
Universidad
Tecmilenio®

Universidad Tecmilenio

Software Testing

Indicadores de calidad

Semana 7

Introducción

Pedro es un ingeniero de pruebas y se le asignó diseñar y ejecutar pruebas de caja blanca a un servicio web para cancelar una factura.

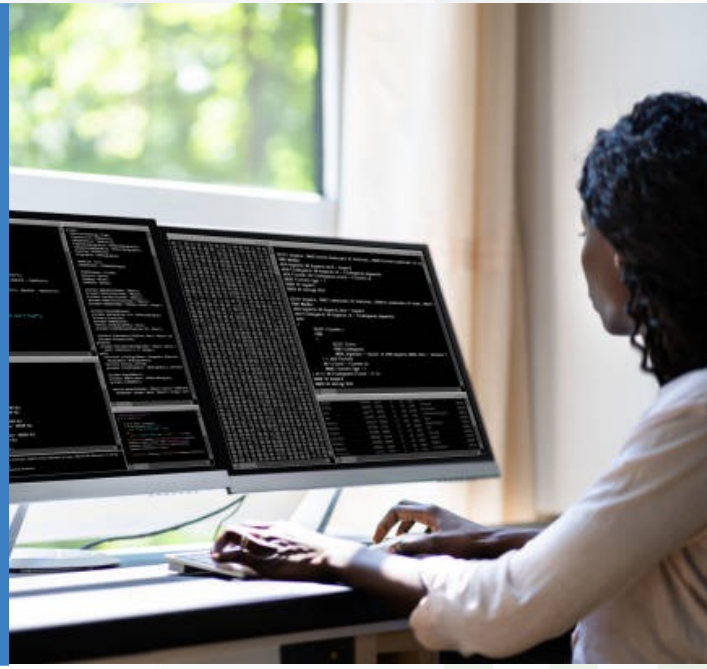
Los interesados del proyecto han solicitado conocer los tipos de pruebas de caja blanca que se diseñarán para probar el servicio web, así como los indicadores de efectividad, cobertura y eficiencia de los resultados que tiene la prueba.

Pedro se pregunta cuáles son las técnicas de pruebas de caja blanca que puede utilizar y que permitan probar los flujos, las condiciones y los bucles del servicio que cancela las facturas.

Explicación

Pruebas de caja blanca

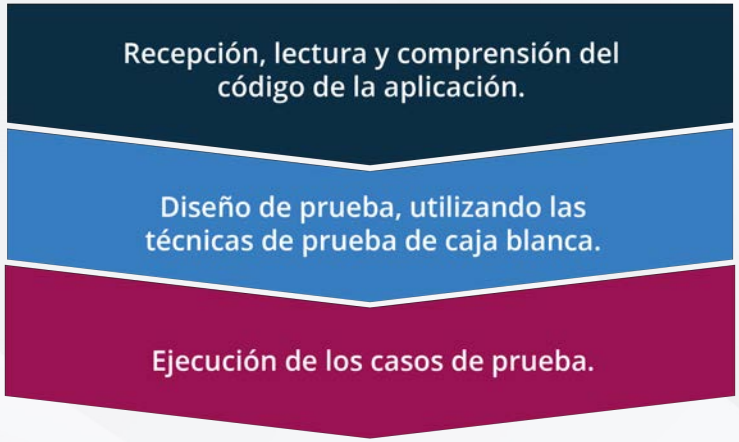
Tienen como objetivo usar la estructura interna de control de un programa y generar casos de prueba que permitan garantizar el funcionamiento correcto de todos sus flujos, la toma de decisiones lógicas, la ejecución de los bucles y el funcionamiento de sus estructuras internas de datos (Mckay, 2021).



Universidad Tecmilenio

Explicación

Proceso para realizar pruebas de caja blanca



Tipos de prueba de caja blanca

- Prueba de ruta básica.
- Prueba de condición.
- Prueba de bucles.

Universidad Tecmilenio

Prueba de ruta básica

Tiene como objetivo generar un conjunto de casos de prueba que deben garantizar que se ejecute cada instrucción del programa por lo menos una vez durante la prueba.

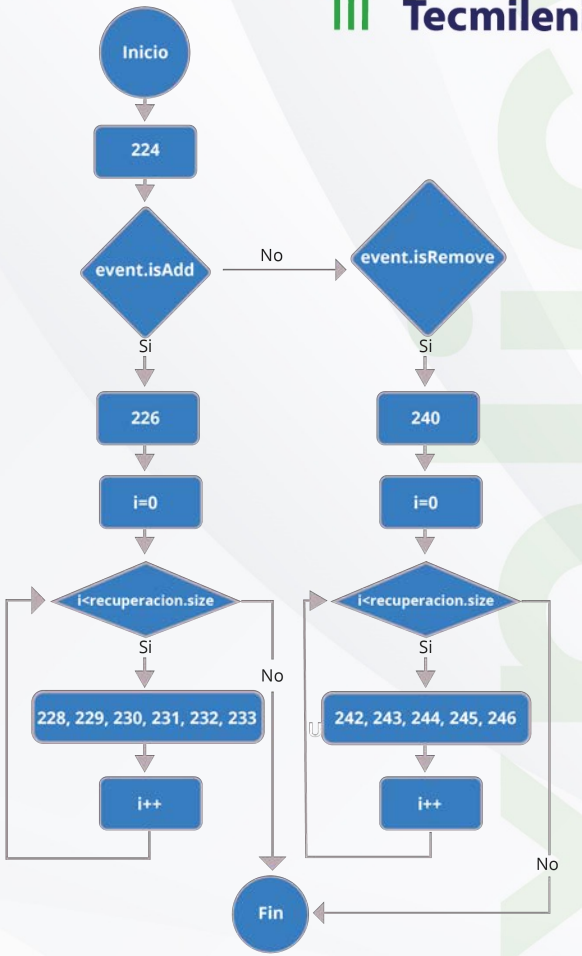
```
223    public void asignarZoonosis(TransferEvent event) {
224        logger.info("asignar Zoonosis");
225        if(event.isAdd()){
226            List<CatalogoDto> recuperacion =(List<CatalogoDto>) event.getItems();
227            for(int i=0; i<recuperacion.size(); i++) {
228                AntecedentePatoDto antec=new AntecedentePatoDto();
229                antec.setCveAntecPatologico(recuperacion.get(i).getCveCatalogo());
230                antec.setCveExpediente(this.cveExpediente);
231                antec.setCveUsuario(this.cveUsuario);
232                logger.info(antec.toString());
233                appServicio.agregarAntecedentePatologico(antec);
234
235            }
236
237
238        }
239        else if(event.isRemove()) {
240            List<CatalogoDto> recuperacion =(List<CatalogoDto>) event.getItems();
241            for(int i=0; i<recuperacion.size(); i++) {
242                AntecedentePatoDto antec=new AntecedentePatoDto();
243                antec.setCveAntecPatologico(recuperacion.get(i).getCveCatalogo());
244                antec.setCveExpediente(this.cveExpediente);
245                antec.setCveUsuario(this.cveUsuario);
246                appServicio.eliminarAntecedentePatologico(recuperacion.get(i).getCveCatalogo(), cveExpediente, cveUsuario);
247
248            }
249        }
250
251
252    }
```

Explicación

Explicación

Prueba de ruta básica

Se deben diseñar casos de prueba donde se forzará la ejecución de cada ruta. El ingeniero de pruebas debe elegir los datos específicos para asegurar las condiciones de la ruta. Finalmente, se ejecutan los casos y se registran los resultados.



Datos que se deben registrar

- Identificador
- Caso de prueba
- Resultado
- Tiempo de procesamiento
- Memoria requerida en la ruta

Explicación

Prueba de condición

Tiene como objetivo generar un diseño de casos de prueba que verifique las condiciones lógicas resguardadas en un programa (Black, 2020).


```
if(event.add()){
    List<CatalogoDto> recuperación=(List<CatalogoDto>
event.getItems);
else if(event.remove()){
   logger.info("Se ha eliminado");
```

Ejemplo: código a probar (condición simple).

Universidad Tecmilenio

Prueba de condición

```
if(conteo<5){
    antec.setCveExpediente(this.cveExpediente)
else
   logger.info("Fin de la condición");
```

Ejemplo: código a probar (condición compuesta).

Se deben preparar dos casos de prueba:

1. El primer caso de prueba debe considerar cuando la variable conteo es menor que 5 y preparamos un escenario con conteo = 1.

2. El segundo caso de prueba debe considerar cuando la variable conteo es mayor que 5 y preparamos un escenario con conteo = 10.

Explicación

Explicación

Prueba de bucles

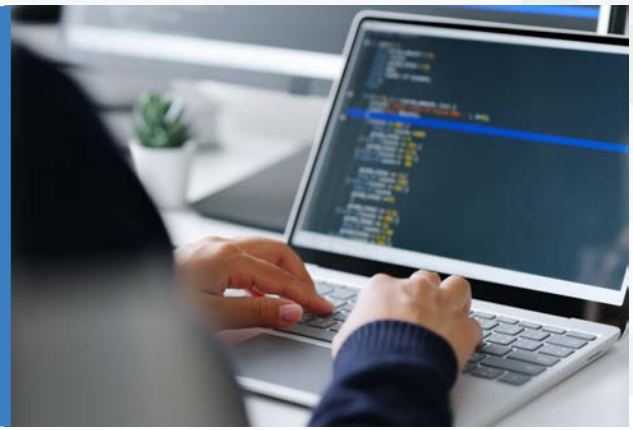
Tiene como objetivo generar un diseño de casos de prueba que verifica la construcción y la validez de los bucles (Black, 2020).

Tipos de bucles

- Simples
- Concatenados
- Anidados
- No estructurados

Casos de prueba a ejecutar para bucles simples

- Omitir el bucle.
- Ejecutar solo un paso por el bucle.
- Dos pasos por el bucle.
- M pasos por el bucle donde m<n.
- N = 1, n, n + 1 pasos por el bucle.



Explicación

Prueba de bucles

Casos de prueba a ejecutar para bucles anidados

- Asignar los valores mínimos del bucle interno y ejecutarlo.
- Asignar los valores mínimos al bucle interno y externo, y ejecutarlo.
- Agregar otras pruebas fuera del valor del rango.
- En caso de existir más bucles anidados, continuar las pruebas con valores mínimos hasta que se prueben todos lo bucles.

Bucles concatenados

- Se usa el enfoque de bucles simples.
- Cuando están concatenados y el contador del bucle usa un valor inicial de 2, se recomienda el de bucles anidados.

Bucles no estructurados

- Se sugiere crear un diseño de acuerdo con las construcciones de la programación estructurada.

Explicación

Indicadores de interés

Indicador de cobertura

- Cuantifica el porcentaje de código cubierto de la app por los casos de prueba diseñados.
- Se encuentran errores en etapas tempranas, garantiza mayor cobertura, ahorra tiempo y previene defectos por caja blanca.

Indicador de efectividad

- Cuantifica el logro de resultados programados en el tiempo óptimo y con costos razonables.
- Evita demoras y costos no necesarios por retrabajo y asegura el cumplimiento de objetivos.

Actividad

¿Cuáles son los tipos de prueba de caja blanca?

¿Qué diferencias existen entre las diversas pruebas de bucles?

Bibliografía

- Black, R. (2020). *Certified Tester Agile*. Alemania: ISTQB.
- McKay, J. (2021). *Certified - Advanced Level Syllabus, Test Manager*. Alemania: ISTQB.

Universidad Tecmilenio

Cierre

Utilizar las pruebas de caja blanca en un proyecto nos ayuda a identificar errores en etapas tempranas, ahorrar tiempo y reducir costos en la solución de defectos. Por esta razón, el indicador de cobertura nos muestra el porcentaje de código que se ha probado.

Usar los indicadores asociados de eficiencia y efectividad nos permite conocer, de manera cuantitativa, el desempeño de los integrantes del equipo respecto al cumplimiento de los objetivos y al gasto de recursos, con el fin de realizar acciones que generen la optimización de estos en los proyectos.

Universidad Tecmilenio

Software Testing

Estrategias de pruebas

Semana 7

Introducción

Cuando los equipos de desarrollo están elaborando un código nuevo para cumplir las expectativas del cliente, es posible que se presenten situaciones en las que los factores como el tiempo, los recursos y los elementos de trabajo presenten áreas de oportunidad y, por esta razón, se realizan pruebas, las cuales se han convertido en procesos esenciales para cualquier equipo de desarrollo que trabaje bajo un esquema de optimización de recursos y centrado en la satisfacción del cliente.

Actualmente se cuenta con diversas herramientas que nos ayudan a realizar pruebas de manera continua y precisa, pero no debe dejarse de considerar la administración del proyecto como pilar de todos los elementos para llegar a una ejecución final satisfactoria.

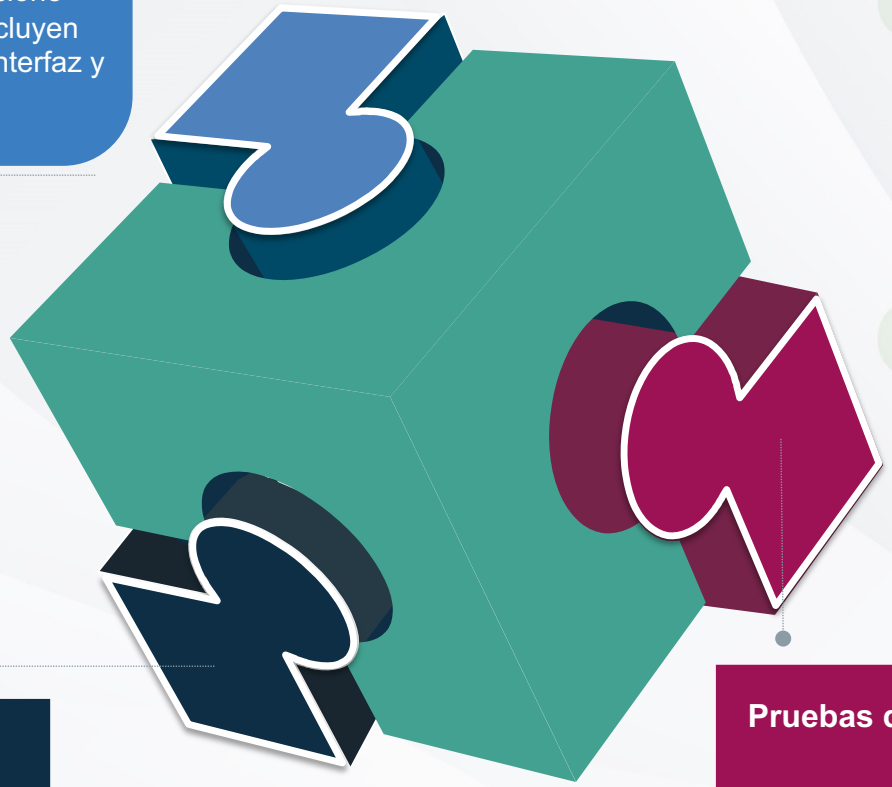
Explicación

Para una estrategia de prueba, se debe contemplar **la totalidad de la aplicación desarrollada para mantenerse de manera exitosa**, entendiendo que un escenario de pruebas puede representar uno o más flujos del negocio.

Explicación

Pruebas funcionales

Para comprobar características críticas del negocio, usabilidad y funcionalidad; garantizan que el software funcione como se espera. Incluyen pruebas unitarias, interfaz y regresión.



Pruebas unitarias

Son el primer tipo de pruebas que se hacen por los desarrolladores para probar unidades individuales del software (unidad puede ser una función o un módulo).

Pruebas de integración

Para probar diversos módulos de la aplicación, como un grupo, ya que algunas aplicaciones integran módulos integrales para lograr su funcionalidad.

Explicación

Pruebas no funcionales

- Parecidas a las pruebas funcionales, sin embargo, tienen la característica de que las funciones se prueban bajo formato de carga, con el fin de determinar el buen rendimiento de los observadores, su fiabilidad, usabilidad, etc.

Pruebas de rendimiento

- Para conocer la estabilidad, escalabilidad y velocidad de una aplicación, con el fin de conocer su rendimiento en diferentes puntos de ejecución del sistema y de la red, como podría ser el uso del CPU, la utilización de recursos del servidor, entre otros.

Explicación

Técnicas de recuperación de proyectos en conflicto

Cuando un proyecto tiene un problema o crisis, es importante conocer cuáles fueron las causas de lo sucedido para gestionar una solución adecuada.

A continuación, se muestran las causas frecuentes que pueden provocar problemas en un proyecto (Nutanix, s.f.):

Explicación

Enfoques para recuperar un proyecto (Zdocs, s.f.):

1. Reducir el tamaño del software, procurando cumplir con los tiempos establecidos.
2. Buscar las mejoras a corto plazo para incrementar la productividad.
3. Tomar conciencia de los tiempos; incluye la posibilidad de retrasar la planificación hasta controlar el problema.

Criterios para evaluar un proyecto:

Explicación

Una vez que se analizan las bases del proyecto, un plan de recuperación debe enfocarse en los pilares del desarrollo rápido.

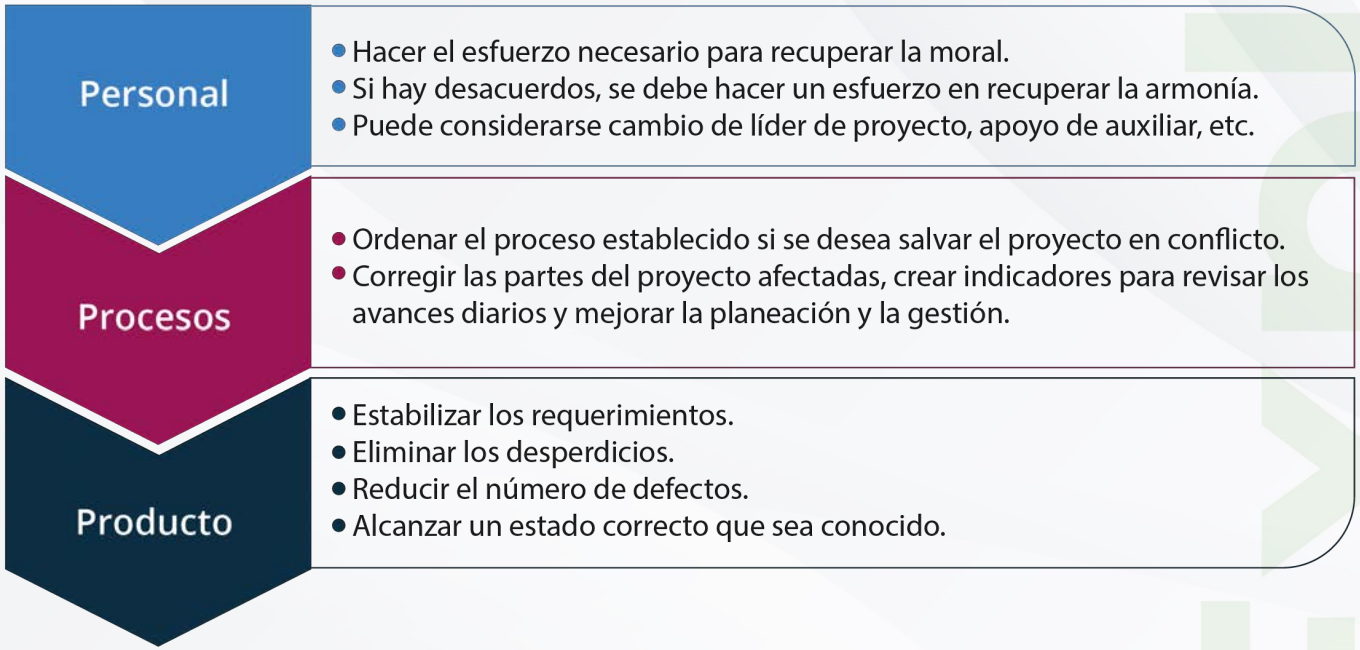
Explicación

Pasos del plan de recuperación

- Evaluar la situación. Establecer el nivel de criticidad de la fecha límite.
- Aplicar el análisis Theory W.
- Preparación para corregir el proyecto, tomar la iniciativa y preparar al equipo para lograr el objetivo.
- Preguntar opciones al equipo, planificar al menos cinco ideas para salvar el proyecto, evaluar y poner en práctica las más adecuadas.
- Ser realistas y presentar el plan para reconstruir el proyecto y establecer una nueva fecha de terminación real.

Explicación

Elementos a tomar en cuenta para dar seguimiento a la técnica de recuperación seleccionada



Actividad

¿Cuál es el proceso para ejecutar un plan de recuperación?

¿Cuáles son las causas más comunes por las que un proyecto entra en conflicto?

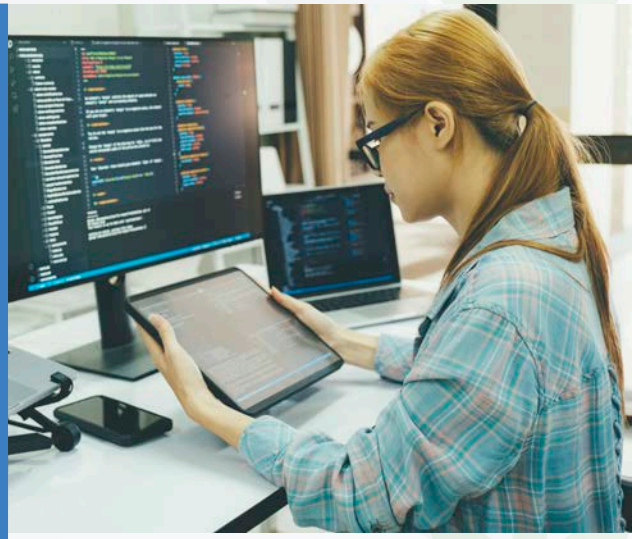
Bibliografía

- Nutanix. (s.f.). *Recuperación ante desastres: Definición y usos.* Recuperado de https://www.nutanix.com/es/info/disaster-recovery
- Zdocs. (s.f.). *Recuperación de Proyectos de Software.* Recuperado de https://zdocs.mx/doc/recuperacion-de-proyectos-de-software-q182n2w58x1v

Cierre

Existen escenarios de prueba que se deben tomar en cuenta para desarrollar proyectos exitosos y alineados al tiempo en el que se han planeado. En ocasiones, pueden ocurrir imprevistos, cambios o situaciones que no permiten que el proyecto pueda avanzar de acuerdo con lo planificado.

Será de suma importancia reaccionar adecuadamente e investigar desde un punto de vista objetivo, es decir, saber cuál es el origen que ocasiona la situación, de tal manera que se pueda contar con un plan de acción aterrizado.



Universidad Tecmilenio

Software Testing

Administración de recursos humanos

Semana 7

Introducción

El ciclo de vida de desarrollo de software contempla todos los elementos que se deben monitorear con el fin de garantizar la realización de una aplicación de alta calidad.

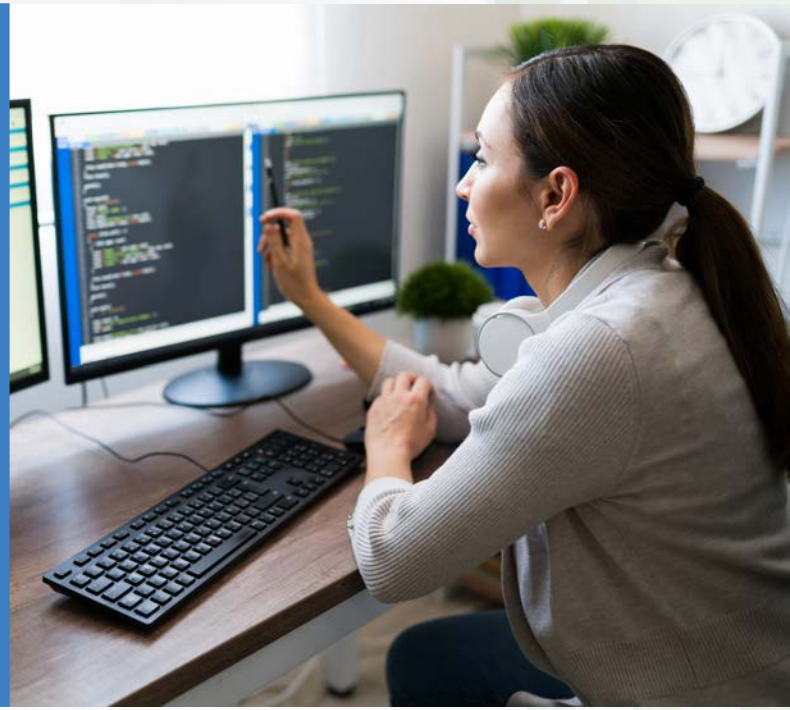
Sin embargo, el *testing* es uno de los pasos que no había tenido tanto impacto como ahora, ya que cada día es más complejo realizar los proyectos en tiempo y forma por las cambiantes tecnologías que se desarrollan y que urgen a las empresas no solo a ser más precisas, sino a asegurar que el tema de la calidad abarque todo el proceso de planeación para evitar fallas que se traducen en tiempos perdidos.

Universidad Tecmilenio

Explicación

Perfiles y especialistas de pruebas

Un *tester* es una persona que realiza pruebas de software o de proyectos similares; está en busca de errores, defectos o cualquier problema que pudiera encontrar el usuario final, construyendo un informe o reporte que servirá al líder del proyecto para conocer cualquier falla y encontrar la manera de mejorar el producto que se entregará finalmente (Hireline, s.f.).



Explicación

Perfiles de realizadores de pruebas

Tester administrativo

- Gestionan las actividades para que se avance en el proyecto.
- Son de alto nivel y se reúnen para seguimiento, avances y obtienen recursos.

Tester técnico

- Piensan en términos de código; crean y usan herramientas.
- Trabajan en equipo para alcanzar las metas más rápido.
- Realizan pruebas funcionales y no funcionales.
- Participan en la planificación de pruebas.

Tester analítico

- Realizan los diagramas matriz y esquemas que se presentan a la gerencia.
- Muestran los resultados de una forma amigable.

Explicación

A continuación, se presentan las funciones de un especialista en realización de pruebas (Taller de informática, 2022):

- Desarrollar y ejecutar pruebas manuales.
- Diseñar y desarrollar planes de testing.
- Verificar las correcciones de los errores encontrados.
- Realizar pruebas de lanzamiento.
- Realizar la documentación de los hallazgos en cada fase del ciclo de testing.
- Analizar y presentar resultados de las pruebas realizadas.

Explicación

Ciclo de vida de las pruebas de software

Explicación

Ahora se muestran las habilidades necesarias para que un tester ejecute correctamente sus actividades (Hireline, s.f.):

Técnicas

- Conocimientos básicos de Linux.
- Conocimientos básicos de SQL.
- Experiencia con herramientas de gestión de pruebas.
- Experiencia con herramientas de automatización.
- Experiencia en herramientas de seguimiento a defectos.

Soft o blandas

- Capacidad analítica.
- Organización y gestión de tiempo.
- Actitud positiva.

Explicación

Antes de decidir por un equipo de pruebas de software interno, es importante considerar la opción de subcontratar este servicio.

Las siguientes son algunas razones por las que esto es recomendable

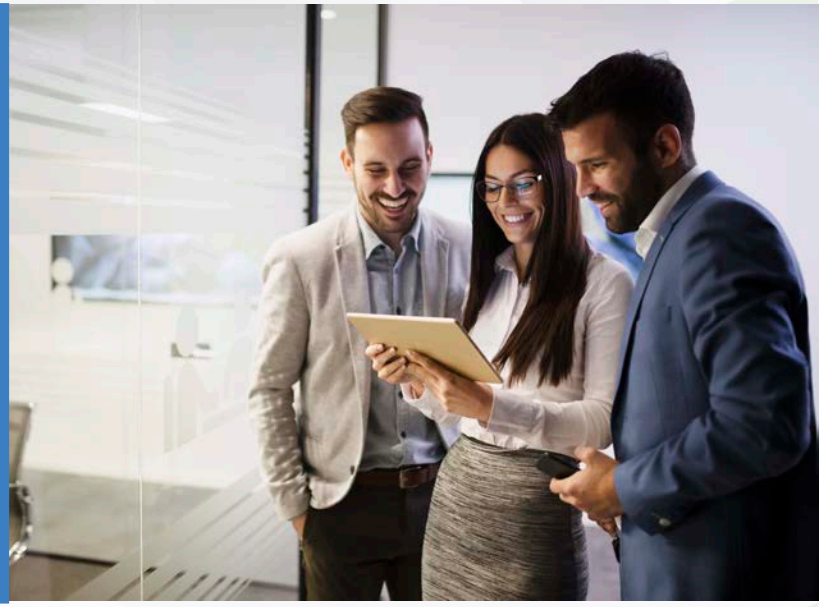
- Reducción de costos.
- Uso de experiencia y tecnología apropiadas.
- Resultados imparciales e independientes.
- Recursos disponibles para otras áreas.
- Mejor calidad del producto final.

El control de calidad es cada vez más necesario para satisfacer a los clientes. Se vuelve crítico poder detectar y reportar errores, haciendo énfasis en la prevención y verificación de soluciones alineadas a los requerimientos planteados (Soler, 2022).

Actividad

¿Qué perfiles de realizadores de pruebas existen y cuáles son sus funciones?

¿Cuáles son los pasos del ciclo de vida de las pruebas del software?



Bibliografía

- Hireline. (s.f.). *Perfil de Tester / Software Tester.* Recuperado de https://hireline.io/mx/enciclopedia-de-perfiles-ti/perfil-de-tester-software-tester
- Soler, J. (2022). *Principales Tendencias de Testing y Calidad de Software para 2022*. Recuperado de https://cl.abstracta.us/blog/tendencias-testing-calidad-software/
- Taller de informática. (2022). *TESTER DE SOFTWARE.* Recuperado de https://www.tallerdeinformatica.edu.uy/Blog/detalle/tester-de-software

Universidad Tecmilenio

Cierre

Actualmente se presentan cambios exponenciales en las tecnologías de la información que impactan a todos los negocios y, a pesar de que no se encuentra todo digitalizado, la tendencia indica que se llegará a ese punto.

Por lo que es importante mantenerse al día con las tendencias emergentes del testing, ya que esto marcará la pauta a los profesionales de la calidad de software para crear nuevas líneas de acción y herramientas que les permitirán ser más competitivos e implementar productos de software de alta calidad, seguros y confiables.