**ANALISIS Y DESARROLLO DE SISTEMAS DE INFORMACION**

**TESTING**

**EDUBIN TORRES**

**GAES 7**

**DANIEL VARGAS**

**NICOLAS VEGA**

**ACTIVIDAD 1 – PRUEBAS DE SOFTWARE**

**PRESENTACIÓN**

Las pruebas de software (en inglés software testing) son las investigaciones empíricas y técnicas cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a la parte interesada o stakeholder. Es una actividad más en el proceso de control de calidad.

**Pruebas de caja Negra:**

Las Pruebas de Caja Negra, es una técnica de pruebas de software en la cual la funcionalidad se verifica sin tomar en cuenta la estructura interna de código, detalles de implementación o escenarios de ejecución internos en el software. En las pruebas de caja negra, nos enfocamos solamente en las entradas y salidas del sistema, sin preocuparnos en tener conocimiento de la estructura interna del programa de software. Para obtener el detalle de cuáles deben ser esas entradas y salidas, nos basamos en los requerimientos de software y especificaciones funcionales.

**Pruebas de caja Blanca:**

Las técnicas de testing de caja blanca se realizan cuando el tester accede al código fuente de la aplicación y en consecuencia a los diferentes algoritmos y estructuras de datos utilizadas. La implementación de este tipo de pruebas requiere habilidades de programación, un conocimiento del framework de desarrollo y un cierto conocimiento funcional que permita conocer qué misión tienen determinadas clases y métodos.

Entre las técnicas de testing de caja blanca más conocidas tenemos la cobertura que consiste básicamente en la verificación de que todos los caminos lógicos de la aplicación son alcanzables teóricamente en función de los diferentes valores de entradas de los parámetros. Este tipo de pruebas, se automatizan con la ejecución de pruebas unitarias.

Otra técnica bastante conocida es la Mutation Testing que se suele utilizar para verificar la bondad de los métodos de testing utilizados. Se basa principalmente en realizar ligeras modificaciones en el programa que darían lugar a un comportamiento anómalo del mismo (resultados distintos) y verificar si la estrategia de testing utilizada es capaz de detectar estos cambios. Ejemplos de estos pequeños cambios lo podemos tener modificando el operador en las guardas de sentencias selectivas o iterativas, eliminando sentencias, etc...

***ACTIVIDADES DE REFLEXION INICIAL:***

**Actividad aprendizaje autónomo 1:**

El proceso de desarrollo de software se encuentra enmarcado en procesos, actividades y tareas involucradas en el desarrollo, puesta en marcha y mantenimiento del producto de software. Este marco de referencia se conoce como el ciclo de vida, una vez finalizadas estas etapas de desarrollo se debe proceder con la parte de instalación o implementación en un servidor plenamente configurado.

Antes de empezar con la fase de implementación debe desarrollarse la fase de pruebas en un ambiente de desarrollo, para posteriormente pasarlo a un ambiente de producción.

Vamos a suponer que ustedes desarrollan un software para llevar el control clínico de un Hospital de cada uno de los pacientes y se pagó una suma considerable de dinero por el desarrollo de esa aplicación. Pasados 3 días después de la fase de implementación y puesta en marcha los llaman porque el operador del sistema (usuario final), se queja de que la información que ingresa en algunos campos no coincide con los informes y consultas que se realizan al final del día.

**Responda las siguientes preguntas:**

* Qué determinación toma usted frente a este hecho.

El fallo de un software en etapa de implementación es un error muy grave por lo que se debe buscar la causa de dicho fallo y a partir de esto buscar la mejor solución para resolver el problema lo más antes posible.

* Analice y regrese su mente a las etapas ejecutadas por usted durante todo el desarrollo del proyecto y determine que pudo haber sucedido.

En el desarrollo software ocasionalmente ocurren errores que suelen pasar desapercibidos, esto seria un causante del fallo del software, también puede que las pruebas de caja negra iniciales no se hayan ejecutado correctamente provocando que no se detecten errores en la funcionalidad y por ende genere el fallo del software.

* Cuál cree que es el procedimiento técnico de su parte a seguir.

Primeramente, se tendría que tomar una rápida acción para determinar qué es lo que está fallando en el desarrollo y la implementación del software, seguidamente corregir los errores que pueda haber y someter el software a una prueba de caja negra para verificar que todo esté funcionando adecuadamente.

* Si usted realiza una validación del caso propuesto por el usuario final y se da cuenta que el usuario tiene toda la razón, describa sus propias conclusiones.

En conclusión, las pruebas de caja negra y de caja blanca son dos procesos fundamentales a la hora de comprobar el correcto funcionamiento de un software ya que evitan este tipo de problemas con el usuario final.

* Realice un documento Word con su respectiva portada y describa responda cada una de las situaciones anteriores.

Guarde el documento y socialícelo con los demás grupos de trabajo, y de esta manera realizar un debate frente al caso presentado y las posturas de cada uno de los grupos para dar una solución óptima al Hospital.

**Actividad de aprendizaje 2:**

Una vez resuelta la actividad 1 se debe proceder a la realización de la actividad 2, con el objetivo de dar cumplimiento al resultado de aprendizaje esperado. Por lo anterior y tomando como base la presentación de la presente guía es necesario dar inicio al desarrollo de las respectivas pruebas de software, y diligenciar el formato respectivo adjunto en el material de apoyo, tomando como alternativa pruebas de Caja Negra, para tres módulos que signifiquen gran importancia para usted para el correcto funcionamiento de la app de sus proyectos.

**ACTIVIDAD 2 - TESTING**

**Actividades de contextualización**

* Vamos a realizar una construcción gráfica en el tablero por GAES acerca de los contextos y ambientes de despliegue de los sistemas de información.

Para el desarrollo de estos ejemplos y ejercicios se deben elegir herramientas de gestión con todos los aprendices, se sugiere utilizar aplicativos Open Source.

* Consultar de forma individual acerca de que son las pruebas funcionales, pruebas unitarias, caja negra y caja blanca. Prepare preguntas y muestre ejemplos de las mismas. Luego con ayuda del instructor y los demás aprendices se resolverán las inquietudes en mesa redonda.

**Actividades de apropiación**

* Para dar inicio al desarrollo de las actividades contenidas en esta guía es necesario comprender los siguientes conceptos de forma preliminar para poder abordar los elementos en el contexto de pruebas de software.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| BPM | PMP | CMMI | ISTQB | Técnicas |
| Pruebas de usabilidad | Pruebas unitarias | Pruebas de stress | Pruebas de sistema | Pruebas de aceptación |

Para la consulta de los términos mencionados anteriormente, es preciso consultar las siguientes referencias bibliografías:

* Sommerville, Ian. Ingeniería del Software. Adison Wesley Madrid. 2005. (Cap. 22 y 23)
* Pressman Roger S. Ingeniería del Software. Un enfoque práctico. Mc Graw Hill. 7Ed. 2006.

PRUEBA DE USABILIDAD

Son un método para evaluar la experiencia del usuario en base a el producto o sitio web, la usabilidad la prueban con un grupo de usuarios o clientes y los evaluadores saben si los usuarios reales pueden usar el servicio de una manera fácil e intuitiva.

Beneficios

* **Brindan un examen imparcial, preciso y directo de la experiencia del usuario del producto o sitio web: cuando se prueba la usabilidad se muestran los usuarios, sus comentarios y la mayoría de dudas**
* **Son convenientes, pues requieren pocas adecuaciones: al llevarse** a cabo el estudio, todo lo que se encontrara es una habitación tranquila y se traerá un equipo de grabación portátil
* **Pueden decirte qué hacen los usuarios: Con tu producto o en tu sitio web** y la razón por la cual se realizan esas acciones
* **Te permiten abordar los problemas del producto o sitio web: Esto es válido para que antes** de gastar recursos en un diseño deficiente
* **Posibilitan un diseño intuitivo** Que, a su vez, aumenta el uso, mejora los resultados que obtendrá el usuario y fomentará una mayor demanda del sitio web o producto

**Tipos de usabilidad:**

1. Pruebas de usabilidad de guerrilla o de pasillo.
2. Pruebas de usabilidad remota no moderada.
3. Pruebas de usabilidad moderadas.

**PRUEBAS UNITARIAS**

**Se basan en la verificación del comportamiento de las unidades mas pequeñas; deben ejecutarse de forma aislada ya que tienen que ser de fácil y rápida ejecución.**

**Propósito:**

**Validar las unidades de trabajo si su comportamiento es el esperado o previsto según su diseño.**

**Beneficios:**

* **Ahorro de tiempo**
* **Disminución de errores**
* **No habrá ruptura del código existente**
* **Reducción de carga de personal por menor número de errores**

**PRUEBAS DE STRESS**

**Proceso por el cual se elegirán las actividades a probar en el sitio web para codificarlas y ejecutarlas en un tiempo estimado desde una ubicación remota.**

**Proceso:**

1. **Actividades claves para customer journey**

* **Tiempo de carga en landing page**
* **Tiempo de respuesta en búsqueda de artículos**
* **Tiempo de carga en agregar un producto al carrito.**
* **Tiempo de carga en borrar un producto**
* **Checkout antes de interactuar con la pasarela de pago**

1. **Diseñar script personalizado con las actividades claves previamente identificadas**
2. **Ejecutar pruebas de cada actividad con un número incremental de usuarios en horarios de bajo o nulo tráfico.**
3. **Se realiza reporte.**

**PRUEBAS DE SISTEMA**

**Su objetivo es ejercitar el sistema comprobando la integración de sistemas de información, verificando el funcionamiento correcto de las interfases entre los subsistemas y el resto de sistemas de información que los comunican.**

**Motivos**

**Se realiza porque permite probar el sistema en su conjunto y verificar con otros sistemas cómo reaccionan las especificaciones funcionales y técnicas, con esto se da una visión de su comportamiento en el entorno de producción.**

**PRUEBAS DE ACEPTACION**

**Son realizadas para determinar la confianza de un sistema dependiendo el grado de requerimientos que se maneje en este; se debe realizar en testing de aceptación el cual se realiza con las pruebas de caja negra para así verificar su entrega definitiva.**

**Requisitos:**

* [Requerimientos del usuario](http://www.pmoinformatica.com/2018/05/que-es-requerimiento-funcional.html).
* Requerimientos de sistema.
* [Casos de uso](http://www.pmoinformatica.com/2014/07/plantilla-de-casos-de-uso.html).
* Procesos de negocio.
* Reportes de análisis de riesgo.

Una vez consultadas realizar un documento con normas APA donde describa cada uno de los conceptos, en este documento se debe hablar de su historia, características y gráficos que ejemplifiquen su uso. El documento debe ser enviado a través del link correspondiente en el drive carpeta evidencias, su instructor le dará la retroalimentación correspondiente por este medio, recuerde adjuntar la actividad a su portafolio.

* Es necesario profundizar en la relación existente entre el desarrollo, actividades de pruebas y productos de trabajo en el ciclo de vida del software. Para ello de manera individual el aprendiz deberá realizar un mapa mental sobre las pruebas que se realizan durante el ciclo de vida de software (Modelo en V). Esta evidencia debe ser enviada a través del link correspondiente en el drive carpeta evidencias, el instructor le dará la retroalimentación que respecta por este medio y adjunte la actividad a su portafolio.

* Vamos a hablar sobre las técnicas de diseño de pruebas, para esto es necesario que de manera individual realice una investigación con normas APA sobre los siguientes temas:
* El proceso de desarrollo de pruebas

Las pruebas de software (en [inglés](https://es.wikipedia.org/wiki/Idioma_ingl%C3%A9s) *software testing*) son las investigaciones empíricas y técnicas cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a la [parte interesada o *stakeholder*](https://es.wikipedia.org/wiki/Parte_interesada_(empresas)). Es una actividad más en el proceso de [control de calidad](https://es.wikipedia.org/wiki/Control_de_calidad).

Las pruebas son básicamente un conjunto de actividades dentro del desarrollo de [software](https://es.wikipedia.org/wiki/Software). Dependiendo del tipo de pruebas, estas actividades podrán ser implementadas en cualquier momento de dicho proceso de desarrollo. Existen distintos modelos de desarrollo de software, así como modelos de pruebas. A cada uno corresponde un nivel distinto de involucramiento en las actividades de desarrollo.

* Diseño de casos de prueba

El diseño de casos de prueba, tiene un único objetivo: tener la mayor probabilidad de encontrar el mayor número de errores con la mínima cantidad de esfuerzo y tiempo posible.

* Categorías de técnicas de diseño de pruebas

El objetivo de una técnica de diseño de pruebas es identificar condiciones de prueba, casos de prueba y datos de prueba.  
Tradicionalmente se distingue entre técnicas de pruebas de caja negra y de caja blanca. Las técnicas de diseño de prueba de caja negra (también conocidas como técnicas basadas en la especificación) son una forma de derivar y seleccionar condiciones de prueba, casos de prueba o datos de prueba en base a un análisis de la documentación básica de la prueba.  
Esto incluye pruebas tanto funcionales como no funcionales. Las pruebas de caja negra, por definición, no utilizan ninguna información sobre la estructura interna del componente o sistema a aprobar. Las técnicas de diseño de prueba de caja blanca (también conocida como técnicas basadas en la estructura o estructurales) se basan en el análisis de la estructura del componente o sistema. Las pruebas de caja negra o de caja blanca también pueden basarse en la experiencia de los desarrolladores, probadores y usuarios para determinar el objeto de las pruebas.

* Técnicas basadas en la especificación o técnicas de caja negra.

Las pruebas de caja negra, también denominadas por el [ISTQB](http://www.pmoinformatica.com/2012/07/5-preguntas-y-respuestas-sobre-istqb.html) como técnicas basadas en especificación, son una forma de derivar y seleccionar condiciones, datos y casos de prueba a partir de la [documentación de requerimientos del sistema](http://www.pmoinformatica.com/2018/04/documento-de-requerimientos-de-software_37.html).  
  
Las pruebas de caja negra no utilizan ninguna información interna de los componentes de software o sistemas que se van a probar, sino que consideran el comportamiento del software desde el punto de vista de un observador externo (.Como los usuarios del sistema).

* Técnicas basadas en la estructura o técnicas de caja blanca.

Las pruebas basadas en la estructura o de caja blanca se basan en una estructura identificada del software o del sistema, según se demuestra en los siguientes ejemplos:  
Nivel de componente: la estructura de un componente de software, es decir, sentencias, decisiones, ramas o incluso caminos distintos.

Nivel de integración: la estructura puede ser un árbol de llamadas (un diagrama en el los módulos llaman a otros módulos).

Nivel de sistema: la estructura puede ser una estructura de menús, procesos de negocio o una estructura de página web.

En este apartado, se tratan tres técnicas de diseño de pruebas estructurales relacionadas con códigos para la cobertura de código, en base a sentencias, ramas y decisiones. Para las pruebas de decisiones, puede utilizarse un diagrama de flujo para visualizar las alternativas para cada decisión.

* Técnicas basadas en la experiencia.

Una **técnica basada en la experiencia** muy usada es la predicción de error. En general los probadores anticipan los defectos en base a su **experiencia**. Un enfoque estructurado de la **técnica** de predicción de error consiste en enumerar una lista de posibles defectos y diseñar pruebas para atacar dichos defectos.

* Selección de las técnicas de pruebas.

Prueba unitaria. Empecemos con el primero y más pequeño de todos – prueba unitaria.

**Pruebas** de integración. La prueba de integración es una extensión lógica de las **pruebas** unitarias.

**Pruebas** funcionales.

**Pruebas** de rendimiento.

Este documento debe ser enviado a través del link correspondiente en el drive carpeta evidencias, además de ser adjuntado a su portafolio; una vez finalizado este paso en la siguiente sesión se realizará una mesa redonda donde se discutirán cada uno de los temas mencionados anteriormente.

**Actividades de transferencia de conocimiento**

Ahora que conocemos los conceptos sobre el ciclo de pruebas de software vamos a empezar a trabajar sobre su proyecto formativo, para ello en GAES van a diligenciar el “*Formato Casos de prueba*” que encontrará en la carpeta Formatos, dentro del material de aprendizaje en la carpeta de Testing, de acuerdo a los casos de uso de su proyecto.

En caso de haber dudas en diseño de los casos de pruebas recuerde que se puede apoyar en su instructor o en el documento “*Casos de prueba*” que encontrará en el material de aprendizaje de la carpeta de Testing.

Una vez diligenciado el formato debe ser enviado a través del link correspondiente en el drive carpeta evidencias, su instructor validará el correcto desarrollo de la actividad mediante una lista de chequeo, y se encargará de retroalimentar.