Universidad ORT

Uruguay Facultad de Ingeniería

Programación de Redes

Obligatorio I

Clavijo Vitale, Tomás Andrés (235426)

Lopez Pérez, Lucas Uriel (257696)

Docentes: Gabriel Bentos, Andres Soria, Martín Olazabal

2022

ÍNDICE:

Descripción del Sistema

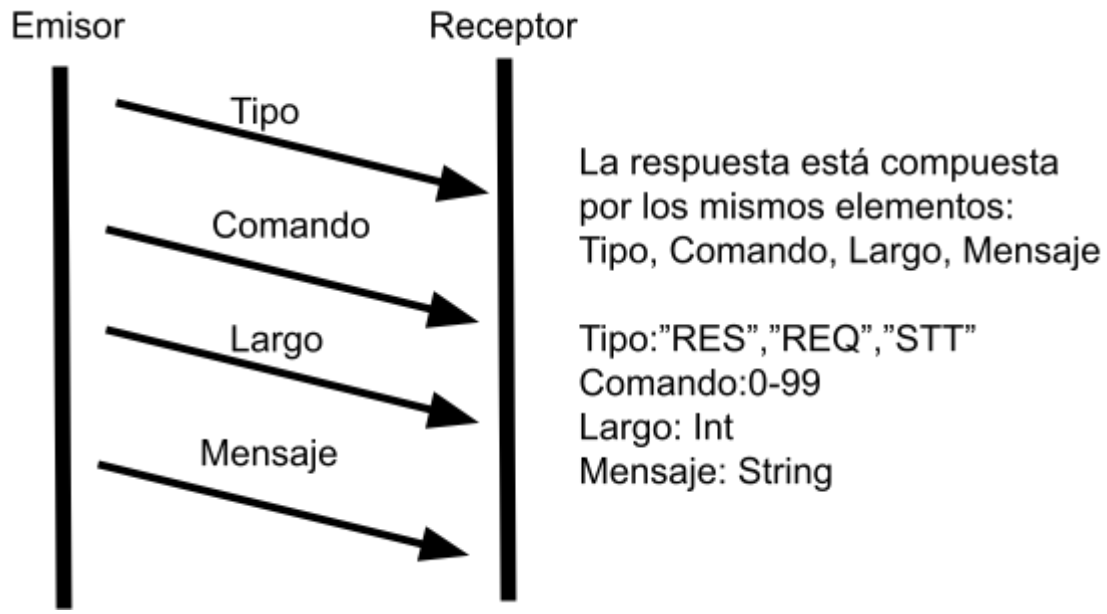
LKdin es un sistema que cuenta con dos aplicaciones, un servidor y un cliente. El servidor maneja la información relacionada a usuarios y sus perfiles laborales, así como también los archivos que esto implica. Por otra parte, la aplicación cliente se encargará de la interacción de los usuarios con el sistema.

Tecnología utilizada

.NET 5.0: Marco de software informático administrado gratuito, de código abierto y alto rendimiento diseñado por Microsoft.

Visual Studio: Es un entorno de desarrollo integrado (IDE) de Microsoft. Se utiliza para desarrollar programas informáticos, así como también sitios web, aplicaciones web, servicios web y aplicaciones móviles.

Protocolo



Nombre Campo	HEADER	CMD	LARGO	MENSAJE
Valores	"RES"/"REQ"/"STT"	0-99	Int	String
Largo	3	2	4	Variable

Es un protocolo orientado a caracteres donde los mensajes siempre son esperados en dicho formato.
RES: Respuesta del servidor a datos esperados por ell cliente
REQ: Petición del cliente
STT: Devolución del servidor sobre el estado de la petición

Control de concurrencia

En un principio decidimos implementar estructuras con monitores con formato de solución, lectores y escritores, de forma que varios pudieran hacer consultas de perfiles en simultáneo. Y aunque llegamos a implementar la clase para ello, decidimos dejarlo fuera pues no era un requisito de esta entrega. Utilizamos finalmente locks, que al ser de alto nivel tenemos menos margen de equivocarnos y nos permite liberar el recurso en caso de salidas con excepciones, entonces dependiendo la función el objeto que recibe el lock es aquel que deseamos entrar en mutua exclusión.
Ejemplo:

```
lock(Perfiles){
        ReadPerfiles(Perfiles);
}
:
:
lock(Perfiles){
        AgregarPerfil(Perfil, Perfiles);
}
```

Control de excepciones

Utilizamos bloques try and catch para realizar nuestras operaciones y de esta forma controlar nuestras propias excepciones por ejemplo Argument Exception cuando intentas agregar caracteres especiales a nombres, o propias de las herramientas que estamos usando, ejemplo Socket Exception para cuando desconectamos el servidor con clientes todavía activo.

Casos de prueba

Procederemos a detallar los distintos errores que el sistema controla, así como también los casos donde la información que se le brinda es la esperada.

Caso 1: Nombre que posee números.

Datos ingresados:

2
Tomas410
Tomas410
contraseña

Se controla que el nombre no tenga números u otros símbolos, por lo que en caso de presentarlos, si bien permite que se ingresen los datos correspondientes al usuario de forma completa, devuelve “Nombre no válido”, no pudiendo registrarlo.

Caso 2: Campos vacíos

Datos ingresados:

2	2	2
	Tomás	Tomás
Tomas410		Tomas410
contraseña	contraseña	

En todos los casos se controla que no falten datos correspondientes al usuario. Obteniendo como respuestas: “Nombre no válido”, “La password es un campo obligatorio”, “Faltaron datos”, “El campo nombre de usuario es obligatorio”.

Caso 3: Usuario existente

2	2
Tomas Clavijo	Tomas Edison
Tomas410	Tomas410
contraseña	contraseña

El sistema no permite que dos usuarios se registren con el mismo nombre de usuario, incluso si el cambio entre un nombre de usuario y otro es únicamente mayúscula/minúscula, como podría ser ingresar: “tomas410”. A esto devuelve “El usuario ya existe” y retorna al menú.

Caso 4: Creación de perfil, sin registro previo de usuario

3
Profesional responsable y eficiente, con capacidad de trabajo en equipo.
Haskell
x

No se permite la creación de un perfil sin haber registrado previamente al usuario. Retorna como mensaje “No está autorizado a realizar esta operación”, volviendo al menú.

Caso 5: Creación de usuario y perfil exitosa

2	3
Lucas	Docente de Sistemas Operativos
UserLucasLopez	Magnánimo
Lucas1111	x

Crea el usuario y posteriormente el perfil. Retornando “Usuario registrado correctamente” y “Perfil creado correctamente”.

Caso 6: Descripción vacía

2	3
Lucas	
UserLucasLopez	Magnánimo
Lucas1111	x

Controla que la descripción no se encuentre vacía retornando como mensaje: “El campo de la descripción es obligatorio”.

Caso 7: Habilidades vacías

2	3
Lucas	Docente de Sistemas Operativos
UserLucasLopez	
Lucas1111	x

En caso de ingresar una habilidad vacía y luego enviar “x” para finalizar, o no ingresar habilidad alguna y presionar “x” en la primera interacción, el sistema volverá a devolver “Ingrese una habilidad o (X) para salir”, no permitiéndole al usuario continuar, ya que es requisito tener al menos una habilidad.

Caso 8: Asociar imagen vacía

Una vez creado el usuario y perfil, podemos asociarle la imagen. Si el peso de la imagen es de 0 bytes, como puede ser, un archivo de texto vacío al cual se le cambia la extensión por .png, el sistema no la asociará, y devolverá “Imagen invalida”

Caso 9: Asociar imagen correcta

Similar al caso anterior, pero devuelve “Imagen cargada correctamente”.

Caso 10: Búsqueda por nombre

Partiendo del ejemplo del caso (3):

5
1
Tomas

Devolverá a ambos usuarios, brindando sus nombres completos, descripciones y habilidades correspondientes. Posteriormente retorna al menú.

Caso 11: Búsqueda por habilidad

Partiendo del ejemplo del caso (5):

5
1
Magnanimo

Devuelve los datos correspondiente al perfil de Lucas López, nombre, descripción y habilidades asociadas.

Caso 12: Búsqueda por username

Partiendo del ejemplo del caso (3), considerando únicamente la primer columna:

5
3
Tomas410

Devuelve los datos correspondiente al perfil del usuario. Se adjunta la información de si tiene foto asociada, junto al texto: “¿Desea descargar la imagen de perfil?” En caso de escribir (S), la imagen se descarga en la carpeta indicada por el usuario.

Caso 13: Búsqueda fallida

5	5	5
1	2	3
Lucas	Amable	Lucas

En caso de realizar cualquiera de las tres búsquedas con datos que no existen, se retornará al usuario al menú principal, retornando únicamente “Perfiles:” y un espacio en blanco, o directamente “El perfil no existe”; esto depende de la búsqueda que se esté aplicando.

Caso 14: Enviar y recibir mensajes

Creación de usuarios y perfiles para la interacción:

2	2
Lucas Lopez	Tomas Clavijo
Lucas6119	Tomas410
Contraseña	Contraseña
3	3
Docente de Sistemas Operativos	Docente de Logica para Computacion
Magnanimo	Fernetero
x	x

Envío de mensaje de parte de Lucas a Tomás:

6
1
Tomas410
Hola, todo bien?

Recibo de mensaje:

6
2
1

Se imprime “Mensaje de:”, seguido del nombre de la persona que lo envió y el contenido del mismo. En caso de consultar por él nuevamente, deberá elegir como último paso la segunda opción, ya que automáticamente queda como leído.