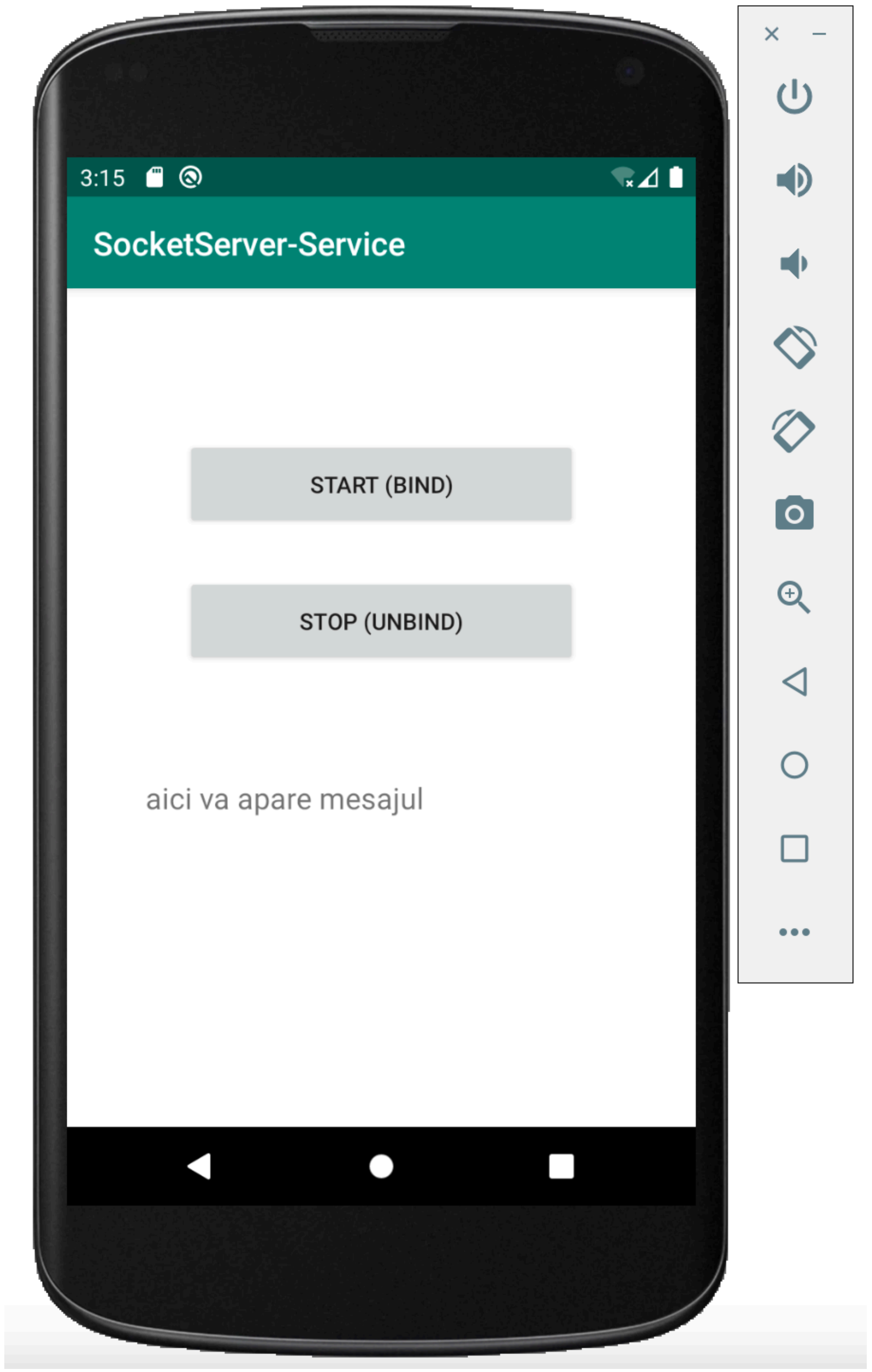
TCP Server - Android Service

Implementati un server TCP (sockets) într-un serviciu Android: verificați comunicația cu un client java din linia de comanda

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="elixer.com.boundservice">

    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme"
        tools:ignore="GoogleAppIndexingWarning">
        <service
            android:name=".SocketService"
            android:enabled="true"
            android:exported="true"></service>
        <service
            android:name=".BoundService"
            android:enabled="true"
            android:exported="true" />

        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

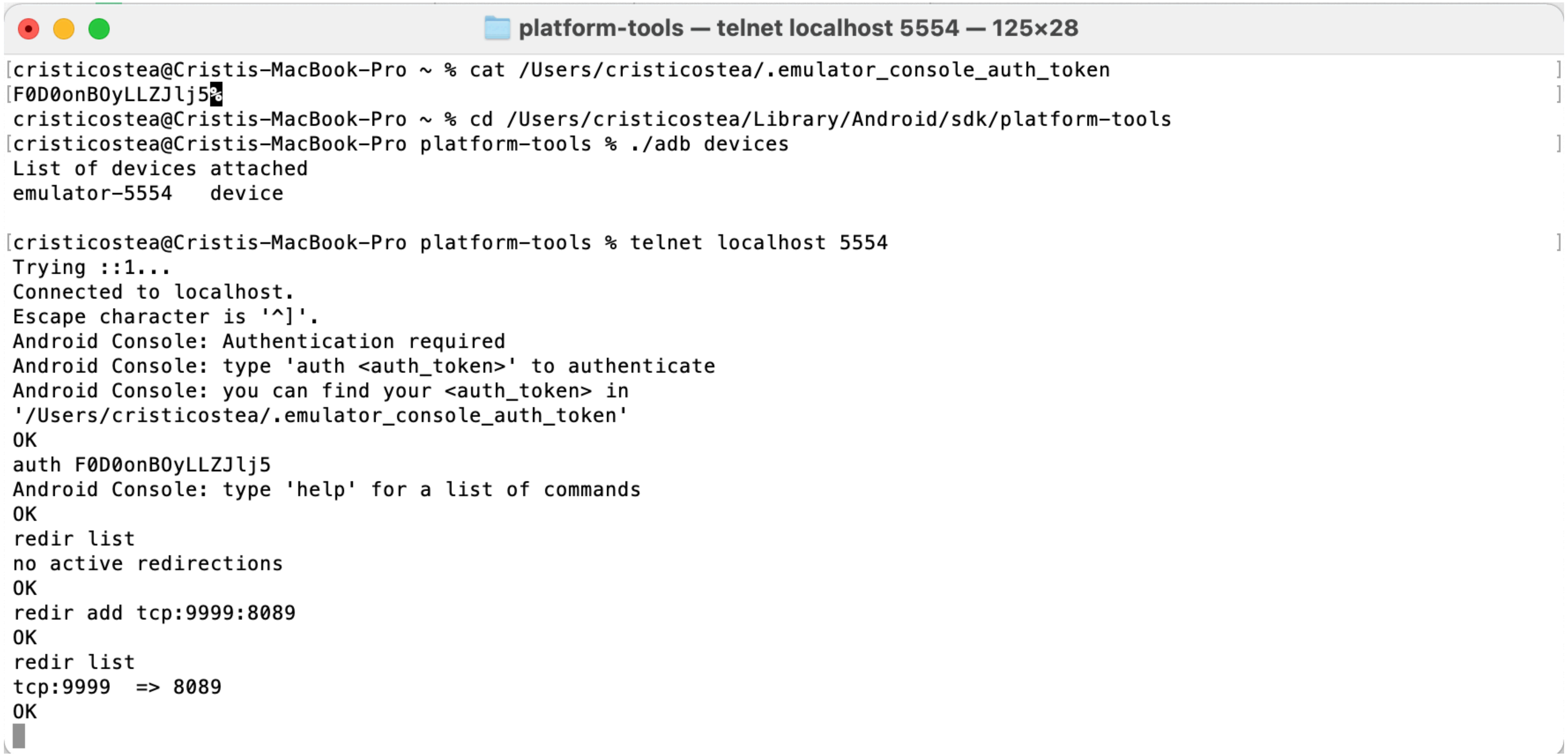
```xml
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/linearLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#FFFFFF">


    <Button
        android:id="@+id/bind"
        android:layout_width="240dp"
        android:layout_height="56dp"
        android:layout_marginStart="32dp"
        android:layout_marginTop="120dp"
        android:layout_marginEnd="32dp"
        android:layout_marginBottom="32dp"
        android:text="Start (bind)"
        app:layout_constraintBottom_toTopOf="@+id/stop_service"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/stop_service"
        android:layout_width="240dp"
        android:layout_height="56dp"
        android:layout_marginStart="32dp"
        android:layout_marginTop="24dp"
        android:layout_marginEnd="32dp"
        android:layout_marginBottom="32dp"
        android:text="Stop (unbind)"
        app:layout_constraintBottom_toTopOf="@+id/txtmsg"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/bind" />

    <TextView
        android:id="@+id/txtmsg"
        android:layout_width="416dp"
        android:layout_height="194dp"
        android:layout_marginStart="48dp"
        android:layout_marginTop="64dp"
        android:layout_marginBottom="48dp"
        android:text="aici va apare mesajul"
        android:textAppearance="@style/TextAppearance.AppCompat.Medium"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/stop_service" />

</android.support.constraint.ConstraintLayout>
```



```
  boolean mServiceBound = false;
  private SocketService mBoundService;
  private Boolean mIsBound;

  public MainActivity /*SocketServiceController*/ ssc;
    @Override
    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
        ssc = this;
        setContentView(R.layout.activity_main);
        Button start = (Button)findViewById(R.id.bind);
        Button stop = (Button)findViewById(R.id.stop_service);
        Button bindButton = findViewById(R.id.bind);

        start.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(getApplicationContext(), SocketService.class);
                bindService(intent, mConnection, Context.BIND_AUTO_CREATE);
            }
        });
       stop.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (mServiceBound) {
                    unbindService(mConnection);
                    mServiceBound = false;
                }
            }
        });

    }

    private ServiceConnection mConnection = new ServiceConnection() {
        public void onServiceConnected(ComponentName className, IBinder service) {
            mBoundService = ((SocketService.LocalBinder)service).getService();
            mServiceBound = true;
        }

        public void onServiceDisconnected(ComponentName className) {
            mBoundService = null;
        }
    };

    private void doBindService() {
        bindService(new Intent(MainActivity.this, SocketService.class), mConnection, Context.BIND_AUTO_CREATE);
        mIsBound = true;
        mBoundService.IsBoundable();
    }


    private void doUnbindService() {
        if (mIsBound) {
            // Detach our existing connection.
            unbindService(mConnection);
            mIsBound = false;
        }
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        doUnbindService();
    }
}
```

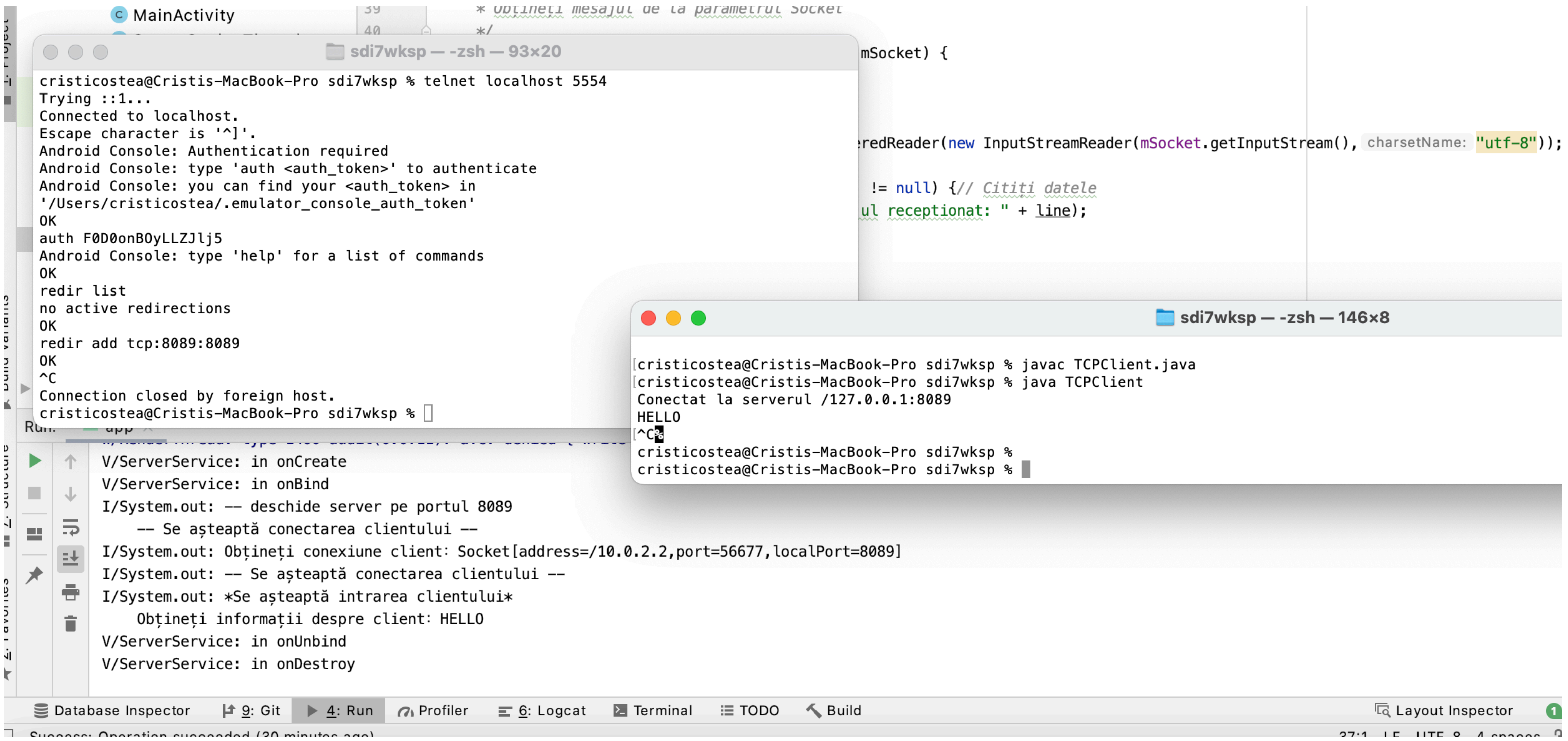
```
public class SocketService extends Service {
    private static String LOG_TAG = "ServerService";

    @Override
    public IBinder onBind(Intent arg0) {
        Log.v(LOG_TAG, "in onBind");
        // TODO Auto-generated method stub
        return myBinder;
    }

    private final IBinder myBinder = new LocalBinder();

    public class LocalBinder extends Binder {
        public SocketService getService() {
            return SocketService.this;
        }
        // Return object of SocketService class which can be used
        // to access all the public methods of this class
    }


    @Override
    public void onCreate() {
        super.onCreate();
        Log.v(LOG_TAG, "in onCreate");
        ServerSocketThread sst = new ServerSocketThread();
        sst.start();
    }

    public void IsBoundable(){
        Toast.makeText(this,"I bind like butter", Toast.LENGTH_LONG).show();
    }

   @Override
    public void onRebind(Intent intent) {
        Log.v(LOG_TAG, "in onRebind");
        super.onRebind(intent);
    }

    @Override
    public boolean onUnbind(Intent intent) {
        Log.v(LOG_TAG, "in onUnbind");
        return true;
    }


    @Override
    public void onDestroy() {
        super.onDestroy();
        Log.v(LOG_TAG, "in onDestroy");
    }
}
```

```java
package elixer.com.boundservice;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;

class ServerSocketThread extends Thread {

    @Override
    public void run() {
        try {
            // Creati ServerSocket
            ServerSocket serverSocket = new ServerSocket(8089);
            System.out.println("Start server, port 8089");

            // așteapta conectarea clientului
            while (true) {
                System.out.println("Se așteaptă conectarea clientului..");
                Socket socket = serverSocket.accept(); //Se așteaptă conectarea clientului
                System.out.println("conexiune client: " + socket);
                startReader(socket);
            }

        } catch (IOException e) {
            e.printStackTrace();
        }
    }


    /**
     * Obțineți mesajul
    */
    private static void startReader(final Socket mSocket) {
        new Thread(){
            @Override
            public void run() {
                try {
                    // Obțineți fluxul de citire
                    BufferedReader in = new BufferedReader(new InputStreamReader(mSocket.getInputStream(),"utf-8"));
                    String line="";
                    while ((line = in.readLine()) != null) {// Citiți datele
                        System.out.println("mesajul receptionat: " + line);
                    }
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }.start();
    }


}
```



platform-tools — telnet localhost 5554 — 125×28

```
cristicostea@Cristis-MacBook-Pro ~ % cat /Users/cristicostea/.emulator_console_auth_token
F0D0onBOyLLZJlj5%
cristicostea@Cristis-MacBook-Pro ~ % cd /Users/cristicostea/Library/Android/sdk/platform-tools
cristicostea@Cristis-MacBook-Pro platform-tools % ./adb devices
List of devices attached
emulator-5554   device

cristicostea@Cristis-MacBook-Pro platform-tools % telnet localhost 5554
Trying ::1...
Connected to localhost.
Escape character is '^]'.
Android Console: Authentication required
Android Console: type 'auth <auth_token>' to authenticate
Android Console: you can find your <auth_token> in
'/Users/cristicostea/.emulator_console_auth_token'
OK
auth F0D0onBOyLLZJlj5
Android Console: type 'help' for a list of commands
OK
redir list
no active redirections
OK
redir add tcp:9999:8089
OK
redir list
tcp:9999  => 8089
OK
```



```
* Obtineti mesajul de la parametrul Socket
*/
mSocket) {

eredReader(new InputStreamReader(mSocket.getInputStream(), charsetName: "utf-8"));

 != null) {// Cititi datele
ul receptionat: " + line);
```

```
cristicostea@Cristis-MacBook-Pro sdi7wksp % telnet localhost 5554
Trying ::1...
Connected to localhost.
Escape character is '^]'.
Android Console: Authentication required
Android Console: type 'auth <auth_token>' to authenticate
Android Console: you can find your <auth_token> in
'/Users/cristicostea/.emulator_console_auth_token'
OK
auth F0D0onBOyLLZJlj5
Android Console: type 'help' for a list of commands
OK
redir list
no active redirections
OK
redir add tcp:8089:8089
OK
^C
Connection closed by foreign host.
cristicostea@Cristis-MacBook-Pro sdi7wksp %
```

```
cristicostea@Cristis-MacBook-Pro sdi7wksp % javac TCPClient.java
cristicostea@Cristis-MacBook-Pro sdi7wksp % java TCPClient
Conectat la serverul /127.0.0.1:8089
HELLO
^C%
cristicostea@Cristis-MacBook-Pro sdi7wksp %
cristicostea@Cristis-MacBook-Pro sdi7wksp %
```

```
V/ServerService: in onCreate
V/ServerService: in onBind
I/System.out: -- deschide server pe portul 8089
    -- Se așteaptă conectarea clientului --
I/System.out: Obțineți conexiune client: Socket[address=/10.0.2.2,port=56677,localPort=8089]
I/System.out: -- Se așteaptă conectarea clientului --
I/System.out: *Se așteaptă intrarea clientului*
    Obțineți informații despre client: HELLO
V/ServerService: in onUnbind
V/ServerService: in onDestroy
```