Maneja errores sin comprometer la seguridad

Aprende a proteger tu sistema y tus usuarios al manejar excepciones

PABLO DEL ÁLAMO

Introducción

El problema:

Cuando ocurre un error, muchos sistemas muestran información detallada.

Esto puede revelar datos sensibles, como rutas, configuraciones o credenciales.

PABLO DEL ÁLAMO

Ejemplo de un error inseguro

NullPointerException at
com.example.App.main(App.java:15)

Caused by: SQL syntax error near 'DROP DATABASE'

Este mensaje expone detalles internos y posibles vulnerabilidades.

PABLO DEL ÁLAMO

¿Por qué es un problema?

1️⃣ Revela detalles del sistema (nombres de clases, rutas, tecnologías).

2️⃣ Facilita ataques como inyección SQL o fuerza bruta.

3️⃣ Genera desconfianza en los usuarios.

PABLO DEL ÁLAMO

Buenas prácticas al manejar errores

Diseña un sistema de manejo de errores que priorice la seguridad y la usabilidad.

PABLO DEL ÁLAMO

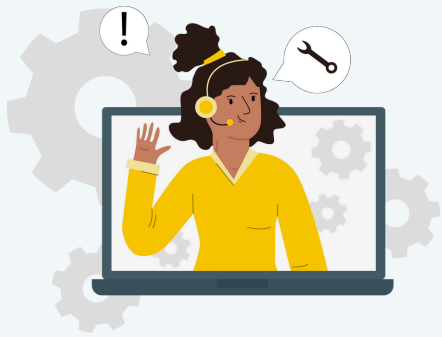
No expongas detalles técnicos

⚠ En lugar de esto:

- Error en la consulta SQL: código 500.

Muestra mensajes genéricos:

- "Ha ocurrido un error. Por favor, inténtalo más tarde."



PABLO DEL ÁLAMO

Usa códigos de error en lugar de detalles

Asigna códigos específicos que los usuarios puedan compartir. Ejemplo: "Error: 1001 - Servicio no disponible."

El equipo técnico puede rastrear este código en los logs.

PABLO DEL ÁLAMO

Registra los detalles en los logs internos

Muestra solo lo necesario al usuario y guarda los detalles técnicos en un lugar seguro. Ejemplo de log:

- [2025-01-12 10:23:45] Error: NullPointerException Clase: UserService.java, Línea: 42

PABLO DEL ÁLAMO

Controla las excepciones inesperadas

Nunca dejes excepciones sin capturar.

❌ Ejemplo inseguro:

```
try {
    // Código...
} catch (Exception e) {
    throw e;
}
```

✅ Solución segura:

```
catch (Exception e) {
    logger.error("Error inesperado", e);
    throw new CustomException("Ocurrió un error. Por favor, inténtalo más tarde.");
}
```

PABLO DEL ÁLAMO

Configura errores genéricos en producción

- En desarrollo: Errores detallados.
- En producción: Mensajes genéricos y logs robustos.

- Ejemplo para frameworks como Spring Boot: server.error.include-message: never

PABLO DEL ÁLAMO

Sanitiza tus mensajes de error

Si usas entradas de usuario, asegúrate de no reflejarlas en los mensajes de error.

Ejemplo inseguro:

- "El usuario admin'; DROP TABLE users; no existe."

Respuesta segura:

- "Usuario o contraseña incorrectos."

PABLO DEL ÁLAMO

Implementa manejo global de errores

Crea un manejador global para centralizar las respuestas de error. Ejemplo en Spring Boot:

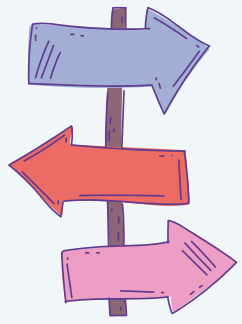
```java
@ControllerAdvice
public class GlobalExceptionHandler {
   @ExceptionHandler(Exception.class)
   public ResponseEntity<String> handleAllExceptions(Exception ex) {
       return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR)
                            .body("Ha ocurrido un error.");
   }
}
```

PABLO DEL ÁLAMO

Proporciona una experiencia de usuario amigable

No solo informes el error, también guía al usuario:

- ¿Qué puede hacer ahora?
- ¿Hay un contacto de soporte disponible?



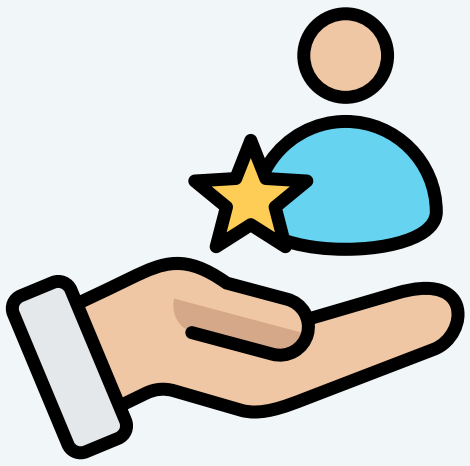
PABLO DEL ÁLAMO

Ejemplo de mensaje seguro

Título: ¡Algo salió mal!

Mensaje: "Hubo un problema al procesar tu solicitud. Por favor, intentalo de nuevo más tarde."

Contacto: soporte@tuempresa.com



PABLO DEL ÁLAMO

Usa herramientas de monitoreo

Implementa herramientas para capturar y analizar errores, como:

- Sentry
- Logstash
- Splunk

PABLO DEL ÁLAMO

Considera la seguridad en APIs

En APIs, devuelve solo el estado HTTP adecuado:

✅ Ejemplo seguro:

```
{
  "error": "Internal Server Error",
  "code": 500
}
```

No expongas detalles internos como stacks o queries.

PABLO DEL ÁLAMO

Beneficios de manejar errores correctamente

- Protección contra ataques.
- Mejora la experiencia del usuario.
- Facilita la resolución rápida de problemas.

PABLO DEL ÁLAMO

Conclusión

La gestión de errores no solo mejora la seguridad, sino también la experiencia del usuario y la eficiencia del equipo técnico.

Evitar mensajes con detalles sensibles, registrar errores internamente y mostrar respuestas claras pero genéricas reduce riesgos y aumenta la confianza en tus sistemas.

Recuerda: un manejo de excepciones bien diseñado no solo protege tus datos, sino también tu reputación.

PABLO DEL ÁLAMO

¿Te ha resultado útil?

- Comparte esta guía con tu equipo o amigos desarrolladores.
- Guárdala para tenerla siempre a mano.
- ¡Dale un like o comenta si tienes preguntas!

PABLO DEL ÁLAMO