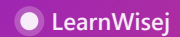


Ejercicio de programación con IA

Construye el ejemplo FieldOps 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.



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 FieldOps; 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

Empaqueta FieldOps para Windows: un shell WebView2 con ventana nativa, título personalizado e icono en la bandeja del sistema, un puente que permita a la aplicación abrir un selector de carpetas nativo para exportar informes de órdenes de trabajo y mostrar una notificación nativa cuando llegue una orden nueva, una configuración que apunte el shell al servidor remoto con un servidor local de respaldo, y un instalador con comprobación de actualizaciones al iniciar.

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

- ✓ Shell de escritorio WebView2 con ventana e icono en la bandeja
- ✓ Puente nativo para el selector de carpetas y las notificaciones
- ✓ Configuración de servidor remoto y local
- ✓ Paquete de instalación
- ✓ Comprobación de actualizaciones al iniciar

Por qué tiene esta forma

Los técnicos de FieldOps pasan un turno entero dentro de la aplicación: un portátil en la furgoneta, una estación de acoplamiento en el depósito. La versión para navegador de los dos primeros módulos hace el trabajo, pero no tiene icono en el menú Inicio, ni presencia en el área de notificaciones cuando se asigna un trabajo, ni forma de dejar un informe exportado en una carpeta elegida por el técnico, ni nada que el departamento de TI pueda distribuir por su canal de software habitual. Este módulo envuelve esa misma aplicación, sin cambios, en un shell de Windows que aporta exactamente esas cuatro cosas.

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
control WebView2	El motor Chromium que Microsoft distribuye como Evergreen Runtime redistribuible, integrado en una ventana WinForms o WPF para que tu página se renderice exactamente como en Edge.
shell de escritorio	Un pequeño ejecutable nativo que controla la ventana, el icono de la bandeja y las llamadas al sistema operativo, y apunta un control WebView2 a tu aplicación Wisej.NET.
marco de la ventana	El título, el icono, el tamaño mínimo, la posición recordada y el comportamiento de instancia única que hacen que la aplicación parezca instalada y no incrustada.
punto de mensajes	El canal bidireccional entre la página y el shell: PostWebMessageAsJson hacia la página, WebMessageReceived hacia el shell, o un objeto host publicado con AddHostObjectToScript.
interoperabilidad con JavaScript	Cómo llega a ese canal el código C# del servidor: Application.Eval y Application.Call ejecutan script en el contexto del navegador de la sesión, y la respuesta vuelve como un evento de cliente.
topología del servidor	Si el shell navega a un servidor remoto de la empresa o inicia un proceso Wisej.NET autoalojado en loopback; se decide al iniciar y se registra, nunca se adivina.
canal de actualización	Un endpoint de versión que el shell consulta al iniciar, más el instalador que descarga cuando la respuesta es más reciente que la versión que está ejecutando.

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 FieldOps 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 - «Empaquetado de escritorio con un shell WebView» - del curso de LearnWisej «Empaquetado híbrido y nativo», 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/hybrid-native-packaging/specs/module-03.md>

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

1. Proyecto «FieldOps», 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: Empaqueta FieldOps para Windows: un shell WebView2 con ventana nativa, título personalizado e icono en la bandeja del sistema, un puente que permita a la aplicación abrir un selector de carpetas nativo para exportar informes de órdenes de trabajo y mostrar una notificación nativa cuando llegue una orden nueva, una configuración que apunte el shell al servidor remoto con un servidor local de respaldo, y un instalador con comprobación de actualizaciones al iniciar.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - Shell de escritorio WebView2 con ventana e icono en la bandeja
 - Puente nativo para el selector de carpetas y las notificaciones
 - Configuración de servidor remoto y local
 - Paquete de instalación
 - Comprobación de actualizaciones al iniciar
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:

- Un técnico inicia dos exportaciones con menos de un segundo de diferencia y el shell responde a los dos cuadros de diálogo de carpetas. ¿Qué elemento del diseño de tu puente impide que la segunda ruta se escriba en el primer informe, y qué fallaría sin él?
- Tu botón Export abre ahora un cuadro de diálogo de carpetas nativo. Describe con precisión qué hace ese mismo botón cuando la página se abre en Edge en lugar de en el shell, y en qué parte de tu código se decide esa diferencia.
- El endpoint de versión no está disponible porque la furgoneta no tiene señal. Argumenta qué debería hacer el shell en ese inicio y explica qué pierde un técnico con cada una de las alternativas que has descartado.

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:

- ✓ Un técnico inicia dos exportaciones con menos de un segundo de diferencia y el shell responde a los dos cuadros de diálogo de carpetas. ¿Qué elemento del diseño de tu puente impide que la segunda ruta se escriba en el primer informe, y qué fallaría sin él?
- ✓ Tu botón Export abre ahora un cuadro de diálogo de carpetas nativo. Describe con precisión qué hace ese mismo botón cuando la página se abre en Edge en lugar de en el shell, y en qué parte de tu código se decide esa diferencia.
- ✓ El endpoint de versión no está disponible porque la furgoneta no tiene señal. Argumenta qué debería hacer el shell en ese inicio y explica qué pierde un técnico con cada una de las alternativas que has descartado.

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.



Todavía no hay un ejemplo de referencia para este curso — el tuyo es el primero

El [repositorio de ejemplos de LearnWisej](#) tiene soluciones completas para la mayoría de los cursos, pero todavía no para este. Eso significa que no hay nada con qué compararte ni ningún atajo: la especificación y tu propia revisión son toda tu red de seguridad. Guarda lo que construyas — y si sale bien, nos encantaría verlo.

Tres prompts para ir más allá

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

- ✓ **Añade el segundo caso.** *"Amplía este proyecto FieldOps 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/hybrid-native-packaging/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."

- ✓ **Revísalo como un pull request.** "Explícame este código de FieldOps como si estuvieras revisando el pull request de un compañero y no lo hubieras escrito tú. ¿Dónde vive el estado?, ¿qué ocurre de principio a fin con una acción del usuario?, ¿qué devolverías para que lo cambien? y ¿qué partes no te sentirías cómodo poniendo delante de un cliente? Sé concreto y cita las líneas a las que te refieres — no me resumas el diseño."



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.