Engineer Asesoramiento

Pruebas y control de calidad

21 logros

Conecte las partes de su sistema

Los trabajos reunidos en «Pruebas y control de calidad» muestran una parte de un sistema en funcionamiento. Podemos ayudar a conectar esa parte con el resto de su operación. Cambios probados, publicados y reversibles, para que lleguen a sus clientes un martes por la tarde en vez de un viernes por la noche.

Explore nuestros servicios Póngase en contacto →

Encuentre un reto comparable

Siga un ejemplo que se parezca a su situación. Su reseña describe las decisiones detrás del resultado, más allá de la etiqueta tecnológica. Procesos renovados: 8.000 horas ahorradas — 8,000 horas ahorradas al reorganizar la forma de trabajar

Procesos renovados: 8.000 horas ahorradas Póngase en contacto →

CI/CD optimizado: 4.000 horas ahorradas

Simplifiqué los procesos CI/CD y ahorré 4.000 horas mediante la automatización de los flujos de desarrollo de software.

API migrada a Huma v2 con paridad del 100 %

Migré la API HTTP de Fiber a Huma v2 — 649 rutas y 760 operaciones — y alcancé y mantuve una correspondencia del 100 % entre las rutas que registra el servidor y la descripción OpenAPI que publica.

578 tareas de automatización go-task

Redacté 578 tareas de automatización go-task para los modos de desarrollo nativo, Docker y HTTPS, así como para análisis estático, pruebas, bases de datos y despliegue.

Cero advertencias en seis lenguajes

Fijé un criterio de calidad de cero advertencias en seis lenguajes — Go, TypeScript, SQL, Python, Shell y Markdown — aplicado mediante hooks de pre-commit.

Pruebas para Go, SQL y navegador

Desarrollé una batería de pruebas automatizadas por capas — 981 pruebas Go, 543 especificaciones de interfaz y navegador y 494 pruebas de comportamiento SQL — con pruebas de mutación, pruebas basadas en propiedades y un control obligatorio de accesibilidad.

Motor de 268 controles para el repositorio

Desarrollé el motor de comprobaciones del repositorio — 268 controles de commit registrados, 277 reglas de análisis estático y 15 reglas ESLint propias —, además de 146 pruebas de los propios controles, para que la compilación haga cumplir el criterio de calidad.

Control de commits con 22 analizadores

Desarrollé un control obligatorio de commits con 22 analizadores de una línea y cinco que requieren una explicación, sin nivel de advertencias ni supresiones en línea permitidas, que cubre HTML, CSS, JavaScript, Python, YAML, Markdown, Shell, enlaces, ortografía, secretos y tipografía.

Una red de seguridad que ya no cuesta media hora

Reduje el tiempo del control de calidad del sitio basado en navegador de 1.636 segundos a 615 al ordenar las comprobaciones de mayor a menor duración y ejecutarlas en un grupo limitado a cuatro trabajadores, tras medir que el orden alfabético requería 320 segundos frente a 224.

Analizador propio para la documentación

Sometí la documentación a un analizador propio — límite de líneas por archivo que solo puede reducirse, índice obligatorio y comprobación de rutas reales — que detectó seis de nueve planes marcados como terminados con tareas pendientes y catorce tareas obligatorias para commits que no figuraban en ningún documento.

Control de calidad de 10.242 líneas analizado

Apliqué un formateador y un analizador a 10.242 líneas de JavaScript del control de calidad tras constatar que eran el mayor bloque de código del repositorio y el único que ninguna herramienta revisaba; corregí 13 hallazgos sin suprimir ninguno.

Generador: 981 pruebas y 92 % de cobertura

Mantuve el generador bajo 981 casos de prueba con un mínimo del 92 % de cobertura de ramas y las advertencias tratadas como errores, y verifiqué la idempotencia ejecutando dos veces todo el flujo de construcción desde un archivo vacío y exigiendo que la segunda pasada no cambiara nada.

106 de 106 comprobaciones del perfil PDF/UA-1

Registré la validación PDF/UA-1 en nueve documentos, con 106 de 106 comprobaciones superadas en el perfil de accesibilidad configurado, mientras la variante de archivo incumplía una de 146 reglas.

Cada regla del analizador tratada como error

Activé como errores todas las reglas del analizador Python y resolví 1.815 hallazgos hasta llegar a cero, con un motivo escrito para cada una de las escasas excepciones y dos de ellas respaldadas por una comprobación, además de un comentario.

Pruebas para los propios verificadores

Escribí pruebas para los propios verificadores tras constatar que un verificador que solo recibe entradas correctas puede acabar informando de éxito por no haber leído nada; introduje una falta de ortografía para confirmar su detección y obtuve un intervalo de identificadores de la base de datos en vez de fijarlo en la prueba.

Diferencias detectadas entre hooks y controles

Detecté que los hooks de commit y el control de calidad ejecutaban comprobaciones distintas aunque un documento prometía que eran iguales, al comparar ambas listas en una prueba: las cinco que solo se ejecutaban a mano eran las que revisaban la redacción del CV.

Comprobación de cifras entre idiomas

Desarrollé una comprobación de contenido entre idiomas que falla si una traducción omite una cifra del inglés o usa una notación ajena al idioma; detectó dos descripciones danesas sin una métrica y dieciséis pasajes ucranianos con comillas del estilo inglés.

Concurrencia estricta Swift 6 sobre C

Adopté la concurrencia estricta completa de Swift 6 sin actores, conectando un bucle de eventos C bloqueante con el actor principal mediante un productor, un consumidor y un único orden, tras constatar que una tarea por evento pierde el orden del que depende la interfaz.

Analizador que verifica cada llamada FFI

Escribí un analizador que lee la cabecera C real de 7.308 líneas y verifica cada punto de llamada, cada constante de enumeración y que toda clase propietaria de un puntero sea final, después de que una cabecera provisional escrita a mano permitiera compilar y enlazar llamadas a tres funciones eliminadas que causaban fallos al ejecutarse.

Pruebas reducidas a 8,9 segundos

Pasé de seis pruebas que superaban un límite de sesenta segundos a 135 pruebas satisfactorias en 8,9 segundos al analizar el hilo principal y eliminar las dos llamadas en las que permanecía durante 3.989 de 4.017 muestras.

Errores que dejan de culpar al usuario

Reescribí los mensajes de error de la aplicación conforme a un criterio de tono documentado después de que un inicio de sesión rechazado culpara al usuario de escribir mal la contraseña cuando el proveedor exigía una específica para aplicaciones, y lo cubrí con una prueba que identifica al proveedor.

Seis recolectores declaran lo que omitieron

Hice que seis recolectores de informes declararan la parte de sus entradas que nunca habían leído — archivos no abiertos, sondeos no instalados y líneas del diario no reconocidas — y ajusté los umbrales a 116 muestras medidas tras constatar que una recolección ciega y un servidor sano generan la misma página.

Centre el esfuerzo de ingeniería

Describa la ruta de despliegue, los riesgos operativos y quiénes se encargan de las guardias. Podemos definir el alcance de un cambio con comprobaciones, una vía de recuperación y responsabilidades claras.

Vea cuánto cuesta Póngase en contacto →

Sitúe el tema en su contexto

Cuéntenos cómo aparece «Pruebas y control de calidad» en su propio sistema. Podemos ayudarle a elegir el servicio relevante y definir el trabajo en torno a él.

Explore nuestros servicios

welcome@engineer.company +45 50 10 61 84 Copenhague, Dinamarca

Llévese esta página

Tarjeta de contacto

Creado con esmero y cariño por D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Derechos de autor © 2025 – actualidad · Engineer ApS · Todos los derechos reservados.

Impreso desde https://advisory.engineer.company/es/categories/testing/