

Osservabilità, diagnostica, prestazioni e profiling delle sessioni

Guida della lezione · Guida al lab / esame

 LearnWisej

Guida della lezione

Salute, log, metriche, trace, tempi client/server, crescita di memoria e sessione. Che cosa significa per un team Wisej.NET avanzato — e come affrontare questo modulo.



Focus di produzione

Salute, log, metriche, trace, tempi client/server, crescita di memoria e sessione.

1. Il problema in produzione

I team avanzati devono diagnosticare i problemi in produzione senza tirare a indovinare da remoto. Hanno bisogno di diagnostica sicura, log strutturati, misure delle prestazioni e pratiche di revisione di sessione e memoria.

2. Concetti chiave

Concetto	Perché è importante
logging strutturato	Scrivere le voci di log come campi con un nome anziché come testo libero, così in produzione gli strumenti possono cercarle, filtrarle e correlarle.
correlation ID	Un identificativo univoco riportato in ogni voce di log di una richiesta o di un job, così una singola azione dell'utente può essere seguita attraverso servizi e server.
pagina di diagnostica	Una schermata riservata agli operatori che mostra numero di sessioni, memoria, job attivi ed errori recenti, dando visibilità senza collegare un debugger.
health check	Un endpoint leggero che riporta se l'applicazione e le sue dipendenze funzionano, usato dai load balancer e dal monitoraggio.
budget di prestazioni	Limiti concordati per tempo di risposta, dimensione del payload e memoria per sessione, così le regressioni vengono intercettate rispetto a un numero anziché a una sensazione.
memoria di sessione	La memoria del server consumata da ogni sessione attiva. Deve restare piccola e prevedibile, perché un server Wisej.NET ospita molte sessioni contemporaneamente.
revisione dei dispose	Verificare che timer, gestori di evento, DbContext e task in background vengano rilasciati quando un form o una sessione terminano, così la memoria non perde.
gestione sicura degli errori	Intercettare e registrare le eccezioni con tutti i dettagli sul server mostrando agli utenti un messaggio generico, così i dettagli interni non vengono mai esposti.

3. Il filo della lezione

In produzione non c'è il debugger

Un runbook di produzione dovrebbe rispondere a quale versione è in esecuzione, in quale ambiente, quali servizi sono in salute, quali errori si sono verificati e quale correlation ID collega la segnalazione dell'utente ai log.

La diagnostica deve essere sicura

Una schermata di diagnostica dovrebbe essere protetta da un ruolo e oscurare i segreti. Può mostrare ambiente, versione, feature flag, tema attivo, stato dei servizi, risultato dell'health check, stato delle code e log recenti sicuri.

Pagine di diagnostica che aiutano senza esporre segreti

Una pagina di diagnostica è il primo posto in cui guarda un tecnico del supporto quando arriva un ticket. Dovrebbe rispondere, in un'unica schermata, a quale versione e build è in esecuzione, quale ambiente e quale nodo hanno servito la sessione, quali feature flag sono attivi, se i servizi da cui dipende sono in salute e quanto sono durate le operazioni recenti. La pagina del lab mostra versione, nodo, flag e budget di prestazioni affiancati, così la risposta alla maggior parte delle prime domande richiede un'occhiata.

La pagina è anche una superficie di attacco, quindi i suoi contenuti vengono scelti per tipo e non a mano. Il `DiagnosticSnapshot` del lab contiene solo campi sicuri da mostrare: niente stringhe di connessione, niente chiavi API, niente dati personali, niente stack trace completi. I valori che potrebbero essere sensibili, come il nome di uno storage account, vengono oscurati prima di arrivare al view model. La pagina è protetta da un controllo di ruolo, e anche la sua apertura viene sottoposta ad audit.

Una buona domanda di revisione è se la pagina potrebbe essere proiettata davanti a un cliente. Se qualche valore andrebbe coperto, non dovrebbe esserci.

Log strutturati, correlazione e messaggi sicuri

Logging strutturato significa che ogni voce è un insieme di campi con un nome anziché una frase. Una voce per una query lenta registra come valori separati il nome dell'operazione, il tenant, l'utente, la durata in millisecondi, la dimensione della pagina e il correlation ID. L' `OperationTimer` del lab racchiude ogni operazione di servizio e scrive una voce di questo tipo quando termina, così in seguito il log può essere filtrato per ottenere tutte le operazioni sopra i due secondi in un tenant.

Il correlation ID è ciò che lega insieme i pezzi. Viene creato quando inizia un'azione dell'utente, portato nel contesto del comando attraverso ogni chiamata di servizio, passato ai job in background e scritto nelle voci di audit e nei report di errore. Quando un utente segnala che un'approvazione è fallita alle 14:32, il supporto trova il correlation ID nell'errore che gli è stato mostrato e lo segue in ogni riga di log prodotta da quell'azione.

Il messaggio che vede l'utente è una cosa diversa dalla voce di log. Dice che l'operazione non è stata completata, fornisce il correlation ID e suggerisce che cosa fare. Non mostra mai il testo dell'eccezione, l'SQL o il percorso sul server, perché aiutano un attaccante più di quanto aiutino l'utente.

Misurare ciò che sentono gli utenti

Le prestazioni si misurano nei punti in cui gli utenti aspettano. L'avvio è il tempo tra la prima richiesta e una pagina principale utilizzabile. Il caricamento di una schermata è il tempo tra un click nel menu e un form visualizzato. Una query sui dati è il tempo che il servizio impiega per recuperare una pagina di risultati. L'aggiornamento del binding è il tempo tra l'arrivo dei dati e la loro comparsa nella griglia. I tempi dei job in background coprono l'attesa in coda e l'esecuzione, e i tempi di aggiornamento coprono quanto impiega una modifica inviata dal server a comparire nel browser.

A ciascuno di questi viene assegnato un budget, cioè un numero sotto il quale il team concorda di restare, registrato nella tabella del budget di prestazioni del lab. Il budget trasforma una lamentela vaga in una regressione misurabile. Il percorso di errore del lab ne mostra il valore: una query che ha impiegato 2.340 ms contro un budget di 500 ms è rossa nella tabella, e la sua voce di log strutturato rivela una dimensione di pagina di 5.000 righe. Correggere la dimensione della pagina fa diventare verde la riga, e il log lo dimostra.

Misura in produzione oltre che in sviluppo, perché volume dei dati, distanza di rete e sessioni concorrenti cambiano i numeri. Un pannello dei tempi che funziona solo sulla macchina di uno sviluppatore è una demo, non uno strumento di diagnostica.

Memoria di sessione ed errori di dispose

Ogni sessione attiva occupa memoria sul server per tutto il tempo in cui esiste, e un server Wisej.NET ospita molte sessioni contemporaneamente. Un form che tiene in un campo un elenco di 50.000 righe, un report in cache o un grande buffer di file moltiplica quel costo per il numero di utenti. La revisione della memoria di sessione del lab elenca che cosa trattiene ogni sessione, perché e per quanto tempo, e mette in discussione ogni elemento oltre qualche megabyte.

Gli errori di dispose sono la seconda fonte di crescita. Un timer avviato in un form e mai fermato mantiene in vita il form dopo la sua chiusura. Un gestore di evento agganciato a un servizio condiviso trattiene un riferimento a un controllo già rilasciato. Un DbContext tenuto in un campo sopravvive all'operazione per cui era stato creato. Un task in background che cattura un form in una closure trattiene tutto ciò a cui il form fa riferimento. Ognuno di questi perde un po' di memoria, e nel corso di una giornata di lavoro quel poco si somma.

- Quali campi di questo form contengono collezioni, e vengono svuotati alla chiusura?
- Quali timer, sottoscrizioni e task avvia questo form, e dove vengono fermati?
- Quali oggetti vengono trattiene per tutta la sessione, e ognuno di essi deve esserlo davvero?



La regola di produzione avanzata

Non fermarti quando la schermata sembra corretta. Prima di considerare completo il modulo, spiega a chi appartiene lo stato, le implicazioni di sicurezza, i percorsi di errore e il comportamento in produzione.



Obiettivi di apprendimento

Alla fine di questo modulo sarai in grado di:

- ✓ Progettare pagine di diagnostica che aiutano le operations senza esporre segreti.
- ✓ Aggiungere logging strutturato, correlation ID e messaggi di errore sicuri per l'utente.
- ✓ Misurare i tempi di avvio, caricamento delle schermate, query sui dati, aggiornamento del binding, job in background e aggiornamenti.
- ✓ Individuare i rischi di crescita della memoria e della sessione e gli errori di dispose.

Guida al lab / esame

Che cosa costruirai nel lab pratico, l'approccio suggerito e i deliverable richiesti.



Lab pratico

Aggiungi l'osservabilità a EnterpriseOps: logging strutturato, correlation ID, pagina di diagnostica, stato di salute, pannello dei tempi di esecuzione e una checklist di revisione della memoria di sessione. Simula una query lenta e mostra come viene diagnosticata.

Approccio suggerito

- ✓ Leggi il problema in produzione e individua il rischio architetturale prima di toccare la UI.
- ✓ Crea o aggiorna per primo l'incapsulamento in un servizio, comando, modello o componente coinvolto.
- ✓ Costruisci la schermata o il componente Wisej.NET seguendo gli standard di progetto stabiliti.
- ✓ Aggiungi una dimostrazione del percorso di errore, non solo del percorso felice.
- ✓ Annota che cosa è cambiato nel registro di apprendimento e aggiorna la checklist relativa.

Deliverable richiesti

- ✓ Pagina di diagnostica
- ✓ Esempio di log strutturato
- ✓ Propagazione del correlation ID
- ✓ Tabella del budget di prestazioni
- ✓ Note dell'audit su memoria e sessione

Domande di revisione per lo studente

- ✓ Il supporto riesce a identificare la versione e il tenant esatti?
- ✓ La diagnostica espone segreti?
- ✓ Quali oggetti vengono trattenuti per tutta la sessione?



La regola di produzione avanzata

Non fermarti quando la schermata sembra corretta. Prima di considerare completo il modulo, spiega a chi appartiene lo stato, le implicazioni di sicurezza, i percorsi di errore e il comportamento in produzione.