



# Universidad de las Fuerzas Armadas ESPE

## Departamento de Ciencias de la Computación

### Carrera de Ingeniería en Software

#### Aplicaciones Distribuidas

## Proyecto Integrador Parcial I

Docente: Geovanny Cudco

5 de noviembre de 2025

## 1 Tema

Sistema de Chat en tiempo real con salas seguras.

## 2 Descripción General:

Desarrollar un aplicativo de chat en tiempo real que permita la gestión de salas de conversación seguras y colaborativas. La aplicación debe contar con un backend para manejar la lógica del servidor, la autenticación (opcional), la persistencia de datos y la comunicación en tiempo real, y un frontend para la interfaz de usuario intuitiva y responsiva.

El enfoque principal es la creación y gestión de salas de chat por parte de un administrador, con acceso controlado para usuarios mediante PINs, y soporte para dos tipos de salas: texto (solo mensajes) y multimedia (mensajes más subida de archivos). El sistema debe implementar hilos (threads) para manejar la concurrencia en operaciones como la autenticación, la creación de salas y la transmisión de mensajes en tiempo real, asegurando escalabilidad y rendimiento sin bloqueos.

Los usuarios solo podrán unirse a una sala de chat a la vez desde un solo dispositivo (ordenador), lo que se verificará mediante un mecanismo de sesión única por IP o dispositivo. El proyecto debe priorizar la seguridad (autenticación, validación de PINs y nicknames), la usabilidad y la comunicación bidireccional en tiempo real.

# 3 Requisitos

## 3.1 Requisitos funcionales

1. **Autenticación de Administrador:** El administrador ingresa al sistema mediante credenciales (usuario y contraseña). Una vez autenticado, puede crear múltiples salas de chat.
2. **Creación de Salas:** Cada sala debe tener un ID único (generado automáticamente) y un PIN de acceso (de al menos 4 dígitos). Al crear una sala, el administrador selecciona el tipo:
  - Texto: Solo permite envío de mensajes de texto.
  - Multimedia: Permite envío de mensajes de texto y subida de archivos (imágenes, PDFs, etc., con límite de tamaño configurable, ej. 10MB).
3. **Acceso de Usuarios:** Los usuarios ingresan proporcionando el PIN de la sala y un nickname único dentro de la sala. No se requiere registro; el acceso es anónimo pero limitado a una sala por dispositivo.
4. **Funcionalidades en Sala:**
  - Envío y recepción de mensajes en tiempo real.
  - En salas multimedia, subida y visualización de archivos compartidos.
  - Lista de usuarios conectados en la sala (visibles por nickname).
  - Desconexión automática al cerrar el navegador o inactividad prolongada.
5. **Gestión de Concurrencia y Seguridad:** Utiliza hilos (threads) para manejar operaciones asíncronas, como:
  - Procesamiento de autenticaciones concurrentes.
  - Transmisión de mensajes a múltiples usuarios sin bloquear el servidor.
  - Manejo de subidas de archivos en paralelo.

## 3.2 Requisitos no funcionales

1. **Tiempo Real:** Actualizaciones instantáneas de mensajes (latencia  $< 1$  segundo).
2. **Escalabilidad:** Soporte para al menos 50 usuarios simultáneos por sala.
3. **Seguridad:** PINs encriptados, validación de entradas para prevenir inyecciones, y sesiones únicas por dispositivo.
4. **Interfaz:** Frontend responsivo (web-based), con diseño simple y accesible.
5. **Documentación:** Incluye un README con instrucciones de instalación, uso y diagrama de arquitectura.

## 4 Criterios de Evaluación

La presente actividad será evaluada sobre un total de 20 puntos, distribuidos según los siguientes criterios:

Tabla 1: Rúbrica de Evaluación del Proyecto de Comunicación en Tiempo Real

Criterio	Descripción y Puntuación
<b>1. Enunciado y Documentación (2 puntos)</b>	
Claridad del README (1 punto)	1: Describe claramente el proyecto, requisitos, instalación y uso. 0.5: Descripción incompleta o confusa. 0: Sin documentación adecuada.
Arquitectura (1 punto)	1: Incluye diagrama de arquitectura funcional y actualizado. 0.5: Diagrama parcial o desactualizado. 0: No incluye diagrama.
<b>2. Autenticación y Creación de Salas (4 puntos)</b>	
Login de administrador (2 puntos)	2: Funcional y seguro, con validación de credenciales. 1: Funciona parcialmente o con errores menores. 0: No funcional.
Gestión de salas (2 puntos)	2: Permite crear salas con ID único, PIN y tipos (texto/multimedia). 1: Creación funcional pero sin validaciones o sin tipos. 0: No crea salas correctamente.
<b>3. Acceso de Usuarios (3 puntos)</b>	
Ingreso mediante PIN/nickname (2 puntos)	2: Funciona correctamente con validación de unicidad por dispositivo. 1: Presenta fallos de validación o errores intermitentes. 0: No permite acceso adecuado.
Manejo de errores (1 punto)	1: Informa errores de acceso de forma clara. 0.5: Mensajes poco informativos. 0: Sin manejo de errores.
<b>4. Comunicación en Tiempo Real (4 puntos)</b>	

*Continúa en la siguiente página*

Cuadro 1 – *continuación de la página anterior*

Criterio	Descripción y Puntuación
Mensajería (2 puntos)	2: Envío y recepción de mensajes vía WebSockets totalmente funcional. 1: Comunicación parcial o con retrasos. 0: No hay comunicación en tiempo real.
Rendimiento y concurrencia (2 puntos)	2: Broadcast eficiente con hilos concurrentes estables. 1: Concurrencia limitada o errores bajo carga. 0: No usa hilos o el sistema se bloquea.
<b>5. Funcionalidades Multimedia (3 puntos)</b>	
Subida de archivos (2 puntos)	2: Permite subir archivos y visualizarlos correctamente en salas multimedia. 1: Subida funcional pero sin validaciones o visualización. 0: No implementa subida de archivos.
Validaciones (1 punto)	1: Aplica límites de tamaño y tipo de archivo. 0.5: Validaciones parciales. 0: Sin validaciones.
<b>6. Backend y Frontend (2 puntos)</b>	
Arquitectura (1 punto)	1: Separación clara entre lógica de negocio (backend) y presentación (frontend). 0.5: Capas parcialmente acopladas. 0: Sin separación de capas.
Diseño responsivo (1 punto)	1: Interfaz responsiva y funcional en diferentes dispositivos. 0.5: Problemas menores de visualización. 0: No responsivo.
<b>7. Pruebas y Escalabilidad (2 puntos)</b>	
Cobertura de pruebas (1 punto)	1: Cobertura superior al 70 % con casos variados. 0.5: Cobertura limitada o sin métricas claras. 0: Sin pruebas unitarias.
Rendimiento bajo carga (1 punto)	1: Soporta más de 50 usuarios simulados sin caídas. 0.5: Inestabilidad leve bajo carga. 0: Colapsa con múltiples usuarios.

## 5 Fecha de entrega

## 6 Entregables

- Código fuente completo (backend y frontend) en un repositorio Git.
- Diagramas de secuencia (ver abajo).
- Pruebas unitarias para al menos el 70 % de cobertura.
- Despliegue local (ej. Docker opcional).

Este proyecto fomenta el aprendizaje en programación concurrente, WebSockets para tiempo real y arquitectura cliente-servidor.

## 7 Tecnologías sugeridas

El estudiante puede elegir libremente las tecnologías que mejor se adapten a sus habilidades y preferencias, siempre priorizando herramientas open-source y compatibles con hilos para concurrencia. Aquí van algunas sugerencias:

- **Backend:** Node.js con Express (para API REST) y Socket.io (para WebSockets en tiempo real). Usa hilos nativos de Node (Worker Threads) o un framework como NestJS para manejar concurrencia. Alternativas: Python con Flask y asyncio/Threading, o Java con Spring Boot y Threads.
- **Frontend:** React.js o Vue.js para la interfaz dinámica, con bibliotecas como Socket.io-client para conexión en tiempo real. Para subida de archivos, usa Axios o Fetch API.
- **Base de Datos:** MongoDB (NoSQL para flexibilidad en mensajes) o PostgreSQL (SQL para relaciones estrictas como salas-usuarios).
- **Autenticación y Seguridad:** JWT para tokens de sesión; bcrypt para hashing de contraseñas. Para sesiones únicas por dispositivo, usa Redis para caching de IPs.
- **Despliegue:** Docker para contenedores; Heroku o Vercel para hosting gratuito.