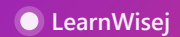


Ejercicio de programación con IA

Construye el ejemplo AccessOps 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.



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 2.

Qué estás pidiendo

Implementa la autenticación de AccessOps: una página de inicio de sesión con bloqueo de cuenta tras cinco fallos, verificación de contraseñas con un hash moderno con sal, un SessionContext creado solo tras la verificación y guardado en la sesión, una expiración por inactividad que devuelve al usuario al inicio de sesión y borra el estado, un Logout explícito que termina la sesión y un registro de auditoría por cada intento de inicio de sesión.

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

- ✓ Página de inicio de sesión con bloqueo de cuenta y mensajes de error genéricos
- ✓ Servicio de hash y verificación de contraseñas
- ✓ SessionContext con identidad verificada
- ✓ Expiración por inactividad y cierre de sesión explícito
- ✓ Registros de auditoría de los inicios de sesión

Por qué tiene esta forma

AccessOps contiene fichas de empleados, documentos confidenciales y solicitudes de aprobación, así que la primera pantalla a la que llega cualquiera es también el primer control que importa. Todo lo que el resto del curso hace cumplir —permisos, reglas a nivel de registro, registros de auditoría— descansa sobre una única suposición: que el servidor sabe quién está al otro lado de esta sesión y puede demostrarlo. El Módulo 1 dibujó los límites de confianza; este módulo coloca una identidad verificada dentro del límite de sesión e impide que se adivine, se tome prestada o se prolongue más allá de su vida.

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
hash con sal y lento	Una derivación unidireccional de la contraseña con una sal aleatoria por usuario y un número de iteraciones deliberadamente costoso, para que una tabla robada no pueda revertirse y cada intento cueste trabajo real.
comparación en tiempo constante	Comparar dos hashes con un número fijo de operaciones mediante <code>CryptographicOperations.FixedTimeEquals</code> , para que el tiempo que tarda el servidor en responder nunca revele cuántos bytes de un intento eran correctos.
SessionContext	El objeto que se crea solo después de verificar una contraseña, con el id del empleado, el departamento, los roles y los datos del inicio de sesión, y que se guarda en <code>Application.Session</code> como el único lugar al que cualquier código pregunta quién es el usuario.
expiración de la sesión	El periodo de inactividad tras el cual Wisej.NET libera la sesión del lado del servidor y todo lo que contiene, configurado para la aplicación en <code>Default.json</code> en lugar de estimarse pantalla por pantalla.
fijación de sesión	Un ataque en el que el identificador que llevaba la página de inicio de sesión anónima lleva después también la sesión autenticada, de modo que un identificador plantado por el atacante se convierte en uno con sesión iniciada.
limitación de intentos de inicio de sesión	Contar los fallos por cuenta y por dirección del cliente y rechazar nuevos intentos durante un periodo de bloqueo, que es lo que convierte una máquina ilimitada de adivinar en unos pocos intentos por hora.
registro de auditoría del inicio de sesión	Una fila por intento con el nombre de usuario probado, la hora, la dirección del cliente y el resultado real: el lugar donde se escribe el motivo concreto del fallo, ya que la pantalla solo muestra uno genérico.

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 2. Puedes abrir tú mismo [module-02.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 AccessOps 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 2 – «Autenticación e identidad de sesión» – del curso de LearnWisej «Patrones de seguridad y autenticación», 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/security-and-auth-patterns/specs/module-02.md>

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

1. Proyecto «AccessOps», 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: Implementa la autenticación de AccessOps: una página de inicio de sesión con bloqueo de cuenta tras cinco fallos, verificación de contraseñas con un hash moderno con sal, un `SessionContext` creado solo tras la verificación y guardado en la sesión, una expiración por inactividad que devuelve al usuario al inicio de sesión y borra el estado, un Logout explícito que termina la sesión y un registro de auditoría por cada intento de inicio de sesión.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - Página de inicio de sesión con bloqueo de cuenta y mensajes de error genéricos
 - Servicio de hash y verificación de contraseñas
 - `SessionContext` con identidad verificada
 - Expiración por inactividad y cierre de sesión explícito
 - Registros de auditoría de los inicios de sesión
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:

- Tu página de inicio de sesión responde a una contraseña incorrecta en 480 ms y a un nombre de usuario desconocido en 6 ms. Explica qué averigua un atacante solo con esa diferencia y describe los dos cambios que eliminan la señal sin cambiar el mensaje.
- Un compañero propone guardar al empleado con la sesión iniciada en una propiedad estática porque «es más sencillo que ir a buscarlo a la sesión cada vez». Describe exactamente lo que ve el segundo usuario concurrente y en qué momento interfieren por primera vez las dos sesiones.
- Tu botón Logout borra el `SessionContext` pero no termina la sesión. Nombra dos cosas que siguen siendo recuperables después y di cuál consideras más grave en un portal que muestra documentos confidenciales.

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:

- ✓ Tu página de inicio de sesión responde a una contraseña incorrecta en 480 ms y a un nombre de usuario desconocido en 6 ms. Explica qué averigua un atacante solo con esa diferencia y describe los dos cambios que eliminan la señal sin cambiar el mensaje.
- ✓ Un compañero propone guardar al empleado con la sesión iniciada en una propiedad estática porque «es más sencillo que ir a buscarlo a la sesión cada vez». Describe exactamente lo que ve el segundo usuario concurrente y en qué momento interfieren por primera vez las dos sesiones.
- ✓ Tu botón Logout borra el SessionContext pero no termina la sesión. Nombra dos cosas que siguen siendo recuperables después y di cuál consideras más grave en un portal que muestra documentos confidenciales.

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.

- ✓ **Revísalo como un pull request.** *"Explícame este código de AccessOps 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 resumes el diseño."*
- ✓ **Cambia una pieza sin mover las juntas.** *"Elige la parte de este proyecto AccessOps 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 AccessOps 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."*



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.