INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

APPLICATION DEVELOPMENT FOR MOBILE DEVICES

“PROYECTO 02: COMUNICACIÓN WIFI”

GRUPO: 3CV16

BOBADILLA SEGUNDO DAN ISRAEL

PROFESOR: CIFUENTES ALVAREZ ALEJANDRO SIGFRIDO

FECHA DE ENTREGA: 23/11/2021

INTRODUCCIÓN

Como hemos visto en el proyecto 01 la conexión mediante Bluetooth es uno de los elementos principales en las conexiones actuales, pero en el caso del Wifi podemos asegurar que es la conexión mas importante de cualquier dispositivo moderno, en este caso realizaremos la conexión mediante Wifi mediante el celular y la computadora.

DESARROLLO

Para comenzar debemos de saber que nuestro modem nos asigna una dirección IP a cada dispositivo conectado en la red, en este caso necesitaremos la IP de nuestra computadora ya que utilizaremos el protocolo TCP. Para encontrar nuestra dirección IP deberemos de abrir la consola del sistema y escribir IPCONFIG, la IP se mostrará en la ilustración 1:

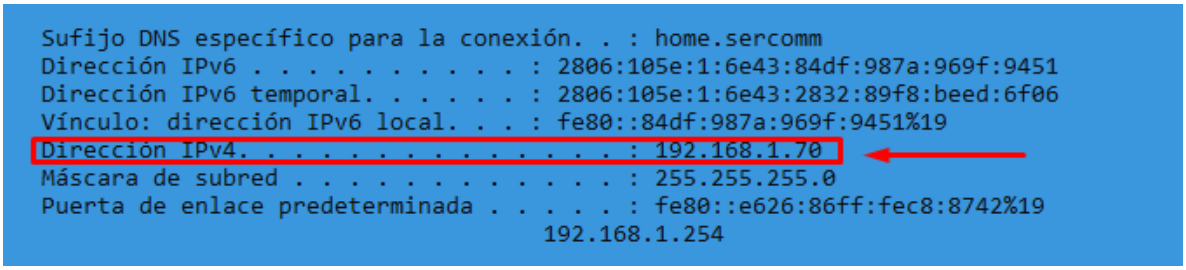

Ilustración 1

La dirección mostrada será la que ingresaremos a nuestro celular y a continuación necesitaremos ingresar un mensaje que será enviado a nuestra computadora como se muestra en la ilustración 2

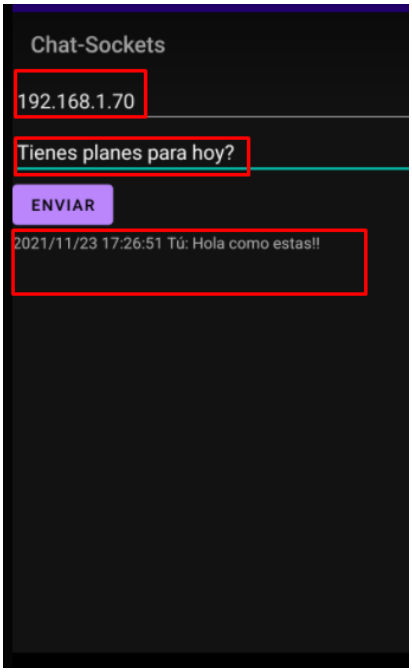

Ilustración 2

Ahora este mensaje llegara a nuestra computadora y podremos contestar en nuestra computadora hacia el celular Ilustración 3.

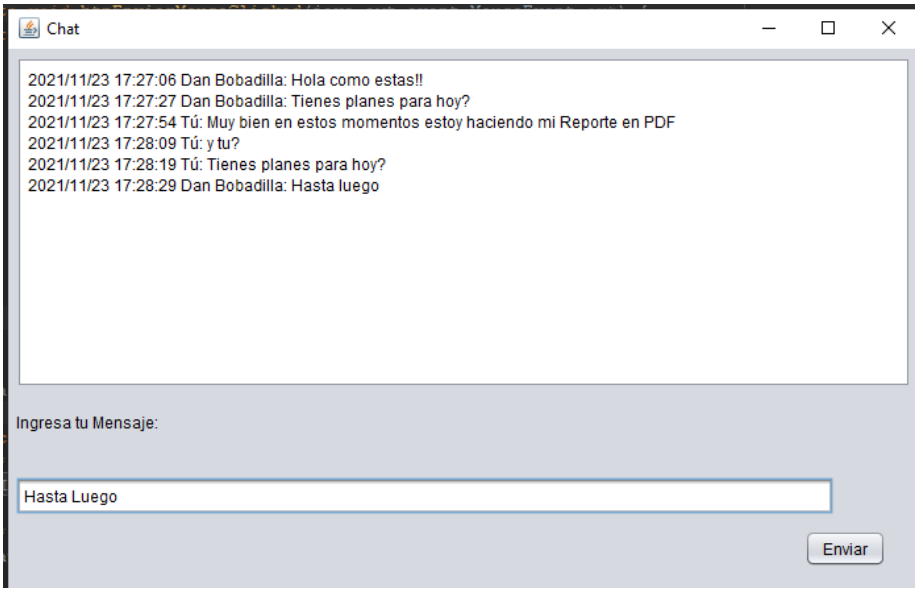

Ilustración 3

Y Ahora podemos ver el chat completo Ilustración 4.

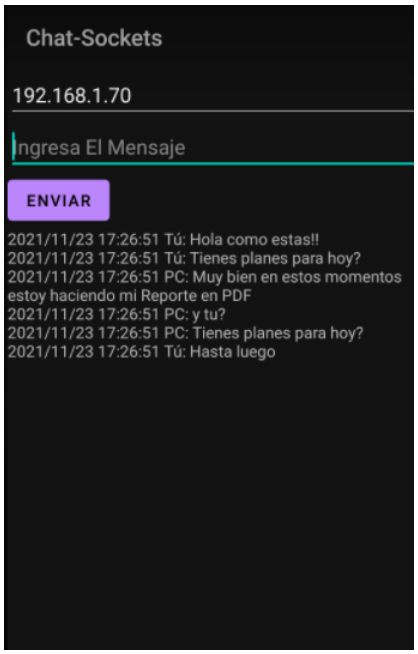

Ilustración 4

CODIGO

Para realizar el código me base en un código previamente realizado en Lenguaje C en Linux, la diferencia es que Java por defecto nos configura todo el socket y en Linux debemos de acceder a las capas mas internas para realizar la configuración.

Para programar el código en Android debemos de otorgarle permisos a la aplicación en el Manifest Ilustración 5.

```
nifest xmlns:android="http://schemas.android.com/apk/res/android
 package="com.example.chat_sockets">
 <uses-permission android:name="android.permission.INTERNET"/>
 <application
```

Ilustración 5

Para realizar la programación de los Sockets se vera desde Java ya que utilizamos Java puro tanto en Netbeans como en Android.

```
try {
    ServerSocket server = new ServerSocket(8000);
    while (true) {
        Socket cliente = server.accept();
        ipDevice = cliente.getInetAddress().toString().substring(1);
        BufferedReader entrada = new BufferedReader(new InputStreamReader(cliente.getInputStream()));
        PrintWriter salida = new PrintWriter(new OutputStreamWriter(cliente.getOutputStream()));
        String dato = entrada.readLine().toString();
        historial.append(dtf.format(LocalDateTime.now()) + " Dan Bobadilla: " + dato + "\n");
        salida.print(dato.toString());
    }
} catch (Exception e) {
    e.printStackTrace();
}
```

Ilustración 6

Como podemos ver para realizar el programa utilizamos Sockets en este caso es el socket del servidor, este corre por el puerto 8000 de la computadora como si fuera en el caso de Apache (8080), el bucle while nos permite mantener escuchando el socket y al recibir una petición de acceso este la acepta y lo guarda en el socket del cliente, ahora obtenemos la IP del cliente para poder crear otro socket que le enviara información al celular, para leer los datos creamos un BufferedReader que nos permitirá leer el socket, y con esto lo pasamos al chat del JFrame.

Para enviar información esta se hace mediante un botón por lo cual necesitamos crear otro socket, además de que si utilizamos el mismo socket anterior no podríamos enviar múltiples mensajes al mismo tiempo seria turnarse para enviar un mensaje.

En este caso creamos un socket con la IP obtenida y le decimos por que puerto se envía, aquí cambiamos al 8001 porque no cerramos el primer socket.

```
try {
    Socket send = new Socket(ipDevice, 8001);
    PrintWriter print = new PrintWriter(send.getOutputStream());
    print.write(txtMsg.getText());
    historial.append(dtf.format(LocalDateTime.now()) + " Tú: " + txtMsg.getText() + "\n");
    print.flush();
    print.close();
    send.close();
    txtMsg.setText("");
} catch (IOException e) {
    e.printStackTrace();
}
```

Ilustración 7

El funcionamiento es el mismo en Android Studio, como esta basado en Java es muy sencillo de implementar en Android únicamente hay cosas que cambian por el SDK ya que es diferente al JDK

CONCLUSIÓN

Me pareció un proyecto muy interesante ya que nunca había trabajado con el protocolo TCP en java, únicamente lo trabaje en Lenguaje C, esto es bastante bueno ya que Angular y React JS utilizan Sockets para la comunicación en tiempo real y claramente la mayor parte de tiempo son utilizados para comunicarse con las páginas web (Computadoras) con los celulares.