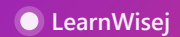


Ejercicio de programación con IA

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 devuelve.



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 AccessOps; 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

Añade autorización a AccessOps: un PermissionService que convierte roles y claims en permisos al iniciar sesión, una protección Authorize a la que llaman todos los métodos de DocumentService y ApprovalService, una regla a nivel de registro por la que los empleados solo ven su departamento y los responsables solo a sus subordinados, una interfaz que oculta las acciones de administración y desactiva los botones de aprobar que el usuario no puede usar, y una prueba que demuestra que el servicio rechaza una solicitud falsificada de alguien que no es responsable.

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

- ✓ PermissionService que resuelve los permisos en el SessionContext
- ✓ Protección de autorización en cada método de servicio
- ✓ Reglas a nivel de registro por departamento y por responsable
- ✓ Adaptación de la interfaz según los permisos
- ✓ Prueba negativa con una solicitud falsificada

Por qué tiene esta forma

AccessOps ya sabe quién ha iniciado sesión: el Módulo 2 dejó una identidad verificada en el SessionContext. Saber quién es alguien no dice nada sobre lo que puede hacer con un documento confidencial o con una solicitud de aprobación de seis cifras. En un portal interno, el fallo rara vez es un inicio de sesión roto: es un administrativo de nóminas que puede leer los contratos del departamento jurídico porque la consulta nunca se acotó, o un jefe de equipo que aprueba su propia solicitud de gastos porque la única comprobación era un botón desactivado.

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
rol	Una función laboral amplia y con nombre, como Employee, Manager o Auditor. Los roles son cómodos de administrar, pero demasiado toscos para comprobarlos directamente, porque una pantalla necesita saber de una acción, no de todo un puesto.
claim	Un dato nombre/valor sobre la identidad verificada: departamento 14, centro de coste EU-North, responsable del equipo 7. Los claims llevan los datos que necesitan las reglas a nivel de registro, y proceden del almacén de identidades, nunca del navegador.
permiso	Una acción con nombre que la aplicación puede realizar, como ApproveRequest o ViewDepartmentDocuments. El código comprueba permisos; los administradores asignan roles; la correspondencia entre ambos es configuración.
resolución de permisos	El paso único, al iniciar sesión, que expande roles y claims en un conjunto de permisos plano e inmutable guardado en el SessionContext, para que ninguna solicitud vuelva a leer la tabla de roles y dos comprobaciones no puedan discrepar.
protección de autorización	Una llamada como Require(Permission.ApproveRequest) al principio de un método de servicio que lanza una excepción antes de que se haga ningún trabajo. Es el único lugar donde realmente se hace cumplir la regla.
regla a nivel de registro	Una restricción sobre qué filas puede tocar una acción permitida —tu propio departamento, tus propios subordinados— aplicada dentro de la consulta o del comando y no después de haber cargado ya los datos.
adaptación de la interfaz	Ocultar, desactivar o explicar una acción según el mismo conjunto de permisos que aplica el servidor. Es una cortesía hacia el usuario y nunca un control.

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 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 3 – «Autorización: roles, claims y permisos» – 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-03.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: Añade autorización a AccessOps: un `PermissionService` que convierte roles y claims en permisos al iniciar sesión, una protección `Authorize` a la que llaman todos los métodos de `DocumentService` y `ApprovalService`, una regla a nivel de registro por la que los empleados solo ven su departamento y los responsables solo a sus subordinados, una interfaz que oculta las acciones de administración y desactiva los botones de aprobar que el usuario no puede usar, y una prueba que demuestra que el servicio rechaza una solicitud falsificada de alguien que no es responsable.
3. Genera estos entregables e indícame dónde está cada uno dentro de la solución:
 - `PermissionService` que resuelve los permisos en el `SessionContext`
 - Protección de autorización en cada método de servicio
 - Reglas a nivel de registro por departamento y por responsable
 - Adaptación de la interfaz según los permisos
 - Prueba negativa con una solicitud falsificada
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:

- A un auditor se le conceden derechos de aprobación mientras su sesión de AccessOps sigue abierta. Explica exactamente cuándo entra en vigor el nuevo permiso con tu diseño, por qué elegiste ese momento y qué tendrías que cambiar para que fuera inmediato.
- Tu consulta de documentos filtra por departamento en el repositorio. Describe qué gana un atacante si ese filtro se traslada al `DataSource` de la cuadrícula, aunque el resultado visible en pantalla sea idéntico.
- Tanto `ApprovalService` como su repositorio podrían comprobar que el aprobador es responsable del solicitante. Defiende uno de ellos como el lugar correcto y di qué se rompe si la comprobación está en el otro, o en ambos.

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:

- ✓ A un auditor se le conceden derechos de aprobación mientras su sesión de AccessOps sigue abierta. Explica exactamente cuándo entra en vigor el nuevo permiso con tu diseño, por qué elegiste ese momento y qué tendrías que cambiar para que fuera inmediato.
- ✓ Tu consulta de documentos filtra por departamento en el repositorio. Describe qué gana un atacante si ese filtro se traslada al DataSource de la cuadrícula, aunque el resultado visible en pantalla sea idéntico.
- ✓ Tanto ApprovalService como su repositorio podrían comprobar que el aprobador es responsable del solicitante. Defiende uno de ellos como el lugar correcto y di qué se rompe si la comprobación está en el otro, o en ambos.

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.

- ✓ **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."

- ✓ **Añade el segundo caso.** *"Amplía este proyecto AccessOps 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.