1. Introducción

El presente documento detalla la entrega final del proyecto "Sistema de Chat Institucional". Esta solución es una aplicación cliente-servidor desarrollada en Python, diseñada para facilitar la comunicación segura y eficiente en entornos educativos y corporativos sin dependencia de internet externo.

El sistema ha evolucionado desde un chat básico de sockets hasta convertirse en una plataforma robusta que integra **concurrencia multihilo**, **cifrado SSL/TLS**, **base de datos SQLite** para persistencia y un módulo de **Inteligencia Artificial (Ollama)** para asistencia y resúmenes de conversaciones.

2. Objetivo General

Desarrollar un sistema de mensajería instantánea escalable y seguro para redes locales (LAN) que permita la colaboración en tiempo real mediante salas temáticas, roles de usuario diferenciados (Estudiante, Profesor, Administrador) y herramientas de moderación, garantizando la integridad de los datos y la disponibilidad del servicio en entornos controlados como laboratorios o aulas.

3. Arquitectura Técnica del Sistema

Basado en el *Manual Técnico* adjunto, la arquitectura del sistema se ha profesionalizado en los siguientes módulos:

3.1 Tecnologías Core

- **Lenguaje:** Python 3.8+
- **Comunicación:** Sockets TCP/IP y WebSockets para tiempo real.
- **Seguridad:** Implementación de SSL/TLS (certificados server.crt y server.key) para cifrado de extremo a extremo.

- **Base de Datos:** SQLite para el almacenamiento estructurado de usuarios, roles, historiales de chat y configuraciones.

3.2 Módulos del Servidor

1. **Gestor de Conexiones:** Maneja múltiples hilos (threading) para soportar decenas de usuarios concurrentes sin bloqueo.
2. **Motor de IA (Ollama):** Integración local de modelos de lenguaje (LLM) para generar resúmenes automáticos de conversaciones y responder dudas técnicas bajo demanda.
3. **Sistema de Autenticación:** Verificación de credenciales con *hashing* seguro (SHA-256) y recuperación mediante preguntas de seguridad.

3.3 Módulos del Cliente

- **Interfaz Gráfica (GUI):** Desarrollada con customtkinter para una experiencia moderna y responsiva.
- **Gestor de Red:** Hilo independiente para la escucha de mensajes entrantes, evitando el congelamiento de la interfaz.

4. Funcionalidades de Usuario e Implementación

Según el *Manual de Usuario*, se han implementado las siguientes características clave:

4.1 Roles y Permisos

- **Estudiante:** Acceso a salas, envío de mensajes y consulta de usuarios conectados.
- **Profesor/Moderador:** Capacidades extendidas como silenciar usuarios (/mute), expulsar (/kick), fijar mensajes importantes (/pin) y realizar anuncios globales (/anuncio).

- **Administrador:** Control total de la gestión de cuentas, creación de salas y mantenimiento del servidor.

4.2 Comandos del Sistema

Se implementó un sistema de comandos de barra (/) para facilitar la interacción:

- /join <sala>: Cambiar de sala.
- /help: Mostrar ayuda contextual.
- /mirol: Consultar permisos actuales.
- /salas: Listar grupos disponibles.

5. Cronograma de Desarrollo e Implementación

A continuación se detalla el registro de cambios y la evolución del proyecto a través de los sprints definidos en la planeación.

Fecha de modificación	Archivo	Sprint	Tipo de Cambio	Componente	Realizado por
15-OCT-2025	Host 0.0.3.py	1	**Chat Simple (Semana 5):** Comunicación básica cliente-servidor con sockets TCP.	Backend	Díaz del Castillo Nava Enrique Mejía Olivares Bruno Eduardo Angulo Aceves Juan Manuel
16-OCT-2025	chat_app.py	1	**Chat Multihilo (Semana 6):** Implementación de threading para múltiples clientes paralelos.	Backend	Díaz del Castillo Nava Enrique Mejía Olivares Bruno Eduardo Angulo Aceves Juan Manuel
8-NOV-2025	Host 0.0.3.py	2	**Robustez y Alias (Semana 7):** Asignación de	Backend	Díaz del Castillo Nava Enrique

			alias y manejo de desconexiones limpias.		Mejía Olivares Bruno Eduardo Angulo Aceves Juan Manuel
10-NOV-2025	Host 0.0.3.py	2	**Registro/Login (Semana 8):** Autenticación, Hashing SHA256 y persistencia de usuarios.	Backend	Díaz del Castillo Nava Enrique Mejía Olivares Bruno Eduardo Angulo Aceves Juan Manuel
13-NOV-2025	chat_app.py	3	**Salas de Chat (Semana 9):** Lógica para comandos /join, /list y segmentación de salas.	Backend	Díaz del Castillo Nava Enrique Mejía Olivares Bruno Eduardo

14-NOV-2025	Historial.json	3	**Historial (Semana 10):** Integración de BD para guardar y recuperar historial de mensajes.	Database	Ríos Márquez Brandon Díaz del Castillo Nava Enrique Mejía Olivares Bruno Eduardo
20-NOV-2025	dockerfile	4	**Seguridad Raspberry (Semana 11):** Configuración de firewall, SSH seguro y puertos.	SysAdmin	Angulo Aceves Juan Manuel
21-NOV-2025	Host 0.0.3.py	4	**Cifrado (Semana 12):** Implementación de certificados SSL/TLS para tráfico seguro.	Security	Angulo Aceves Juan Manuel Díaz del Castillo Nava Enrique Mejía Olivares Bruno

					Eduardo
27-DEC-2025	Host 0.0.3.py	5	**Roles y Permisos (Semana 13):** Lógica de moderación (Admin/Profesor) y comandos /kick, /ban.	Backend	Díaz del Castillo Nava Enrique Mejía Olivares Bruno Eduardo
28-DEC-2025	design_constants.py	5	**Interfaz Gráfica (Semana 14):** Desarrollo de GUI con customtkinter (reemplazo de terminal).	Frontend	Ríos Márquez Brandon Angulo Aceves Juan Manuel
4-DEC-2025	Dockerfile	6	**Dockerización (Semana 15):** Creación de contenedor para despliegue portátil del servidor.	DevOps	Angulo Aceves Juan Manuel
5-DEC-2025		6	**Pruebas de Rendimiento (Semana 16):** Simulación de	QA	Angulo Aceves Juan Manuel

			carga y optimización de recursos.	
6-DEC-2025	Manual Tecnico.pdf Manual de Usuario.pdf	7	**Documentación (Semana 17):** Redacción de Manual Técnico y de Usuario.	Docs	Angulo Aceves Juan Manuel Ríos Márquez Brandon Díaz del Castillo Nava Enrique
17-DEC-2025	Presentacion	7	**Entrega Final (Semana 18):** Demo final y presentación del proyecto completo.	General	Mejía Olivares Bruno Eduardo Angulo Aceves Juan Manuel Ríos Márquez Brandon

6. Conclusión

El proyecto ha cumplido satisfactoriamente con los requerimientos académicos y técnicos planteados. La transición de una arquitectura simple a una solución profesional con seguridad SSL, base de datos e Inteligencia Artificial demuestra el dominio adquirido en programación concurrente, redes y arquitectura de software. El sistema resultante es una herramienta viable para su implementación real en entornos institucionales.

Fuentes

- Tkinter. com. (2023, August 08). Modern GUI Design With CustomTkinter! - Tkinter CustomTkinter 1. Youtube. Retrieved from https://www.youtube.com/watch?v=Y01r643ckfI&list=PLfZw_tZWahjxJl81b1S-vYQwHs_9ZT77f

- Official Documentation And Tutorial | CustomTkinter. (2025, July 20). Retrieved from https://customtkinter.tomschimansky.com

  Hammond, J. (2012, October 01). Python [hashlib] 01 Introduction. Youtube. Retrieved from https://www.youtube.com/watch?v=rJANKZ682ks&list=PL1H1sBF1VAKUcc5BrTB0rEvJLtIIQVKe1

- hashlib — Hashes seguros y resúmenes de mensajes — documentación de Python - 3.10.19. (2025, October 10). Retrieved from https://docs.python.org/es/3.10/library/hashlib.html

  Sematext. (2023, May 22). SSL/TLS Explained in 7 Minutes. Youtube. Retrieved from https://www.youtube.com/watch?v=67Kfsmy_frM

  What is SSL, TLS and HTTPS? | DigiCert. (2025, December 07). Retrieved from https://www-digicert-com.translate.goog/what-is-ssl-tls-and-https?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc

- JSON. (2025, October 11). Retrieved from https://www.json.org/json-es.html

  Web Dev Simplified. (2025, December 07). Learn JSON in 10 Minutes. Youtube. Retrieved from https://www.youtube.com/watch?v=iiADhChRriM&t=15s

- Grim, J. (2025, June 27). How To Run AI On Raspberry Pi 5 With Ollama. Youtube. Retrieved from https://www.youtube.com/watch?v=FUFWGhuHwh8&t=386s

- Emmet. (2025). How to run Ollama on the Raspberry Pi. Pi My Life Up. Retrieved from https://pimylifeup-com.translate.goog/raspberry-pi-ollama/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc

- In the Sky, J. P. (2025, November 02). Run Private AI on Your Raspberry Pi! (Ollama + Llama 3.2). Youtube. Retrieved from https://www.youtube.com/watch?v=bMPQ_jVVaIQ

- ProgrammingKnowledge. (2023, April 09). How to “Dockerize” Your Python Applications | How To Build And Run A Python App In Docker Container. Youtube. Retrieved from https://www.youtube.com/watch?v=KUECJHlV1LE

- "Part 1: Containerize an application". (2025, April 01). Retrieved from https://docs.docker.com/get-started/workshop/02_our_app