Programacion de Redes

Obligatorio 1

Martin Gulla - 254564
Joaquin Sommer - 184441

Profesores:
Gabriel Bentos
Andres Soria

Martin Gulla - 254564
Joaquin Sommer - 184441

Descripción de la arquitectura

Los proyectos fueron desarrollados en la version .net 6.0.
Nuestra aplicación se basa en 4 proyectos, uno para la aplicación del servidor, el cual está constantemente esperando conexiones de usuarios en una ip y puerto especificados en un archivo. Otro es la aplicación cliente, la cual es la que inicia la comunicación con el servidor, esta también tiene un puerto que se puede cambiar a gusto. Los otros dos proyectos son de utilidades comunes entre ambas aplicaciones, como por ejemplo, el envío de imágenes.

Decisiones a nivel de protocolo

En cuanto al protocolo entre cliente-servidor, no nos parecio necesario el uso del header que identifica si el mensaje es de peticion (“REQ”) o de respuesta (“RES”), ya que en la logica de nuestro codigo, siempre el servidor espera peticiones y el cliente respuestas, nunca es en el otro sentido, por lo tanto decidimos quitarlo. Los mensajes incluyen 2 Headers, primero uno que envia un numero entre 01 y 99 (de largo 2), el cual identifica el comando al que se esta queriendo acceder, también, en el caso de los mensajes enviados por el servidor, si ocurrio un problema, este devuelve 00, codigo reservado para errores, el cliente verifica si se recibio este codigo y muestra un error en pantalla. Luego viene el header del largo del mensaje, el cual es un numero entre 0000 y 9999. Por ultimo, esta el mensaje, el cual es texto plano.
Los comandos pueden tener partes divididas por pipes (“|”), e incluso subdivididas por numerales (“#”), como es el caso del alta de perfil de trabajo, un mensaje de este comando puede ser del tipo: “1|descripcion|programador#tester#muchas otras cosas”, el cual indica que es del usuario 1, con una descripcion “descripcion”, y sus habilidades son programador, tester, y “muchas otras cosas”.

Manejo de los errores

Utilizamos bloques try-catch en los lugares criticos donde pueden haber errores, por ejemplo, al momento de enviar archivos, leer mensajes de comunicacion entre servidor-cliente. Al momento de atrapar estos errores, devolvemos un codigo al cliente que significa que algo ocurrio mal, asi el usuario entiende porque no ocurrio lo esperado.

Mecanismos de concurrencia utilizados

El mecanismo de concurrencia utilizado en nuestro obligatorio fue los Locks. En nuestro servidor las únicas variables globales que debíamos tener cuidado son la lista de Usuarios y la lista de Mensajes. Estas dos variables son leídas y escritas por el servidor mientras está en comunicación con los distintos clientes. Por esta razón es que es necesario utilizar un lock cuando accedemos a leer o escribir alguna de ellas.
La clase Sistema de nuestro servidor tiene estas dos listas y todos los métodos que escriben y leen las listas están en esta clase. Entonces es aquí donde agregamos locks a los métodos para bloquear que no entren dos al mismo tiempo. Ejemplo:

```
public string GuardarPathFoto(int id, string extension)
{
    lock (Usuarios)
    {
        User user = BuscarUsuario(id);
        if (user != null)
        {
            string path = "Fotos\\" + user.Username + "." + extension;
            user.pathFoto = Path.Combine(Directory.GetCurrentDirectory(), path);
            return "Foto guardada";
        }
        return "No se encontro el usuario";
    }
}
```

Funcionamiento de la aplicación

La aplicación está cargada con dos usuarios de prueba.

- Nombre: **admin** Psw: **admin**
- Nombre: **user** Psw: **user**

Flujo Login

```
Iniciando Aplicacion Cliente....!!!
Cliente Conectado al Servidor...!!!
Menu
1. Login
2. Registrarse
3. Desconectarse
1
Ingrese su usuario
admin
Ingrese su contraseña
admin
Login exitoso, bienvenido admin
```

```
Menu
1. Alta perfil de trabajo
2. Asociar foto al perfil de trabajo
3. Listar perfiles de trabajo
4. Consultar perfil especifico
5. Enviar mensaje
6. Listar mensajes
7. Cerrar sesion
```

Registrarse

```
Iniciando Aplicacion Cliente....!!!
Cliente Conectado al Servidor...!!!
Menu
1. Login
2. Registrarse
3. Desconectarse
2
Ingrese nuevo nombre de usuario
juan
Ingrese contraseña
juan
```

Alta de perfil

```
Menu
1. Alta perfil de trabajo
2. Asociar foto al perfil de trabajo
3. Listar perfiles de trabajo
4. Consultar perfil especifico
5. Enviar mensaje
6. Listar mensajes
7. Cerrar sesion
1
Ingrese descripcion del perfil
Desarrollador
Ingrese habilidad
React
Desea agregar otra habilidad? (s/n)
s
Ingrese habilidad
Vue
Desea agregar otra habilidad? (s/n)
n
```

Asociar foto de Perfil

```
Menu
1. Alta perfil de trabajo
2. Asociar foto al perfil de trabajo
3. Listar perfiles de trabajo
4. Consultar perfil especifico
5. Enviar mensaje
6. Listar mensajes
7. Cerrar sesion
2
Ingrese la ruta completa al archivo:
C:\Users\somme\Desktop\Redes2\programacion-de-redes\Protocolo\El-fantasma-de-la-opera-gaston-leroux.jpeg
Se envio el archivo al Servidor
```

Flujo para descargar foto de perfil

```
1. Alta perfil de trabajo
2. Asociar foto al perfil de trabajo
3. Listar perfiles de trabajo
4. Consultar perfil especifico
5. Enviar mensaje
6. Listar mensajes
7. Cerrar sesion
4
Ingrese el id del perfil
1
Perfil encontrado:
<====== Inicio de perfil ======>
Id: 1
Nombre: admin
Descripcion: Administrador
Habilidades: Administrador
<======= Fin del perfil =======>
Desea descargar la foto de perfil? (s/n)
s
Foto de perfil descargada
```

Martin Gulla - 254564
Joaquin Sommer - 184441

Decisiones a nivel de aplicacion

Los usuarios pueden registrarse con una contraseña vacia.
Los usuarios solo pueden registrar su perfil de trabajo, si ya tienen uno activo, el servidor responde con error.
Los usuarios pueden asociar una foto de perfil, incluso cuando no dieron de alta su perfil de trabajo.
Al pedir todos los usuarios con perfil de trabajo, se puede filtrar por nombre, descripcion, habilidades, o ningun filtro. Se devuelve una lista con todos los perfiles, con su Id, nombre, descripcion y habilidades.
Consultar perfil especifico pide un id de usuario, y devuelve el usuario en caso de encontrarlo, con el formato anterios.
Al enviar mensaje, se ingresa el nombre al usuario al que se le quiere enviar, y luego el mensaje.
Para listar mensajes, primero te muestra la cantidad que tenes, y luego pide que ingreses el nombre de un usuario para ver todos los mensajes con ese usuario (enviados y recibidos).