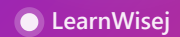


Arquitectura de seguridad: identidad, SSO, autorización, auditoría y despliegue seguro

Guía de la lección · Guía del laboratorio / examen



Guía de la lección

Integración de identidad, modelo de permisos, pistas de auditoría, HTML seguro, configuración segura. Lo que esto significa para un equipo avanzado de Wisej.NET — y cómo leer este módulo.



Enfoque de producción

Integración de identidades, modelo de permisos, pistas de auditoría, HTML seguro, configuración segura.

1. El problema en producción

Una arquitectura del lado del servidor reduce muchas superficies de ataque del navegador, pero los sistemas avanzados siguen necesitando integración de identidades, autorización a nivel de servicio, pistas de auditoría, renderizado seguro y una configuración operativa segura.

2. Conceptos clave

Concepto	Por qué importa
límite OIDC/SSO	El punto en el que un proveedor de identidades externo entrega a la aplicación una identidad verificada. Todo lo que hay después de ese límite es responsabilidad de la aplicación.
mapeo de claims	Traducir los claims específicos del proveedor a los roles y permisos propios de la aplicación, para que el código de autorización nunca dependa del formato de un proveedor de identidades concreto.
servicio de permisos	Un único servicio que responde si el usuario actual puede realizar una acción, para que la regla viva en un solo lugar y no en cada pantalla.
autorización a nivel de servicio	Aplicar los permisos dentro de los servicios de aplicación, no solo en la UI, para que un botón oculto nunca sea lo único que protege una operación.
registro de auditoría	Un registro a prueba de manipulaciones de quién hizo qué y cuándo en las acciones importantes, para poder investigar los incidentes de seguridad y demostrar el cumplimiento normativo.
revisión de AllowHtml	Comprobar cada lugar en el que un control permite HTML sin procesar para asegurarse de que el contenido aportado por el usuario se sanea, porque el HTML sin revisar es un riesgo de cross-site scripting.
gestión de secretos	Guardar las cadenas de conexión, las claves y los tokens en una configuración segura o en un almacén de secretos, nunca en el control de código fuente ni en ajustes visibles desde el cliente.
cabeceras seguras	Cabeceras de respuesta HTTP como la política de seguridad de contenido y la seguridad de transporte estricta, que reducen la superficie de ataque de la aplicación en el navegador.

3. Desarrollo de la lección

Autenticación no es autorización

Un usuario que ha iniciado sesión no tiene permiso automáticamente para aprobar una orden de trabajo, ver otro tenant, exportar datos o ejecutar una tarea de administración. La comprobación de permisos pertenece al servicio que ejecuta la acción.

El contexto de sesión une la identidad y la UI

Tras la autenticación, la aplicación debería convertir la identidad en un contexto de sesión: ID de usuario, nombre visible, tenant, roles, permisos, referencia cultural, tema e IDs de correlación. Este contexto ayuda a las pantallas a adaptarse, mientras los servicios siguen aplicando las reglas.

Un modelo de identidad y autorización para la empresa

La autorización empresarial tiene más dimensiones que un nombre de rol. La misma persona puede ser operador de despacho en un tenant, auditor de solo lectura en otro y tener permiso para aprobar órdenes

de trabajo solo por debajo de un límite de coste. Un modelo capaz de expresar eso empieza con tres capas: la identidad, que dice quién es el usuario; la pertenencia, que dice qué tenants y roles tiene; y los permisos, que son las acciones detalladas que concede un rol, como `ApproveWorkOrder` o `ExportData`.

El laboratorio convierte esto en una matriz de permisos: las filas son permisos, las columnas son roles y cada celda está concedida, denegada o condicionada. La matriz es el artefacto que pueden leer tanto los propietarios de producto como los revisores de seguridad y los desarrolladores, así que debería existir antes que el código. Los pasos del flujo de trabajo también se reflejan en ella. Enviar un escalado, aprobarlo y cerrarlo pueden requerir permisos distintos, y la matriz lo hace explícito.

Mantén el modelo centrado en el tenant. Un permiso siempre se evalúa dentro de un contexto de tenant, de modo que un usuario con un rol en el tenant A no obtiene nada en el tenant B. La llamada `PermissionService.Demand` del laboratorio comprueba primero la protección de tenant y solo después consulta el almacén de roles.

Controles de autenticación, contexto de sesión y servicios

Un proveedor de identidades externo, al que se accede mediante un inicio de sesión OpenID Connect o SSO, le dice a la aplicación quién es el usuario. Ahí termina el trabajo del proveedor. A continuación, la aplicación convierte los claims recibidos en su propio contexto de sesión: ID de usuario, nombre visible, tenant, roles, permisos y un ID de correlación. Como el mapeo se ejecuta una vez al iniciar sesión y se guarda en la sesión, las pantallas pueden leerlo con poco coste y los servicios pueden confiar en él sin repetir el análisis de los claims.

El control debe situarse delante de cada punto de entrada, no solo de la página principal. Los trabajos en segundo plano, las llamadas a WebMethod, los manejadores de postback y las páginas de diagnóstico necesitan saber quién llama. Un servicio nunca debería leer la identidad de un control de la UI, de una cadena de consulta o de un campo oculto, porque el usuario puede modificar todos ellos. Lee el contexto de sesión, y el contexto de sesión se construyó a partir de claims verificados.

Diseña el contexto para que se pueda refrescar. Los roles cambian mientras hay sesiones abiertas, y una sesión de larga duración debe poder recargar los permisos sin obligar a iniciar sesión de nuevo. La pregunta de revisión para cada método de servicio es sencilla: ¿qué línea demuestra que quien llama tiene permiso para estar aquí?

Pistas de auditoría para comandos sensibles

Una pista de auditoría responde quién hizo qué, sobre qué registro, en qué tenant, cuándo y con qué resultado. La escribe el servicio que ejecuta el comando, dentro de la misma operación, para que una acción no pueda tener éxito sin quedar registrada. La pantalla del registro de auditoría del laboratorio muestra cada `Demand`, concedido o denegado, y cada comando sensible, como una aprobación, una exportación o un cambio de permisos.

Las denegaciones son tan importantes como los éxitos. Si un botón habilitado por error permite a un usuario solicitar `ExportData`, el servicio lo detecta, lo rechaza con un error con nombre y escribe una entrada de denegación. Las denegaciones repetidas desde una misma cuenta indican que alguien está sondeando la aplicación, y solo son visibles si se registran. La entrada debería llevar el ID de correlación para enlazarla con los registros estructurados del Módulo 11.

- Actor, tenant e indicador de suplantación, si lo hay.
- Nombre del comando, registro de destino y los valores anteriores y posteriores que importan.
- Desenlace: concedido, denegado o fallido, con el motivo con nombre.
- Marca de tiempo e ID de correlación, escritos en la misma transacción que el cambio.

Una lista de comprobación de seguridad para toda la superficie

El renderizado en el servidor elimina muchos riesgos del lado del navegador, pero una aplicación de Wisej.NET sigue enviando HTML, aceptando cargas de archivos, estableciendo cookies, escribiendo registros, leyendo secretos y funcionando detrás de un proxy. Cada uno de esos puntos es un lugar donde un pequeño error se convierte en un incidente. La lista de comprobación de endurecimiento del laboratorio es el documento que convierte esos puntos en elementos de revisión que se comprueban en cada versión, en lugar de recordarse de vez en cuando.

Empieza por el HTML. Cualquier control que renderice HTML sin procesar a partir de la entrada del usuario debe sanearlo, y la revisión de `AllowHtml` del laboratorio enumera cada uno de esos controles. Las cargas de archivos necesitan límites de tamaño, comprobaciones de tipo y almacenamiento fuera de la raíz web. Las cookies deberían marcarse como secure y HTTP-only, con una política same-site. Una política de seguridad de contenido limita qué scripts se pueden ejecutar. Los registros nunca deben contener contraseñas, tokens ni registros personales completos. Los secretos viven en un almacén de secretos o en una configuración protegida, nunca en el control de código fuente. Un proxy inverso debe reenviar el esquema y la dirección del cliente originales para que la aplicación vea los valores reales.

Trata la lista de comprobación como código vivo: añade un elemento cada vez que una revisión detecte una carencia y registra quién comprobó cada elemento en cada versión.



Regla avanzada de producción

No te detengas cuando la pantalla parezca correcta. Explica a quién pertenece el estado, las implicaciones de seguridad, las rutas de fallo y el comportamiento en producción antes de dar el módulo por terminado.



Objetivos de aprendizaje

Al terminar este módulo serás capaz de:

- ✓ Diseñar un modelo de identidad y autorización para roles, tenants y flujos de trabajo empresariales.
- ✓ Integrar los controles de autenticación con el contexto de sesión y los servicios de aplicación.
- ✓ Construir pistas de auditoría para los comandos sensibles.
- ✓ Crear listas de comprobación de seguridad para HTML, cargas de archivos, cookies, CSP, registros, secretos y proxies inversos.

Guía del laboratorio / examen

Lo que construirás en el laboratorio práctico, el enfoque sugerido y los entregables requeridos.



Laboratorio práctico

Añade una simulación de autenticación empresarial, el mapeo de claims a permisos, la aplicación de roles por tenant, autorización a nivel de servicio, la revisión de HTML seguro, la aprobación de exportaciones y el registro de auditoría de las acciones sensibles.

Enfoque sugerido

- ✓ Lee el problema en producción e identifica el riesgo de arquitectura antes de tocar la UI.
- ✓ Crea o actualiza primero el límite correspondiente de servicio, comando, modelo o componente.
- ✓ Construye la pantalla o el componente Wisej.NET siguiendo los estándares de proyecto establecidos.
- ✓ Añade una demostración de la ruta de fallo, no solo del caso ideal.
- ✓ Anota lo que ha cambiado en el diario de aprendizaje y actualiza la lista de comprobación correspondiente.

Entregables requeridos

- ✓ Diseño del mapeo de identidades
- ✓ Matriz de permisos
- ✓ Implementación del servicio de permisos
- ✓ Pantalla del registro de auditoría
- ✓ Lista de comprobación de endurecimiento de la seguridad

Preguntas de repaso

- ✓ ¿Dónde se aplica la autorización?
- ✓ ¿Qué elementos de la UI renderizan HTML?
- ✓ ¿Puede un botón habilitado eludir las reglas del servicio?



Regla avanzada de producción

No te detengas cuando la pantalla parezca correcta. Explica a quién pertenece el estado, las implicaciones de seguridad, las rutas de fallo y el comportamiento en producción antes de dar el módulo por terminado.