***PRÁCTICA DE LABORATORIO #2***

***PROGRAMACIÓN DE APLICACIONES DE RED***

***SISTEMA DE GESTIÓN DE USUARIOS***

**Parte 1: Sistema de gestión de usuarios**

En esta práctica de laboratorio vamos a desarrollar una versión muy simple de un sistema de gestión de usuarios, con base en el sistema de eco desarrollado en la Práctica de Laboratorio #1, usando el protocolo TCP en la capa de transporte. Este sistema se compone de dos aplicaciones: una aplicación cliente que establece una conexión con una aplicación servidor y le envía un mensaje y recibe una respuesta. El usuario, en el lado del cliente, ingresa un nombre. El cliente crea un mensaje con una palabra clave seguida del nombre ingresado por teclado, separados por un espacio. Cuando llega el mensaje al servidor, el mensaje es procesado obteniendo el nombre que viene el mensaje. En el servidor se utiliza un hashmap para almacenar los usuarios del sistema. En este caso, se almacenan los nombres de los usuarios registrados y la cantidad de veces que ha ingresado al sistema.

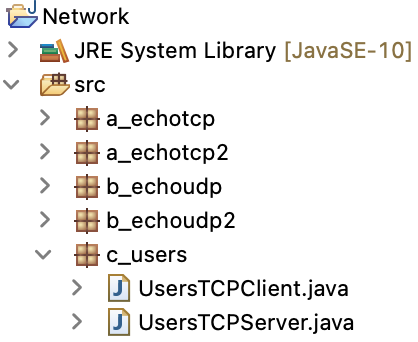
El mensaje enviado por el cliente es procesado en el servidor. La aplicación servidor recibe el mensaje y hace eco de ese mensaje enviándolo de nuevo a la aplicación cliente.



El protocolo de capa de aplicación de este sistema se puede resumir de la siguiente forma:



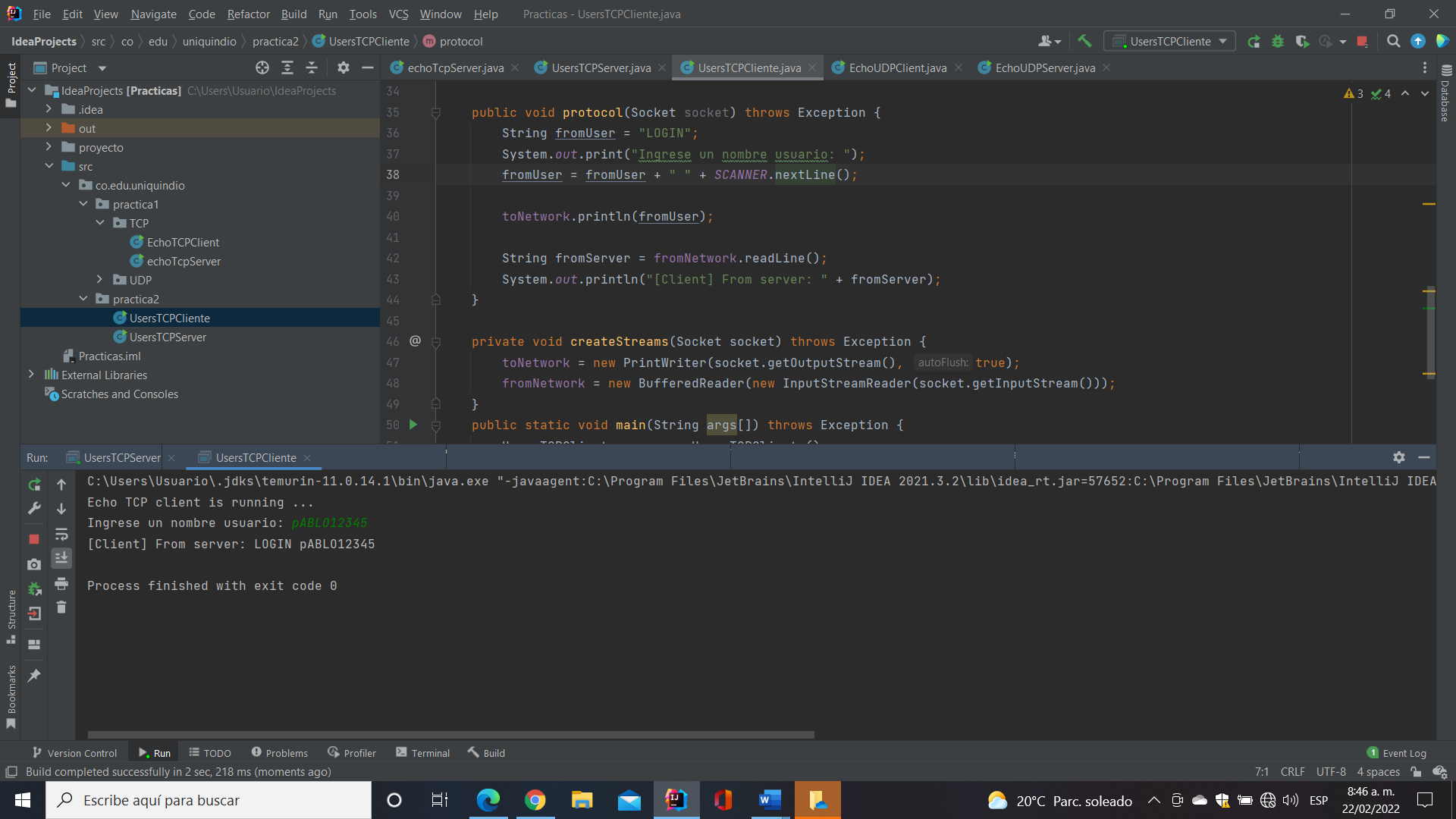
Cree un paquete adicional llamado **c\_users** a la estructura de paquetes de la práctica anterior.



**Programa Cliente**

Desarrolle la aplicación cliente con el siguiente protocolo:

* El sistema solicita el nombre al usuario para que lo ingrese por teclado.
* El sistema crea un mensaje con el siguiente formato: **"LOGIN"** seguido por el nombre del usuario ingresado por el teclado, separados por un espacio en blanco.
* El sistema queda esperando un mensaje de respuesta del servidor y lo imprime en la consola.



**Programa Servidor**

Desarrolle la aplicación servidor con el siguiente protocolo:

* El sistema utiliza un HashMap para almacenar los nombres de los usuarios que han sido registrados y la cantidad de veces que han ingresado luego de su registro inicial.
* El sistema inicia inicia esperando la llegada de un cliente.
* Cuando se conecta un cliente, el sistema queda esperando la llegada de un mensaje.
* Cuando el servidor recibe un mensaje, si inicia con la palabra **"LOGIN"**, extrae el nombre que viene en el mensaje. Si el nombre corresponde a un usuario nuevo, lo registra en el HashMap y responde al cliente con el mensaje **"BIENVENIDO fulano, usted es el usuario #n"**, donde **n** aumenta a medida que se registran usuarios en el sistema. Si el nombre ya está registrado, se incrementa la cantidad de ingresos al sistema y responde al cliente con el mensaje **"Fulano: Usted ha ingresado #k veces al sistema"**, donde **k** es la cantidad de veces que ha ingresado al sistema, sin tener en cuenta el registro en el sistema.

[Foto 02 – Agregue una captura de pantalla de la consola de la ejecución del servidor]

[Foto 03 – Agregue una captura de pantalla de la consola de la ejecución del cliente, luego de haber registrado varios usuarios y de haber hecho múltiples accesos]

**Ejercicio**

Haga una copia del proyecto y adicione las siguientes funcionalidades:

**Programa Cliente**

Desarrolle la aplicación cliente con el siguiente protocolo:

* El sistema solicita al usuario un comando para que lo ingrese por teclado.
* Si el comando es **".LOGIN"**, debe ir acompañado del nombre del usuario separados por un espacio en blanco.
* Otros dos comandos que pueden ser enviados son **".INFORME"** y **".INFORME\_DETALLADO"**, los cuales no llevan información adicional.
* El sistema queda esperando un mensaje de respuesta del servidor y lo imprime en la consola.

[Foto 04 – Agregue una captura de pantalla de la consola de la ejecución del cliente luego de usar el comando .LOGIN]

[Foto 05 – Agregue una captura de pantalla de la consola de la ejecución del cliente luego de usar el comando .INFORME cuando hay varios usuarios registrados]

[Foto 06 – Agregue una captura de pantalla de la consola de la ejecución del cliente luego de usar el comando .INFORME\_DETALLADO cuando hay múltiples usuarios registrados y múltiples accesos]

**Programa Servidor**

Desarrolle la aplicación servidor con el siguiente protocolo:

* El sistema utiliza un HashMap para almacenar los nombres de los usuarios que han sido registrados y la cantidad de veces que han ingresado luego de su registro inicial.
* El sistema inicia inicia esperando la llegada de un cliente.
* Cuando se conecta un cliente, el sistema queda esperando la llegada de un mensaje.
* Cuando el servidor recibe un mensaje, si inicia con el comando **".LOGIN"**, extrae el nombre que viene en el mensaje. Si el nombre corresponde a un usuario nuevo, lo registra en el HashMap y responde al cliente con el mensaje **"BIENVENIDO fulano, usted es el usuario #n"**, donde **n** aumenta a medida que se registran usuarios en el sistema. Si el nombre ya está registrado, se incrementa la cantidad de ingresos al sistema y responde al cliente con el mensaje **"Acceso #k"**, donde **k** es la cantidad de veces que ha ingresado al sistema, sin tener en cuenta el registro en el sistema.
* Si el comando es **".INFORME"**, el servidor enviará como respuesta la lista de usuarios registrados en el sistema, separados por coma. Ejemplo: **"juan, ana, luis, jose"**
* Si el comando es **".INFORME\_DETALLADO"**, el servidor enviará como respuesta la lista de usuarios registrados en el sistema, incluyendo la cantidad de veces que el usuario ha ingresado al sistema con el siguiente formato: **"juan 4, ana 1, luis 10, jose 3"**

**Parte 2: Comunicación con un servidor en redes caseras**

**Ngrok** es una herramienta que permite exponer aplicaciones tipo servidor que se ejecutan en redes locales detrás de dispositivos NAT y/o firewalls, de manera que aplicaciones tipo cliente se puedan conectar a través de Internet usando túneles seguros.

En esta actividad vamos a aprender a configurar **ngrok** para ejecutar el sistema de eco vista en clase.

**Creación de la cuenta y ejecución de la aplicación cliente-servidor de eco en un solo computador**

1. Descargue la aplicación **ngrok** en su computador. Ingrese a: <https://ngrok.com/download> y siga las instrucciones.
2. Cree una cuenta en **ngrok** y obtenga su token de autenticación.
3. En su computador, en una consola, ejecute el comando **ngrok authtoken** seguido del token asignado.
4. Exponga el puerto utilizado para el programa servidor de eco, ejecutando el comando ﻿**ngrok tcp 3400**. Observe la dirección y puerto asignados para el puerto que acaba de exponer en su computador.

[Foto 07 – Agregue una captura de pantalla de la terminal luego de exponer el puerto]

1. Ejecute el programa servidor de eco sin modificar nada. Asegúrese de que el puerto que se acaba de exponer sea el puerto que utiliza el servidor para escuchar.
2. Modifique el código fuente del programa cliente de eco, de manera que la dirección donde va a encontrar el servidor en ejecución sea la que estableció **ngrok** en el numeral 4. Por ejemplo: **6.tcp.ngrok.io** y el número de puerto, por ejemplo **15109**. Ejecute el programa cliente.

[Foto 08 – Agregue una captura de pantalla que incluya las modificaciones al código fuente en el programa cliente y la consola al finalizar la ejecución del programa cliente]

[Foto 09 – Agregue una captura de pantalla de la consola al finalizar la ejecución del programa servidor]

**Ejecución de la aplicación cliente-servidor de eco en dos computadores**

En grupos, seleccione cuál de los compañeros va a ejecutar el programa servidor utilizando el puerto que expuso en el numeral 4 de la parte 1 de este laboratorio. El compañero que va a ejecutar el servidor debe compartir la dirección y el puerto asignados por **ngrok**.

Los demás compañeros, van a ejecutar el programa cliente, en la misma forma como lo hicieron en el numeral 6 de la parte 1 de este laboratorio.

Por cada integrante del grupo,

[Foto 10 – Agregue una imagen de la consola al finalizar la ejecución del programa cliente]

[Foto 11 – Agregue una imagen de la consola al finalizar la ejecución del programa servidor]