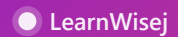


Ejercicio de programación con IA

Construye el ejemplo EnterpriseOps de este módulo con ChatGPT o Claude a partir de un prompt listo para usar que dirige al modelo a la especificación del módulo, y después revisa, ejecuta y amplía lo que te devuelva — incluso comparándolo con nuestra propia solución de referencia en GitHub.



Instrúyelo, léelo y luego rómpelo.

Este ejercicio tiene tres partes, y solo la primera trata del prompt. Le das al asistente la especificación del módulo y recibes una solución EnterpriseOps; lees lo que escribió y decides si de verdad hace lo que pedía el módulo; y después lo llevas más allá de lo que pedía el módulo. En la segunda y la tercera parte es donde se aprende.

Qué estás pidiendo

Amplía EnterpriseOps con selección de tenant, servicios conscientes del tenant, IDs de correlación y concurrencia optimista en las órdenes de trabajo. Crea un diálogo de conflicto que explique las ediciones obsoletas y ofrezca las opciones Recargar, Comparar y Cancelar.

El módulo pide estos entregables, y el prompt de abajo le pide al asistente exactamente esos:

- ✓ SessionContext consciente del tenant
- ✓ CommandContext con ID de correlación
- ✓ Informe de auditoría del estado estático
- ✓ Implementación de concurrencia optimista
- ✓ Diálogo de resolución de conflictos

Por qué tiene esta forma

Las aplicaciones avanzadas atienden a muchos usuarios, tenants, pestañas del navegador y operaciones en segundo plano. Los errores de estado se convierten en fallos de seguridad, fugas de datos o flujos de trabajo incoherentes.

Estos son los términos que el asistente tiene que dominar. Si lo que te devuelve los usa a la ligera, no ha entendido el módulo:

Término	Qué significa aquí
contexto de sesión	El estado de servidor por usuario que Wisej.NET mantiene vivo para cada sesión del navegador. Es donde deben estar la identidad del usuario y el estado de trabajo, nunca en campos estáticos.
contexto de tenant	La identidad del tenant actual, resuelta una vez por sesión y aplicada en los servicios y el acceso a datos, para que un cliente nunca pueda ver los datos de otro cliente.
ID de correlación	Un identificador único asociado a una solicitud o a un trabajo y propagado por los registros, para poder seguir una acción del usuario a través de servicios y servidores.
concurrency optimista	Detectar que dos usuarios han editado el mismo registro comparando una versión o una marca de tiempo, y gestionar el conflicto en lugar de sobrescribir en silencio.
estado por pestaña	Estado que pertenece a una pestaña o ventana del navegador y no a toda la sesión, para que dos pestañas abiertas no estropeen el trabajo de la otra.
auditoría del estado estático	Una revisión de cada campo estático y singleton para confirmar que no contiene datos por usuario ni por tenant, porque los estáticos los comparten todas las sesiones del servidor.
revisión de condiciones de carrera	Comprobar las rutas de código en las que tareas en segundo plano, temporizadores o varias solicitudes tocan el mismo estado de sesión, para que las actualizaciones se sincronicen correctamente.

Constrúyelo — elige tu asistente

Los dos botones abren una pestaña nueva con el **mismo prompt listo para usar** ya escrito. El prompt dirige al modelo a la especificación de este módulo — un único archivo Markdown con la lección, el enunciado del laboratorio y los entregables — para que trabaje con el material real en lugar de adivinar de qué trata el Módulo 3. Puedes abrir tú mismo [module-03.md](#) para ver exactamente lo que leerá.



Usar ChatGPT

Abre [chatgpt.com](#) con el prompt de este módulo ya cargado. Funciona con GPT-5 o Codex; cualquier plan que pueda navegar por la web descargará la especificación.



Usar Claude

Abre [claude.ai](#) con el mismo prompt ya cargado. Si usas Claude Code, puedes pegarlo en una terminal abierta en una carpeta vacía.



Envíalo tal cual y obtendrás un proyecto, no un fragmento

El prompt termina pidiendo toda la solución como un único `.zip` con los pasos para compilarla y ejecutarla. Descárgalo, abre la solución en Visual Studio, pulsa F5 y EnterpriseOps se abrirá en `localhost`. Necesitas el SDK de .NET y el paquete NuGet `Wisej-4`; todo lo demás, el prompt le pide al asistente que lo escriba él mismo.

Estoy siguiendo el Módulo 3 – «Advanced Session, State, Tenant, and Concurrency Design» – del curso de LearnWisej «Enterprise Wisej.NET: Architecture to Cloud», y quiero construir contigo el ejemplo de este módulo. Respóndeme en español.

Empieza leyendo la especificación del módulo. Es un único archivo Markdown (en inglés) que contiene la lección completa, el enunciado del laboratorio y los entregables:

<https://learnwisej.com/assets/courses/azure-docker/specs/module-03.md>

Después construye un proyecto Wisej.NET 4 completo que pueda abrir en Visual Studio:

1. Proyecto «EnterpriseOps», C#, net8.0 o posterior, paquete NuGet Wisej-4 4.1.x, host Kestrel que llame a `app.UseWisej()`.
2. Construye lo que pide el laboratorio de la especificación: Amplía EnterpriseOps con selección de tenant, servicios conscientes del tenant, IDs de correlación y concurrencia optimista en las órdenes de trabajo. Crea un diálogo de conflicto que explique las ediciones obsoletas y ofrezca las opciones Recargar, Comparar y Cancelar.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - SessionContext consciente del tenant
 - CommandContext con ID de correlación
 - Informe de auditoría del estado estático
 - Implementación de concurrencia optimista
 - Diálogo de resolución de conflictos
4. Sigue la especificación siempre que nombre una clase, propiedad o evento de Wisej.NET – usa exactamente ese. Donde deje la elección abierta, dime qué API elegiste y por qué.
5. Mantén la autoridad en el servidor: el estado y la validación viven en .NET, y el navegador muestra e informa. Nunca traslades una decisión a JavaScript porque allí resulte más fácil.
6. Haz que el resultado se pueda demostrar: debo poder ejecutarlo y ver el comportamiento del que trata este módulo, incluido lo que ocurre cuando algo sale mal.

Antes de escribir una sola línea de código, explica con tus propias palabras qué vas a construir y dónde vivirá cada pieza, para que pueda corregir el diseño mientras todavía es barato.

Cuando el código esté terminado, empaqueta toda la solución en un único archivo .zip descargable y dame los pasos exactos para compilarla y ejecutarla.

Por último, responde a estas preguntas de la especificación usando tu propia implementación como prueba:

- ¿Se puede falsificar el ID de tenant desde un control de la UI?
- ¿Qué ocurre cuando la misma orden de trabajo se edita en dos sesiones?
- ¿Qué servicios se pueden compartir de forma segura?

Pregúntame por cualquier cosa ambigua antes de empezar.

Después revisa lo que te devuelve

Que el código compile no significa que sea correcto. Las preguntas interesantes son de diseño: ¿es este el control de Wisej.NET que la tarea realmente necesita?, ¿sigue el ciclo de vida que te ofrece el framework?,

¿el estado de referencia está en el servidor o se quedó en el navegador porque era más cómodo? Lee el resultado como revisor, no como cliente.

Este módulo te da la lista de comprobación. Responde a cada punto basándote en el código que acabas de recibir:

- ✓ ¿Se puede falsificar el ID de tenant desde un control de la UI?
- ✓ ¿Qué ocurre cuando la misma orden de trabajo se edita en dos sesiones?
- ✓ ¿Qué servicios se pueden compartir de forma segura?

Todo lo que hayas tenido que corregir merece quedar por escrito. Una nota breve con las convenciones y decisiones que acordaste, pegada en el primer mensaje de tu próxima sesión, vale más que cualquier prompt ingenioso.



Este ya lo construimos — compara tu versión con la nuestra

La implementación de referencia del Módulo 3 está en el repositorio público de ejemplos. Construye primero la tuya y después abre la nuestra: lo interesante son las diferencias.



[Abrir el ejemplo del Módulo 3 en GitHub](#)

O clona el conjunto completo con `git clone https://github.com/iceteagroup/LearnWisej-Samples` y abre `Enterprise Wisej.NET Course/Module 3`.



Nuestro ejemplo es un punto de partida, no un producto terminado

Se construyó igual que estás construyendo el tuyo — con un prompt y luego corregido — y se detiene a propósito donde se detiene el módulo. Espera soluciones provisionales donde iría código de producción, un manejo de errores mínimo y ninguna prueba. Tómallo como una invitación: usa lo que aprendiste en este módulo, junto con el asistente, para llevarlo más lejos. Mejorar código que no escribiste tú es precisamente la habilidad que enseña este curso.

Tres prompts para ir más allá

Dirige tu asistente al proyecto que acabas de construir — o al nuestro y prueba estos. Cada uno es un ejercicio de revisión y ampliación, no una reescritura.

- ✓ **Haz que aguante un mal día.** *"Revisa este proyecto EnterpriseOps en busca de los modos de fallo que todavía no gestiona — una entrada incorrecta o ausente, una dependencia lenta o no disponible, trabajo que llega después de que el usuario haya salido de la pantalla y dos acciones que compiten entre sí. Para cada uno, dime primero si es un riesgo real en Wisej.NET 4 y por qué, y solo después escribe la corrección. Si decides que algo no necesita gestión, dilo y explica el razonamiento en lugar de añadir código defensivo."*
- ✓ **Añade el segundo caso.** *"Amplía este proyecto EnterpriseOps con un segundo caso que siga el mismo patrón que el primero — otra entidad, pantalla o control con la misma forma, conectado de la misma manera. Extrae a una clase base o a un servicio lo que de verdad sea común, pero solo donde se justifique; dos cosas parecidas no siempre piden una abstracción. Muéstrame el diff y justifica cada cosa que hayas extraído."*
- ✓ **Demuestra que de verdad funciona.** *"Recorre este proyecto entregable por entregable frente a la especificación del módulo en <https://learnwisej.com/assets/courses/azure-docker/specs/module-03.md> y muéstrame la prueba de que cada uno está realmente terminado — la ruta del código, dónde hacer clic y qué debería ver. Si un entregable solo se cumple en parte, dilo claramente en lugar de justificarlo. Después añade lo que falte."*



Llévalo al laboratorio

El laboratorio práctico que cierra este módulo pide los mismos entregables que acabas de hacer que generara el asistente. Usa lo que tienes aquí como forma de tu respuesta y como prueba que la respalda — no como la respuesta en sí. El laboratorio se califica por tu razonamiento, y esa parte tiene que ser tuya.