



**Instituto Tecnológico de Costa Rica**  
Escuela de Ingeniería en Computación

Lenguajes de Programación

Proyecto 1: Programación Imperativa

Alumnos:

Andrés Bonilla Solano | 2023101220  
Emily Sánchez Orozco | 2021067314

I Semestre 2025

## Índice

Enlace de GitHub: .....	3
Pasos de instalación del programa .....	4
Manual de usuario .....	5
Arquitectura lógica utilizada .....	10
Flujo de Datos e Interacción entre cliente y servidor .....	11
Librerías adicionales .....	11

**Enlace de GitHub:**

[https://github.com/AndyBS1/Proyecto1\\_Lenguajes.git](https://github.com/AndyBS1/Proyecto1_Lenguajes.git)

## Pasos de instalación del programa

### Requisitos previos

Antes de compilar y ejecutar el programa, asegúrese de contar con lo siguiente:

- Un sistema operativo basado en Linux (probado en Ubuntu).
- Un compilador de C++ (como g++).
- Acceso a terminal.
- Permisos para crear y leer archivos locales

Para instalar la librería adicional se deben ejecutar los siguientes comandos en la terminal de Linux:

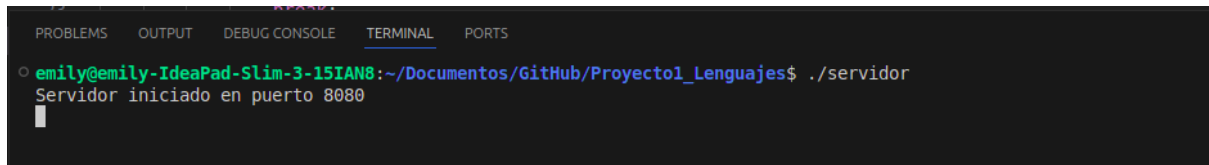
```
sudo apt update
sudo apt install qtbase5-dev qtcreator
sudo apt install pkgconf
sudo apt install qtbase5-dev qtchooser qttools5-dev-tools
qttools5-dev qtscript5-dev
sudo apt install mesa-utils
```

## Manual de usuario

### Ingresar al servidor:

Se debe ejecutar el servidor.cpp en la terminal y si se ejecuta correctamente debe de mostrar un mensaje: “Servidor iniciado en puerto 8080”

Para ejecutar el servidor solo debe colocar en la terminal ./servidor, como se muestra en la siguiente imagen:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
○ emily@emily-IdeaPad-Slim-3-15IAN8:~/Documentos/GitHub/Proyecto1_Lenguajes$ ./servidor
Servidor iniciado en puerto 8080
```

Si no se conecta correctamente el sistema le mostrará un error, por ejemplo:

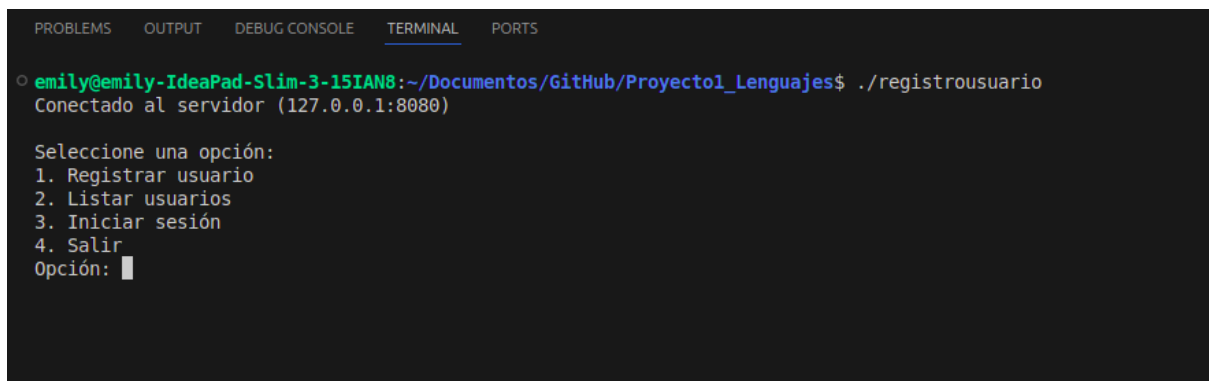
“Error creando el socket”

“Error conectando el servidor”

### Funcionalidades:

Para acceder a las funcionalidades del sistema, deberá ejecutar el archivo registrousuari.cpp en otra terminal.

Para ejecutar este archivo solo debe colocar en la terminal ./registrousuari, como se muestra en la siguiente imagen:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
○ emily@emily-IdeaPad-Slim-3-15IAN8:~/Documentos/GitHub/Proyecto1_Lenguajes$ ./registrousuari
Conectado al servidor (127.0.0.1:8080)

Seleccione una opción:
1. Registrar usuario
2. Listar usuarios
3. Iniciar sesión
4. Salir
Opción: 
```

Si funciona correctamente le deberá aparecer que se conectó al servidor y además se le deberá desplegar el menú de opciones.

## Registro usuario:

Para registrar un usuario deberá escribir 1 y darle ENTER:

```
Opción: 1
Nombre de usuario: █
```

Luego deberá escribir su nombre de usuario, tome en cuenta que si escribe un nombre de usuario que ya fue utilizado no le permitirá registrarse, el nombre de usuario debe ser único para que le sea más fácil encontrar a los usuarios a los que le desea enviar un mensaje y no hayan repetidos.

```
Opción: 1
Nombre de usuario: Emily23
Contraseña: █
```

Cuando ya ingreso su nombre de usuario, deberá ingresar su contraseña, la contraseña no tiene ninguna restricción en general.

```
Opción: 1
Nombre de usuario: Emily23
Contraseña: 2304
Respuesta del servidor: OK|Registro exitoso|PUERTO|5003
```

Si el registro fue completado con éxito le deberá de aparecer ese mensaje, además ahí podrá ver en qué puerto le fue asignado por el sistema.

En caso de que haya errores el sistema se los deberá mostrar señalándole el error.

```
Opción: 1
Nombre de usuario: Emily23
Contraseña: 7896
Respuesta del servidor: ERROR|Usuario ya existe
```

Además, para asegurarse que si está conectado al servidor se le desplegara la opción escogida registro y la información del usuario, para corroborar que si se está registrando en el servidor.

```
emily@emily-IdeaPad-Slim-3-15IAN8:~/Documentos/GitHub/Proyecto1_Lenguajes$ ./servidor
Servidor iniciado en puerto 8080
Error leyendo datos o conexión cerrada
Mensaje recibido: REGISTRO|Emily23|127.0.0.1|2304
Usuario registrado: Emily23 IP: 127.0.0.1 Puerto: 5003
█
```

### Lista de usuarios registrados:

Para ver la lista de usuarios que están registrados en el sistema, deberá escribir 2 en el menú y dar ENTER:

```
Seleccione una opción:  
1. Registrar usuario  
2. Listar usuarios  
3. Iniciar sesión  
Opción: 2  
Usuarios registrados: USUARIOS|Andres|Carlos|Emily23|Maria
```

Y se le desplegará la lista de usuarios, solo podrá ver los nombres de usuario.

### Iniciar sesión:

Para iniciar sesión debe escribir 3 en el menú y dar ENTER:

```
Seleccione una opción:  
1. Registrar usuario  
2. Listar usuarios  
3. Iniciar sesión  
Opción: 3  
Nombre de usuario: 
```

Deberá de ingresar su nombre de usuario:

```
3. Iniciar sesión  
Opción: 3  
Nombre de usuario: Emily23  
Contraseña: 
```

Y luego ingresar su contraseña:

```
Opción: 3  
Nombre de usuario: Emily23  
Contraseña: 456
```

En el caso de que las credenciales sean incorrectas se le mostrara un error.

```
3. Iniciar sesión  
Opción: 3  
Nombre de usuario: Emily23  
Contraseña: 456  
Respuesta del servidor: ERROR|Credenciales inválidas
```

### Envío de mensajes:

Para enviar mensajes debe escribir en otra terminal ./Cliente:

```
emily@emily-IdeaPad-Slim-3-15IAN8:~/Documentos/GitHub/Proyecto1_Lenguajes$ ./Cliente  
Usuario: 
```

Deberá escribir su nombre de usuario:

```
emily@emily-IdeaPad-Slim-3-15IAN8:~/Documentos/GitHub/Proyecto1_Lenguajes$ ./Cliente
Usuario: Emily23
Contraseña: █
```

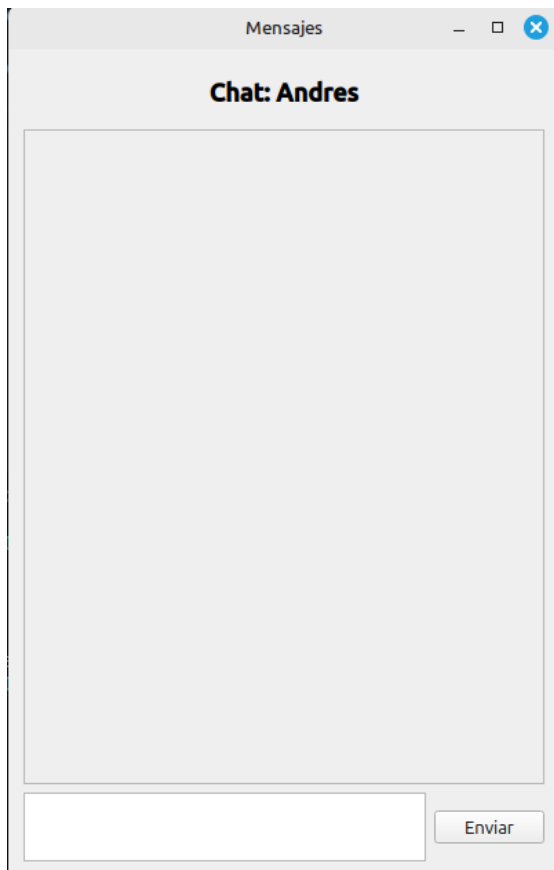
Luego escribir su contraseña:

```
emily@emily-IdeaPad-Slim-3-15IAN8:~/Documentos/GitHub/Proyecto1_Lenguajes$ ./Cliente
Usuario: Emily23
Contraseña: 456
Puerto asignado: 5004
```

Y por último escribir al usuario al que le enviara el mensaje

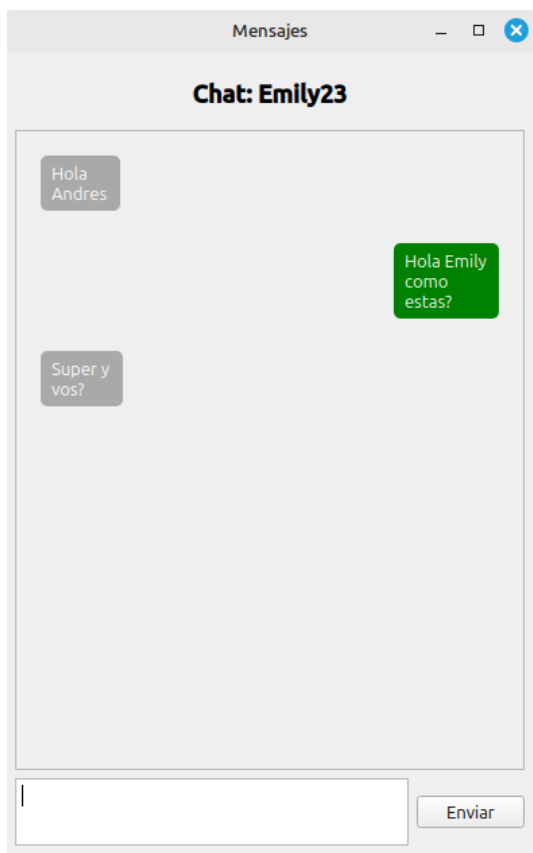
```
emily@emily-IdeaPad-Slim-3-15IAN8:~/Documentos/GitHub/Proyecto1_Lenguajes$ ./Cliente
Usuario: Emily23
Contraseña: 456
Puerto asignado: 5004
Para: Andres
```

Y le deberá aparecer esta pantalla:





Para enviar el mensaje solo lo escribe y le da enviar en el botón de abajo a la izquierda:



También puede ver los mensajes en la terminal del servidor:

```
Mensaje recibido: MENSAJE|Emily23|Andres|Hola Andres
Usuario destino: Andres
Mensaje: Hola Andres
Conectando al destinatario: Andres IP: 127.0.0.1 Puerto: 5000
Mensaje final a enviar: MENSAJE|De: Emily23|A: Andres|Mensaje: Hola Andres
Mensaje enviado de: Emily23 a: Andres -> Hola Andres
Mensaje recibido: MENSAJE|Andres|Emily23|Hola Emily como estas?
Usuario destino: Emily23
Mensaje: Hola Emily como estas?
Mensaje recibido: MENSAJE|Emily23|Andres|Super y vos?
Usuario destino: Andres
Mensaje: Super y vos?
Conectando al destinatario: Andres IP: 127.0.0.1 Puerto: 5000
Mensaje final a enviar: MENSAJE|De: Emily23|A: Andres|Mensaje: Super y vos?
Mensaje enviado de: Emily23 a: Andres -> Super y vos?
```

## Arquitectura lógica utilizada

El sistema está compuesto por varios módulos.

- Servidor:

El servidor va a estar escuchando en el puerto 8080 donde recibe las conexiones de los clientes. Esto permite que se pueda enviar y recibir mensajes desde múltiples conexiones. Para gestionar estas conexiones, se utiliza un fork(). Cuando un cliente envía un mensaje, el servidor lo recibe, lo procesa y lo distribuye al usuario que fue definido como destinatario. Además, el servidor puede manejar funcionalidades adicionales, como la gestión de conexiones o el registro de actividades de los usuarios.

- Registro de Usuarios: Para el registro de usuario es necesario que ya se haya establecido la conexión con el servidor, el usuario selecciona la opción de registro y se le solicita que ingrese un nombre de usuario que debe ser único y una contraseña que no tiene restricciones y con la dirección IP local del usuario se concatenan con el siguiente formato:

REGISTRO|<usuario>|<ip>|<contrasena>.

Si el nombre de usuario no está registrado, el servidor asigna un puerto al nuevo usuario, este puerto se elige comenzando desde el 5000 y se va incrementando hasta encontrar uno disponible que no haya sido asignado previamente a otro usuario.

- Cliente

En este módulo es donde se tienen las funcionalidades principales. El usuario puede enviar mensajes y recibir mensajes mediante el uso de una interfaz gráfica. Dentro de la interfaz, los mensajes enviados aparecen de color verde, mientras que los recibidos en color gris. Además, en la parte superior aparece el nombre del usuario con el que se está interactuando.

En cuanto a la estructura de los documentos, para la parte de cliente se tienen archivo de código fuente de C++(.cpp) que contiene la lógica principal del programa y los archivo header(.h) que contienen las declaraciones de las clases, funciones y variables globales que serán utilizadas en los archivos .cpp. En la parte del servidor

y el registro de usuarios solamente se tienen los .cpp. Finalmente, los otros archivos que se encuentran en la carpeta del proyecto son los necesarios para compilar y ejecutar el proyecto ya que se utiliza QT. (Estos archivos son: Makefile, cliente.pro y los archivos con la extensión .o y los ejecutables)

### Flujo de Datos e Interacción entre cliente y servidor

- El usuario se registra y envía la información al servidor. El servidor verifica que la información es válida y la añade al documento donde se almacenan los usuarios. Además, le asigna un puerto.
- De manera opcional se pueden listar los usuarios registrados. Al realizar este comando el servidor recibe una notificación.
- El usuario utiliza el módulo de cliente donde envía un mensaje, el mensaje se envía al servidor a través de un socket.
- El servidor recibe el mensaje y lo procesa. Luego, lo reenvía al usuario que se definió como destino.
- Finalmente, los mensajes se muestran en la interfaz.

### Librerías adicionales

En este caso solamente se utilizó una librería externa. Esta es **QT Creator** que fue utilizada para realizar la interfaz gráfica. En resumen, Qt es un *framework* multiplataforma ampliamente utilizado para desarrollar interfaces gráficas de usuario (GUI) en C++. Permite crear aplicaciones utilizando distintos componentes y *widgets* de forma sencilla y fácil de personalizar.