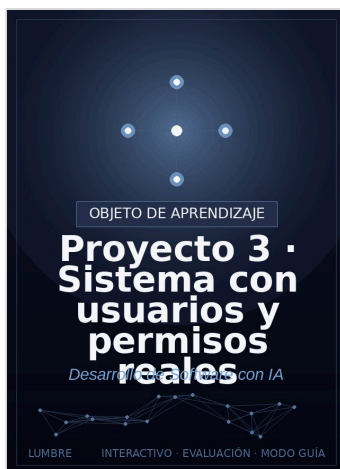


L U M B R E



Proyecto 3 · Sistema con usuarios y permisos reales

Materia: Desarrollo de Software con IA

07/10/2026

Antes de empezar

El proyecto donde tu app deja de ser juguete: cualquiera puede abrir tu agenda del proyecto 2, y ya no puede serlo —necesita cuenta, sesión y un rol que diga qué puede hacer. La matriz de permisos la escribes tú (el error clásico de la IA es implementar «admin» que no restringe nada), el registro con hash y la sesión los monta la IA bajo tus criterios de aceptación, y la sesión de auditoría con token caduco es la prueba de que entiendes por qué esta parte es sensible.

QUÉ APRENDERÁS

- Agregar registro, login, roles y matriz de permisos sin abrir puertas traseras.

La misión

Convierte tu agenda en un sistema para un pequeño equipo: **registro y login con contraseña hasheada, recuperación por correo simulada, y tres roles — propietario, editor, solo lectura— con una matriz de permisos que TÚ defines antes de pedir una línea**. Cada contacto pasa a tener dueño y visibilidad por equipo. El objetivo oculto del proyecto: que la auditoría final encuentre cero puertas traseras. Si «funciona» pero cualquiera puede editar al otro, falló.

La ruta, seis tiempos

ACTIVIDAD INTERACTIVA DISPONIBLE EN LUMBRE **1. Ruta del proyecto 3**



El prompt de autenticación (con los no-negociables)

La línea 6 y la prueba de propiedad cruzada: donde mueren los sistemas a medias

Capa nueva del proyecto 3: autenticacion y roles. Stack del documento maestro.\n\nNo negociables (cada uno es un criterio de aceptacion VERIFICABLE):\n1. contraseñas con hash de libreria estandar (bcrypt/argon segun stack), nunca en claro\n2. la contraseña nunca aparece en logs ni en mensajes de error, ni en la respuesta del API\n3. limite de intentos de login (5 por correo/hora) con espera creciente\n4. sesion con token de duracion limitada; el cliente guarda el token segun la practica\n correcta del stack y lo manda en cada peticion\n5. recuperacion: token de un solo uso con expiracion corta (simulable por correo local)\n6. TODOS los endpoints protegidos chequean: token valido + rol adecuado + propiedad del\n recurso. El frontend que oculta botones es UX, no defensa: la barrera vive aca.\n\nMatriz de permisos (definida por mi en ESPECIFICACION.md, pegarla completa aqui):\n[pega tu tabla operacion x rol literal]\n\nPruebas por endpoint: sin token (401), rol equivocado (403), propiedad cruzada\n(el editor del equipo A toca un contacto del equipo B -> 403), token caducado.\n\nImplementa, corre todo, muestra diff y salida. No toques la logica de contactos de\nla capa 2 sin mostrarme el diff de lo que cambia.

Qué entrega este proyecto

Si tu auditoría de las tres puertas encuentra cero errores en el primer intento, sospecha: no es que esté perfecto, es que la prueba fue suave. Los mejores HALLAZGOS.md de este proyecto nacen de la puerta que sí se abrió.

Entregable	Barra alta de este proyecto
ESPECIFICACION.md	la matriz operación × rol sin casillas en blanco y el caso del propietario que se va
HALLAZGOS.md	el endpoint que dejaba pasar al usuario equivocado, el login que logueaba la contraseña
PRUEBAS.md	las tres puertas auditadas + propiedad cruzada por endpoint crítico
App publicada	dos cuentas reales de prueba editando cada una lo suyo, nada más

Prueba

2. Tu sistema: el menú muestra el botón «eliminar» solo al propietario, y los demás no lo ven. ¿Es suficiente para cumplir la matriz?

- ☐ Sí: si no lo ven, no lo usan.
- ☐ No: hay que verificar que el endpoint DELETE rechace con 403 al editor y al de lectura —la barrera vive en el servidor, no en el menú.
- ☐ Sí, mientras el token tenga duración limitada.
- ☐ No: hay que ocultar también el botón de editar.

3. Almacena la contraseña con hash (no en claro) y ya: con eso la autenticación del proyecto está bien protegida.

- ☐ Verdadero
- ☐ Falso

4. Empieza por el punto que la IA no puede escribir por ti: la matriz operación × rol completa, en ESPECIFICACION.md. Luego el prompt de capa con los seis no-negociables. Cierra con la auditoría de las tres puertas Y la propiedad cruzada documentadas —si la primera prueba te deja limpio, prueba más sucio.

Un sistema de permisos es una decisión de negocio escrita dos veces: una en la especificación, otra en cada endpoint.

Referencia curricular

Materia:	Desarrollo de Software con IA
-----------------	-------------------------------

Versi\u00f3n interactiva completa (con actividades, v\u00eddeos y autoevaluaci\u00f3n):



<https://empresa.lumbre.lat/object/fsi-pj-oa03>

Material educativo generado desde Lumbre · lumbre.lat