

Nube, contenedores, balanceo de carga e ingeniería de versiones

Guía de la lección · Guía del laboratorio / examen

● LearnWisej

Guía de la lección

IIS, Kestrel, proxies inversos, contenedores, sticky sessions, health checks, CI/CD, runbooks. Lo que esto significa para un equipo avanzado de Wisej.NET — y cómo leer este módulo.



Enfoque de producción

IIS, Kestrel, proxies inversos, contenedores, sesiones persistentes, comprobaciones de estado, CI/CD, runbooks.

1. El problema en producción

Las aplicaciones avanzadas de Wisej.NET deben ir más allá de las máquinas de los desarrolladores y pasar a entornos repetibles con disciplina de configuración, comprobaciones de estado, balanceo de carga, reversión y responsables operativos.

2. Conceptos clave

Concepto	Por qué importa
IIS	El servidor web de Windows que aloja aplicaciones Wisej.NET en muchas empresas, con grupos de aplicaciones, reglas de reciclaje y compatibilidad con websockets que hay que configurar.
Kestrel	El servidor web multiplataforma de ASP.NET Core que ejecuta directamente una aplicación Wisej.NET, normalmente detrás de un proxy inverso en producción.
proxy inverso NGINX/Apache	Un servidor frontal que termina TLS, reenvía el tráfico websocket y dirige las solicitudes a una o varias instancias de la aplicación.
Azure/AWS	Plataformas en la nube que alojan la aplicación como servicios de aplicaciones, contenedores o máquinas virtuales, cada una con su propio modelo de escalado y configuración.
Docker	Empaqueta la aplicación y su runtime en una única imagen, de modo que la misma compilación se ejecuta de forma idéntica en la máquina de un desarrollador, en un servidor de pruebas y en la nube.
HealthCheck.json	La configuración de comprobación de estado de Wisej.NET que permite a los balanceadores de carga y a los orquestadores sondear si una instancia está lista para atender sesiones.
sesiones persistentes	La afinidad del balanceador de carga que dirige todas las solicitudes de un usuario a la misma instancia del servidor, algo imprescindible porque el estado de la sesión vive en la memoria del servidor.
CI/CD	Canalizaciones automatizadas de compilación, pruebas y despliegue que convierten cada fusión en una versión repetible en lugar de un procedimiento manual y propenso a errores.

3. Desarrollo de la lección

El despliegue forma parte de la arquitectura

El alojamiento cambia el comportamiento de la identidad, los WebSockets, el registro, la configuración, el almacenamiento de archivos, las comprobaciones de estado y el escalado. Los desarrolladores avanzados diseñan para el entorno de destino lo bastante pronto como para evitar sorpresas.

Escalado consciente de las sesiones

Las aplicaciones Wisej.NET mantienen un estado rico en el servidor. En entornos con balanceo de carga, la infraestructura debe dirigir cada sesión de forma coherente o usar un modelo de despliegue que conserve el comportamiento de las sesiones. Las comprobaciones de estado ayudan a desviar el tráfico de los nodos que no están en buen estado.

Elegir un destino de despliegue

Una aplicación Wisej.NET se ejecuta en cualquier lugar donde se ejecute ASP.NET Core, lo que significa que la elección depende de la organización y no del framework. IIS en Windows encaja con equipos que ya tienen operaciones Windows, gestión de grupos de aplicaciones y autenticación de Windows. Kestrel detrás de un proxy inverso NGINX o Apache encaja con servidores Linux y deja en manos del proxy la terminación TLS y el reenvío de WebSockets. Azure App Service y sus equivalentes de AWS eliminan la gestión de servidores, pero añaden configuración específica de la plataforma, reglas de escalado e integración del registro.

Los contenedores cambian la unidad de despliegue, no el anfitrión. Una imagen Docker agrupa la aplicación y su runtime para que el mismo artefacto se ejecute en la máquina de un desarrollador, en un clúster de pruebas y en producción. La imagen sigue necesitando alojarse en algún sitio, y ese anfitrión sigue necesitando los ajustes de WebSocket, comprobación de estado y afinidad de sesión de los que depende la aplicación.

Sea cual sea el destino, verifica tres cosas antes de comprometerte: que los WebSockets atraviesan todas las capas entre el navegador y la aplicación, que la infraestructura puede alcanzar el endpoint de comprobación de estado y que la aplicación ve la dirección y el esquema reales del cliente a través de las cabeceras del proxy.

Balancear la carga de una aplicación consciente de las sesiones

Cada sesión de Wisej.NET vive en la memoria de un único proceso de servidor. Un balanceador de carga que repartiera las solicitudes entre nodos sin afinidad enviaría la segunda solicitud de un usuario a un nodo que nunca ha oído hablar de él, y la aplicación iniciaría una nueva sesión. Por tanto, las sesiones persistentes, en las que el balanceador dirige todo el tráfico de un cliente al mismo nodo, son un requisito y no una optimización. Las notas de balanceo de carga del laboratorio lo dejan claro para el equipo de operaciones.

Las funciones en tiempo real añaden una segunda restricción. El push desde el servidor depende de un WebSocket que permanece abierto entre el navegador y el nodo que contiene la sesión, así que el balanceador y cualquier proxy deben mantener vivas las conexiones de larga duración y no cortarlas por inactividad. Escalar horizontalmente significa, por tanto, añadir nodos que alojan cada uno una parte de las sesiones, y reducir la escala significa vaciar un nodo de forma ordenada para que sus sesiones terminen en lugar de cortarse.

Las comprobaciones de estado alimentan al balanceador. En la ruta de fallo del laboratorio, el nodo B no supera su comprobación, el balanceador desvía las nuevas sesiones y los usuarios del nodo A no notan nada. Documenta qué ocurre con las sesiones que ya estaban en el nodo B, porque es lo primero que preguntará operaciones.

Canalizaciones, pruebas de humo, reversión y runbooks

Una canalización de versiones convierte cada fusión en la misma secuencia: compilar, ejecutar las pruebas, producir un artefacto versionado, desplegarlo en un entorno de staging, ejecutar las pruebas de humo y promoverlo. Su valor está en la repetición. Cuando todas las versiones siguen el mismo camino, un fallo es una diferencia que investigar y no un misterio. El artefacto está versionado, así que el panel de versiones puede mostrar que se está desplegando la 2.4.2 y que la 2.4.1 es la que se ejecutaba antes.

Las pruebas de humo son las pocas comprobaciones que demuestran que la aplicación desplegada está viva: la página de inicio de sesión se renderiza, se inicia una sesión, la comprobación de estado devuelve un resultado correcto, una consulta conocida devuelve filas y un WebSocket se conecta. Se ejecutan en minutos y contra el entorno real. El plan de reversión es lo que ocurre cuando fallan. Indica el artefacto anterior, el comando que lo vuelve a desplegar, la persona que puede ejecutarlo y cómo confirmar que ha funcionado.

El runbook reúne todo esto en el documento que lee un ingeniero de guardia a las dos de la madrugada. El runbook del laboratorio termina con la reversión, porque un procedimiento de versión que no se puede deshacer no está terminado.

La configuración como contrato

La misma compilación debe ejecutarse en desarrollo, en pruebas y en producción, así que nada específico de un entorno puede vivir en el código. Las cadenas de conexión, las URL del proveedor de identidades, las cuentas de almacenamiento, los feature flags, los niveles de registro y los hosts permitidos son configuración, la proporciona el entorno y se lee al arrancar. La tabla de configuración de entornos del laboratorio enumera cada uno de esos ajustes, su origen en cada entorno y quién es responsable de cambiarlo.

Wisej.NET lee sus propios ajustes de `Default.json` y el comportamiento de la comprobación de estado de `HealthCheck.json`. Los ajustes de la aplicación siguen el modelo de configuración de ASP.NET Core, de modo que una variable de entorno o un archivo de secretos montado puede sobrescribir un valor de un archivo sin tocar la imagen. Los secretos nunca aparecen en los archivos de configuración confirmados en el repositorio. Se inyectan en el momento del despliegue desde un almacén de secretos o desde el almacén de secretos de la plataforma.

- ¿Se puede desplegar el mismo artefacto en todos los entornos cambiando solo la configuración?
- ¿Tiene cada ajuste de la tabla un responsable y un origen por entorno?
- ¿Expondría algún secreto una filtración del repositorio?



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:

- ✓ Elegir destinos de despliegue entre IIS, Kestrel con proxy inverso, Azure, AWS y contenedores.
- ✓ Preparar notas de balanceo de carga para aplicaciones en tiempo real conscientes de las sesiones.
- ✓ Crear canalizaciones de versiones, pruebas de humo, planes de reversión y runbooks.
- ✓ Separar la configuración de entorno del código de la aplicación.

Guía del laboratorio / examen

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



Laboratorio práctico

Crea un paquete de despliegue para EnterpriseOps con configuración específica de cada entorno, HealthCheck.json, lista de comprobación de pruebas de humo, notas de ejemplo sobre contenedores, notas sobre el proxy inverso, pautas de sesiones persistentes y balanceo de carga, y un runbook de reversión.

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

- ✓ Diagrama de la arquitectura de despliegue
- ✓ Tabla de configuración de entornos
- ✓ HealthCheck.json
- ✓ Runbook de versiones
- ✓ Lista de comprobación de reversión y pruebas de humo

Preguntas de repaso

- ✓ ¿Qué requiere sesiones persistentes?
- ✓ ¿Dónde se guardan los secretos?
- ✓ ¿Cómo sabe operaciones que un nodo no está en buen estado?



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.