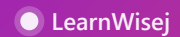


Ejercicio de programación con IA

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.



El asistente pone las manos rápidas. El arquitecto sigues siendo tú.

Un asistente puede generar el ejemplo de este módulo en minutos. Lo que no puede hacer es decidir si el resultado es el diseño adecuado para tu aplicación — esa decisión sigue siendo tuya. Tú pones el criterio: dale unas buenas instrucciones y luego contrasta lo que te devuelva con lo que te acaba de enseñar el Módulo 7.

Qué estás pidiendo

Crea un Escalation Wizard que recopile el motivo, los adjuntos, el aprobador, la fecha límite y las opciones de notificación. Implementa la orquestación en un servicio, no en las páginas del asistente, y añade compensación para un fallo de notificación simulado.

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

- ✓ Flujo de pantallas del asistente
- ✓ Servicio de flujo de trabajo
- ✓ Objetos tipados de comando y resultado
- ✓ Matriz de rutas de fallo
- ✓ Ejemplo de compensación

Por qué tiene esta forma

Los flujos de trabajo empresariales rara vez caben en un único botón Save. Las pantallas avanzadas coordinan aprobaciones, asignaciones, validaciones, cargas de archivos y comprobaciones externas sin que el usuario pierda la confianza.

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
orquestración del flujo de trabajo	Un servicio que es dueño de la secuencia, las reglas y el estado de un proceso de negocio de varios pasos, para que el flujo no quede enterrado entre varios formularios.
estado del asistente	Las respuestas acumuladas y el estado de validación de un asistente de varios pasos, guardados en un único objeto tipado en lugar de dispersos entre los controles.
resultado tipado	Un valor de retorno que modela explícitamente el éxito, el fallo de validación y los errores, de modo que quien llama debe gestionar cada ruta en lugar de depender de excepciones.
comando de compensación	Una operación que deshace los efectos de un paso anterior cuando falla un paso posterior, manteniendo el sistema coherente sin una única transacción gigantesca.
flujo de trabajo reanudable	Un proceso que persiste su progreso para que el usuario pueda cerrar el navegador y continuar más tarde, o para recuperarse de un fallo desde el último paso correcto.
matriz de rutas de fallo	Una tabla que enumera cada paso, lo que puede salir mal en él y cómo responde el flujo, para que la gestión de fallos se diseñe en lugar de improvisarse.

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 7. Puedes abrir tú mismo [module-07.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 7 – «Advanced Workflow UX: Wizards, Modal Orchestration, and Compensation» – 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-07.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: Crea un Escalation Wizard que recopile el motivo, los adjuntos, el aprobador, la fecha límite y las opciones de notificación. Implementa la orquestación en un servicio, no en las páginas del asistente, y añade compensación para un fallo de notificación simulado.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - Flujo de pantallas del asistente
 - Servicio de flujo de trabajo
 - Objetos tipados de comando y resultado
 - Matriz de rutas de fallo
 - Ejemplo de compensación
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 probar el flujo de trabajo sin el asistente?
- ¿Qué estado sobrevive a una actualización del navegador?
- ¿Qué ocurre si la notificación tiene éxito pero falla el registro de auditoría?

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 probar el flujo de trabajo sin el asistente?
- ✓ ¿Qué estado sobrevive a una actualización del navegador?
- ✓ ¿Qué ocurre si la notificación tiene éxito pero falla el registro de auditoría?

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 7 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 7 en GitHub](#)

O clona el conjunto completo con `git clone https://github.com/iceteagroup/LearnWisej-Samples` y abre `Enterprise Wisej.NET Course/Module 7`.



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.

- ✓ **Cambia una pieza sin mover las juntas.** *"Elige la parte de este proyecto EnterpriseOps que sea más claramente un sustituto de algo real — una fuente de datos falsa, un almacén en memoria, un servicio simplificado — y reemplázala por una implementación con forma de producción. El contrato a ambos lados no debe cambiar: los mismos tipos, las mismas llamadas, el mismo comportamiento ante fallos. Dime qué te obligó a gestionar la implementación real que el sustituto no gestionaba."*
- ✓ **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."*



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.