if (tmp.charAt(0) == '+')
                    phoneNumber = tmp.substring(3);
                else
                    phoneNumber = tmp;

                if (!telNumbers.contains(phoneNumber))
```

```
                    telNumbers.add(phoneNumber);
            }
            phones.close();


connectorService.comunicate(Utils.checkUsers(telNumbers),
Main.this, new Handler() {
                @Override
                public void handleMessage(Message msg) {
                    super.handleMessage(msg);
                    Bundle bundle = msg.getData();
                    Msg responce = (Msg)
bundle.getSerializable("res");

                    if (responce != null && responce.getId() !=
151836) {
                        if (!responce.getData().isEmpty()) {
                            new
SQLiteManager(Main.this).clearUsers(true);
                            for (Object c : responce.getData()) {
                                Contact contact = (Contact) c;
                                new
SQLiteManager(Main.this).addUser(contact.getName(),
                                        contact.getSurname(),
                                        contact.getUsername(),
                                        contact.getTel());
                            }
                            addUsersToList();
                        }
                    } else {
                        try {
                            throw new Exception();
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                    }
                }
            });
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private ServiceConnection serviceConnection = new
ServiceConnection() {
        @Override
        public void onServiceConnected(ComponentName componentName,
IBinder iBinder) {
```

```
           ConnectorService.ConnectorBinder binder =
(ConnectorService.ConnectorBinder) iBinder;
           connectorService = binder.getService();
           isBound = true;

connectorService.comunicate(Utils.logIP(securePreferences.getStrin
g("tel")), Main.this);
           //if(securePreferences.getBoolean("firstLog")) {

           getUsers();
           //securePreferences.putBoolean("firstLog", false);
           //}
       }
       @Override
       public void onServiceDisconnected(ComponentName
componentName) {
           isBound = false;
       }
   };
}
```

Work In Process