ISTQB

0.4 resultados de negocio

Esta sección enumera los 14 resultados de negocio esperados de una persona que ha logrado la

certificación de nivel básico.

Un probador certificado de nivel básico puede...

FL-BO1 Comprender qué son las pruebas y por qué son beneficiosas

FL-BO2 Comprender los conceptos fundamentales de las pruebas de software

FL-BO3 Identificar el enfoque de la prueba y las actividades a implementar dependiendo del

contexto de la prueba

FL-BO4 Evaluar y mejorar la calidad de la documentación

FL-BO5 Aumentar la eficacia y eficiencia de las pruebas

FL-BO6 Alinear el proceso de prueba con el ciclo de vida de desarrollo de software

FL-BO7 Comprender los principios de gestión de pruebas

FL-BO8 Escribir y comunicar informes de defectos claros y comprensibles

FL-BO9 Comprender los factores que influyen en las prioridades y esfuerzos relacionados con las

pruebas

FL-BO10 Trabajar como parte de un equipo multifuncional

FL-BO11 Conocer los riesgos y beneficios relacionados con la automatización de pruebas

FL-BO12 Identificar las habilidades esenciales requeridas para las pruebas

FL-BO13 Comprender el impacto del riesgo en las pruebas

FL-BO14 Informar eficazmente sobre el progreso y la calidad de la prueba

Fundamentos de la prueba

1.1 ¿Qué es probar?

1.1.1 EL software hace parte de la vida diaria, todas las personas interactúan diariamente con el software, si no funciona como se esperaba puede ocasionar perdida de dinero, tiempo, reputación y seguridad en lsa personas.
Las pruebas de software reducen el riesgo de falla del software en funcionamiento.
Sirven para descubrir defects y evaluar la calidad de los artefactos de software (Objetos de prueba)
Las pruebas implican verificación, es decir verificar si el sistema cumple con los requisitos, e implica validad, que significa comprobar si el sistema cumple con las necesidades del usuario y partes interesadas.
Deben ser planificadas, gestionadas, estimadas, monitoreadas y controladas.
Las pruebas son una actividad intelectual, uso de habilidades analíticas y pensamiento sistémico
Los objetivos de una prueba pueden ser:

- Evaluar productos, requisitos, historias de usuario diseños y código
- Desencadenar fallas y encontrar defectos
- Garantizar la cobertura requerida de un objeto de prueba
- Reducir el nivel de riesgo de una calidad
- Verificar si se cumplen requisitos
- Verificar si se cumplen requisitos contractuales, legales y reglas
- Informar a las partes interesadas para tomar decisiones
- Generar confianza en el producto
- Validar si el producto esta completo y funciona según lo definido

1.1.2 Las pruebas y la depuración son actividades separadas, las pruebas desencadenan fallas o encuentran directamente defectos, la depuración se encarga de encontrar las causas de la falla, analizar y eliminar, un proceso de depuración típico es: Reproducir falla, diagnosticar, corregir.

1.2 ¿Por qué es necesario probar?--> Acercar el proyecto al éxito lo pueden hacer todos

1.2.1 La prueba proporciona una manera de evaluar la calidad de un objeto de prueba en diferentes etapas del desarrollo del software
Se debe tener en uenta las necesidades de los usuarios, se puede involucrar

un conjunto de usuarios como parte del desarrollo, pero generalmente no es posible por costos

1.2.2 Pruebas y QA no son lo mismo

Las pruebas son una forma de control de calidad.

QA es un enfoque preventivo orienado a procesos, centrado en implementación y mejora de procesos, se basa en que si sigue un buen proceso se obtiene un buen producto

Los resultados de la prueba son utilizados por QA y QC. En QC se utilizan para corregir defectos, mientras que en QA proporcionan retroalimentación sobre qué tan bien se están desempeñando los procesos de desarrollo y prueba.

1.2.3 Los seres humanos cometen errores, que producen defectos (Bugs), que a su vez pueden resultar en fallas. Algunos defectos siempre resultaran en una falla si se ejecutan, mientras que otros solo resultarán en una falla en circunstancias específicas, algunas pueden no llevar nunca a una falla.

Las fallas también puede ser causados por problemas externos, no necesariamente por errores y defectos.

Una causa raíz es una razón fundamental para la ocurrencia de un problema (por ejemplo, una situación que conduce a un error).

1.3 Principios de las pruebas

1.3.1 Las pruebas muestran la presencia, no la ausencia de defectos

1.3.2 Las pruebas exhaustivas son imposibles

1.3.3 Las pruebas tempranas ahorran tiempo y dinero

1.3.4 Los defectos se agrupan

1.3.5 Las pruebas se desgastan

1.3.6 Las pruebas dependen del contexto

1.3.7 Falcia de la ausencia de defectos.

1.4 Actividades de la prueba, testware y roles de la prueba

1.4.1 Actividades de la prueba

1.4.1.1 Planificación

1.4.1.2 El monitoreo y control de prueba

1.4.1.3 El análisis de prueba

1.4.1.4 El diseño de la prueba

1.4.1.5 La implementación de prueba

1.4.1.6 La ejecución de pruebas

1.4.1.7 Las actividades de la finalización de pruebas

1.4.2 El proceso de la prueba en contexto

1.4.3 Testware

1.4.4 Trazabilidad entre la base de prueba y el test ware

1.4.5 Roles en las pruebas

1.5 Habilidades esenciales y buenas prácticas en las pruebas

1.5.1 Habilidades genericas

- Conocimiento en pruebas (para aumentar la efectividad de las pruebas, por ejemplo, mediante el
- uso de técnicas de prueba)
- Rigurosidad, cuidado, curiosidad, atención a los detalles, ser metódico (para identificar
- defectos, especialmente los que son difíciles de encontrar)
- Buenas habilidades de comunicación, escucha activa, ser un jugador de equipo (para
- interactuar de manera efectiva con todas las partes interesadas, para transmitir información a
- los demás, para ser entendido, y para informar y discutir defectos)
- Pensamiento analítico, pensamiento crítico, creatividad (para aumentar la efectividad de las pruebas)
- Conocimientos técnicos (para aumentar la eficiencia de las pruebas, por ejemplo, mediante el uso de herramientas de prueba apropiadas)
- Conocimiento del dominio (para poder comprender y comunicarse con los usuarios finales/representantes comerciales)

1.5.2 Enfoque de equipo completo

1.5.3 Independecia de las pruebas

2 Pruebas a lo laro del ciclo de vida del software

2.1 Pruebas en el contexto de un ciclo de vida de desarrollo de software

2.1.1 Las pruebas se adaptan al SDLC, la elección de este impacta en:

- Alcance, momento o secuencia de las actividades de prueba (por ejemplo, niveles de prueba y tipos de prueba)
- Nivel de detalle de la documentación de la prueba
- Elección de técnicas de prueba y enfoque de prueba
- Alcane de la automatización de pruebas
- Rol y responsabilidades de un probador

2.2 Tipos de pruebas

2.2.1 Las pruebas funcionales, evalúan las funciones que un componente debe realizar

2.2.2 Las pruebas no funcionales evalúan atributos distintos de las características funcionales de un componente o sistema. Son las pruebas de que tan bie se comporta el sistema

- • Eficiencia del rendimiento
- • Compatibilidad
- • Usabilidad
- • Fiabilidad
- • Seguridad
- • Mantenibilidad
- • Portabilidad

2.2.3 La prueba de caja negra, se basa en especificaciones y derivan pruebas de de documentación que es externa al objeto de prueba. El objetivo principal de las pruebas de caja negra es comprobar el comportamiento del sistema contra sus especificaciones.

2.2.4 Las pruebas de caja blanca, se basan en la estructura y derivan pruebas de la implementación del sistema o la estructura interna

2.2.5 Los cuatro tipos de prueba mencionados anteriormente se pueden aplicar a todos los niveles de prueba, aunque el enfoque será diferente en cada nivel. Se pueden usar diferentes técnicas de prueba para derivar las condiciones de prueba y los casos de prueba para todos los tipos de prueba mencionados.

2.2.6 Pruebas de confirmación, confirman que un defecto original se ha leído con éxito

Los cuatro tipos de prueba mencionados anteriormente se pueden aplicar a todos los niveles de prueba, aunque el enfoque será diferente en cada nivel. Se pueden usar diferentes técnicas de prueba para derivar las condiciones de prueba y los casos de prueba para todos los tipos de prueba mencionados.

2.2.7 Las pruebas de regresión: n confirman que no se han causado consecuencias adversas por un cambio, incluida una corrección que ya ha sido probada

Niveles de prueba

- Las pruebas de componentes
- Las pruebas de integración de componentes
- Las pruebas del sistema
- Las pruebas de integración de sistemas
- Las pruebas de aceptación

CAPITULO 3

Pruebas Estáticas:

- A diferencia de las pruebas dinámicas, las pruebas estáticas no requieren la ejecución del software bajo prueba.
- Se evalúan productos de trabajo, como código, especificaciones y arquitecturas, mediante revisión manual o herramientas como análisis estático.
- Objetivos: mejorar calidad, detectar defectos y evaluar características como legibilidad, completitud y consistencia.
- Se aplican para verificación y validación.

Colaboración durante el Mapeo de Ejemplos:

- Probadores, representantes comerciales y desarrolladores trabajan juntos para garantizar que historias de usuario y productos de trabajo cumplan con criterios definidos.
- Técnicas de revisión se aplican para asegurar completitud y comprensión de historias de usuario, utilizando preguntas para mejorarlas.

Análisis Estático:

- Identifica problemas antes de pruebas dinámicas y a menudo requiere menos esfuerzo.
- Se incorpora en marcos de trabajo de Integración Continua.
- Utilizado para detectar defectos específicos, evaluar mantenibilidad y seguridad.
- Puede incluir herramientas como correctores ortográficos y de legibilidad.

Productos Evaluables por Pruebas Estáticas:

- Cualquier producto de trabajo que se pueda leer y comprender puede ser examinado.
- Revisiones aplicadas a documentos, código, planes de prueba, entre otros.

Valor de las Pruebas Estáticas:

- Detectan defectos en fases tempranas del SDLC, cumpliendo con el principio de pruebas tempranas.
- Identifican defectos no detectables mediante pruebas dinámicas.
- Evalúan calidad y generan confianza en productos de trabajo.
- Se pueden realizar temprano en el SDLC, promoviendo entendimiento compartido y mejor comunicación.

Diferencias entre Pruebas Estáticas y Dinámicas:

- Complementarias pero con diferencias.
- Pruebas estáticas encuentran defectos directamente, mientras que las dinámicas producen fallas que requieren análisis posterior.
- Pruebas estáticas pueden aplicarse a productos no ejecutables, mientras que las dinámicas solo a ejecutables.
- Detectan más fácilmente defectos en rutas de código raramente ejecutadas.

Beneficios de Retroalimentación y Revisiones:

- Retroalimentación temprana y frecuente previene malentendidos y problemas de calidad.
- Participación de partes interesadas es crucial para evitar fracasos en cumplir con la visión original.
- Retroalimentación en todo el SDLC mejora comprensión, comunicación y evita echar la culpa.

Actividades del Proceso de Revisión:

- Planificación, inicio, revisión individual, comunicación y análisis, reparación e informes son fases clave.
- Roles incluyen director, autor, moderador, escriba, revisor y líder de revisión.
- Tipos de revisiones van desde informales hasta formales, seleccionados según necesidades y contexto.

Factores de Éxito para las Revisiones:

- Definir objetivos claros y criterios medibles.
- Elegir tipo de revisión apropiado.

- Realizar revisiones en trozos pequeños.
- Proporcionar retroalimentación y tiempo suficiente para preparación.
- Apoyo de la gerencia, cultura organizacional de revisiones y formación adecuada son clave.

CAPITULO 4

El análisis y diseño de pruebas se apoyan en técnicas específicas que ayudan a desarrollar un conjunto eficiente de casos de prueba. Estas técnicas también facilitan la definición de condiciones de prueba y la identificación de datos relevantes. En el Programa de Estudio, las técnicas de prueba se clasifican en caja negra, caja blanca y basadas en la experiencia.

- **Caja Negra:** Se enfoca en el comportamiento especificado del objeto de prueba sin considerar su estructura interna. Los casos de prueba son independientes de la implementación del software, lo que los hace útiles incluso si la implementación cambia.
- **Caja Blanca:** Analiza la estructura interna y el procesamiento del objeto de prueba. Los casos de prueba dependen del diseño del software y solo se pueden crear después del diseño o la implementación.
- **Basadas en la Experiencia:** Utilizan el conocimiento y la experiencia de los probadores para diseñar e implementar casos de prueba. Son complementarias a las técnicas de caja negra y caja blanca, detectando defectos que podrían pasar desapercibidos.

Se destacan algunas técnicas específicas de caja negra, como:

- **Partición de Equivalencia (EP):** Divide los datos en particiones según la expectativa de que elementos en una partición deben ser procesados de la misma manera. Se busca ejercer todas las particiones identificadas para lograr una cobertura del 100%.
- **Análisis de Valores Límite (BVA):** Se basa en ejercitar los límites de particiones ordenadas, centrándose en valores límite donde los desarrolladores pueden cometer errores. Existen versiones de BVA de 2 y 3 valores, cada una con diferentes requisitos de cobertura para identificar defectos.

4.2.3. Pruebas de Tabla de Decisión:

- Las tablas de decisión prueban la implementación de requisitos del sistema.
- Definen condiciones y acciones en filas y reglas de decisión en columnas.
- Condiciones y acciones se representan como valores booleanos o múltiples valores.
- Se simplifican o minimizan las tablas para reducir el número de reglas que deben ser ejercidas.
- Los elementos de cobertura son las columnas que contienen combinaciones factibles de condiciones.
- Fortaleza: Enfoque sistemático para identificar todas las combinaciones de condiciones.

4.2.4. Pruebas de Transición de Estado:

- Los diagramas de transición de estado modelan el comportamiento del sistema.
- Las tablas de estado son equivalentes a los diagramas de transición.
- Criterios de cobertura incluyen estados, transiciones válidas y todas las transiciones.
- Cobertura de transiciones válidas es el criterio más utilizado.
- Pruebas de API, neuronales y otras técnicas no relacionadas no se abordan en este programa.

4.3. Técnicas de Caja Blanca:

- Incluye pruebas de sentencias y cobertura de sentencias, pruebas de ramas y cobertura de ramas.
- Proporciona una visión sistemática de la implementación del software.
- La cobertura de caja blanca ofrece medidas objetivas de la cobertura de código.
- Técnicas más rigurosas se utilizan en entornos críticos de seguridad.

4.4. Técnicas Basadas en la Experiencia:

- Predicción de errores anticipa defectos basándose en la experiencia.

- Pruebas exploratorias se diseñan, ejecutan y evalúan simultáneamente para explorar el objeto de prueba.
- Pruebas basadas en listas de comprobación utilizan listas para cubrir condiciones de prueba.
- Las listas deben actualizarse regularmente en función del análisis de defectos.

4.5. Enfoques de Prueba Basados en la Colaboración:

- Redacción colaborativa de historias de usuario utiliza técnicas como lluvia de ideas y mapeo mental.
- Criterios de aceptación definen condiciones para la aceptación de la implementación.
- Desarrollo guiado por pruebas de aceptación (ATDD) crea casos de prueba antes de la implementación.
- Los casos de prueba deben cubrir características de calidad no funcionales y no exceder el alcance de la historia.

CAPITULO 5

5.1.1. Propósito y Contenido de un Plan de Prueba:

- Un plan de prueba detalla los objetivos, recursos y procesos para un proyecto de prueba.
- Documenta medios y cronograma para lograr objetivos.
- Garantiza que las actividades de prueba cumplan con criterios establecidos.
- Sirve como medio de comunicación con el equipo y otras partes interesadas.
- Asegura la adhesión a la política y estrategia de pruebas existentes.

Contenido Típico de un Plan de Prueba:

- Contexto de las pruebas (alcance, objetivos, limitaciones, base de la prueba).
- Supuestos y limitaciones del proyecto de prueba.
- Partes interesadas (roles, responsabilidades, relevancia para pruebas, contratación, necesidades de capacitación).
- Comunicación (formularios, frecuencia de comunicación, plantillas de documentación).

- Registro de riesgos (riesgos de producto, riesgos de proyecto).
- Enfoque de prueba (niveles, tipos, técnicas, resultados, criterios de entrada/salida, independencia, métricas, requisitos de datos y entorno de prueba, desviaciones de política y estrategia).

Presupuesto y Cronograma:

- Detalles sobre recursos financieros y tiempo planificado.

5.1.2. Contribución del Probador a la Planificación de la Iteración y Entrega:

- En SDLC iterativos, se aborda la planificación de entregas y planificación de iteraciones.
- La planificación de entregas anticipa productos y define/refina la pila de productos.
- Involucra a probadores en redacción de historias de usuario, análisis de riesgos, estimación de esfuerzo de prueba.
- Planificación de iteraciones se enfoca en el final de una iteración, involucrando probadores en análisis detallado de riesgos, estimación de esfuerzo, y la identificación de aspectos del objeto de prueba.

5.1.3. Criterios de Entrada y Criterios de Salida:

- Criterios de entrada establecen condiciones previas para actividades.
- Criterios de salida definen logros para declarar una actividad completada.
- Varían según objetivos de prueba y niveles de prueba.

5.1.4. Técnicas de Estimación:

- La estimación del esfuerzo de prueba implica predecir la cantidad de trabajo relacionado con la prueba.
- Cuatro técnicas: Estimación basada en proporciones, extrapolación, Delphi de banda ancha, Estimación de tres puntos.
- Importancia de aclarar que la estimación se basa en supuestos y está sujeta a errores.
- Enfoque de descomposición para tareas grandes.

CAPITULO 6

Objetivos de Aprendizaje para el Capítulo 6:

6.1 Soporte de Herramientas para las Pruebas

FL-6.1.1 (K2) Explicar cómo los diferentes tipos de herramientas de prueba soportan las pruebas

6.2 Beneficios y Riesgos de la Automatización de Pruebas

FL-6.2.1 (K1) Recordar los beneficios y riesgos de la automatización de pruebas