**Clase 2 – Primera parte**

Bienvenidos y bienvenidas a la primera parte de la segunda clase del curso de testing funcional y accesibilidad web.

**¿Qué veremos en esta clase?**

Esta clase la dedicaremos a comenzar a sumergirnos en el mundo del testing, veremos: roles en el desarrollo de software, concepto del testing, y sus fundamentos, atravesando los conceptos que vimos la clase pasada y profundizándolos.

El material lo encontrarán en Word (“doc”) para facilitar la navegación como también la edición, subrayado o resaltado. Si lo prefieren podrán descargar la clase también en formato “pdf”.

1. **Roles en el desarrollo de software**

El desarrollo de software no es una actividad simple y requiere de la participación de diferentes actores, es por ello, que necesitamos definir quienes son los participantes y como se vinculan entre sí.

Un equipo de trabajo está formado por 2 o más personas que trabajan con un objetivo en común. Cada persona tendrá un rol, es decir, actividades asociadas a la tarea a desarrollar. En equipos grandes es habitual tener funciones bien diferenciadas.

Las definiciones de roles dentro de un proyecto posibilitan la estructuración de este y se convertirán en uno de los pilares para que este camino sea exitoso.

**Entonces, ¿cuáles son los roles dentro de un equipo?**

* **Project manager / Líder de proyecto:** se ocupa de coordinar el equipo y busca asegurar que los demás participantes cumplan con las tareas asignadas. Debe tener una visión integral del proyecto, saber cómo se vinculas los diferentes módulos y administrar los tiempos para evitar retrasos.
* **Analista funcional:** se ocupa de entrevistar y explicitar las necesidades del cliente en lo que llamaremos los *requerimientos* para poder definir lo que se espera del sistema.
* **Diseñador:** a partir del trabajo del analista funcional genera el diseño del sistema y *prototipo.*
* **Desarrollador:** toma los *requerimientos* y el *prototipo* y lo convierte en un programa funcional en un lenguaje de programación específico.
* **Tester / probador:** se ocupa de asegurar la calidad del producto ya que verifica el funcionamiento y lo contrasta con los requerimientos del cliente y el prototipo.

Adentrándonos en el corazón de este curso, tomaremos **uno de los roles** del equipo de trabajo e iremos conociendo sus tareas y las habilidades necesarias.

**¿Cuáles son las tareas que realiza un tester?**

* Reunirse con usuarios del sistema para comprender el alcance de los proyectos.
* Trabajar con desarrolladores de software y equipos de soporte.
* Identificar las necesidades del negocio.
* Planificar proyectos.
* Supervisar aplicaciones y sistemas de software.
* Llevar a cabo pruebas de estrés, pruebas de rendimiento, pruebas funcionales y pruebas de escalabilidad.
* Escribir y ejecutar scripts de prueba.
* Realizar pruebas manuales y automatizadas.
* Pruebas en diferentes entornos, incluyendo web y móvil.
* Escribir informes de fallos.
* Llevar a cabo la planificación de recursos.
* Revisar la documentación.
* Trabajar para cumplir los con plazos departamentales y de proyectos.
* Proporcionar garantía de calidad.
* Proporcionar información objetiva a los equipos de proyectos de desarrollo de software.
* Detectar potenciales fallos.
* Pruebas de diseño para mitigar el riesgo.
* Presentar los resultados a los equipos de desarrollo de software y al cliente.
* Trabajar en múltiples proyectos a la vez.
* Análisis de documentación.

**¿Qué habilidades necesita un tester?**

* Ser curioso
* Ser minucioso en el detalle
* Ser imaginativo
* Ser un buen comunicador

El área de calidad de software es multidisciplinar, por lo que podemos tener una carrera orientada hacia las pruebas manuales, hacia el análisis de pruebas o hacia la gestión de equipos. También existen otros ámbitos en los cuales las pruebas son muy importantes, por ejemplo, las pruebas de rendimiento, las pruebas de seguridad o las pruebas de automatización. El testing está evolucionando mucho en los últimos años y es una profesión que está en auge. Se están creando cada vez más y más necesidades sobre las pruebas.

1. **Concepto de testing o pruebas de software**

Cuando pensamos en el concepto de testing, pueden surgir varias interpretaciones, veamos algunas de ellas:

* Metodología para encontrar defectos en el software.
* Proceso utilizado para medir la calidad de una aplicación.
* Proceso de verificación del correcto funcionamiento de un sistema.

El testing es el proceso que permite verificar la calidad de un producto de software, permitiendo identificar posibles fallos en la implementación, calidad o usabilidad de un programa.

El proceso de testing no solo es realizar una serie de pruebas en forma secuencial, es decir, una tras otra, sino que también este proceso incluye:

* *actividades de planificación* de las pruebas,
* la *preparación específica* de la prueba y
* la *evaluación* de los productos de software para determinar si cumple o no con los requerimientos.

El testing puede probar la presencia de errores pero no la ausencia de estos. Durante la planificación, lo que haremos es determinar que pruebas vamos a realizar.

Luego, durante la ejecución vamos a correr esas pruebas, pero ¿qué sucede si existe alguna funcionalidad que no estamos teniendo en cuenta? Es por ello que **solo** podremos probar todo aquello que hayamos definido adecuadamente.

1. **Fundamentos del testing**
2. **Concepto de error, defecto y fallo**

* **Error:** es una acción realizada por una persona que produce un resultado incorrecto. El error es una equivocación del desarrollador o del analista. Un error puede llevarnos a generar uno o más *defectos.*
* **Defecto (defect):** desperfecto que se encuentra en un componente o sistema y puede causar que el sistema falle en su funcionamiento.
* **Fallo (failure):** es la exposición física o visible de un defecto. Un ejemplo es cuando nuestra pantalla se pone de color “azul” por un fallo del sistema.

Como resumen, un error puede generar uno o más defecto y un defecto va a generar un fallo.

Un error puede deberse a dos posibles causas: al error humano o a causas ambientales (fallas en el disco, fluctuación en el suministro de electricidad entre otros).

1. **Concepto de requerimiento y calidad**

* **Requerimiento:** se trata de describir un atributo funcional del sistema que se considera obligatorio, es decir, que un sistema no será aprobado por el cliente si no cumple con los requisitos o requerimientos definidos.

Por ejemplo, si el cliente especificó que un documento puede ser guardado con su nombre original o puede generarse una copia con la opción de guardar como.

Si yo genero un sistema que solo permite guardar sin especificar el nombre y la ubicación de mi archivo, ese requerimiento no se verá aprobado.

* **Calidad:** es el grado en que un componente de un sistema satisface los requerimientos especificados y las expectativas del cliente.
* **Calidad del software:** es la suma de todos los atributos que se refieren a la capacidad de satisfacer los requerimientos planteados.

**¿Qué son los atributos de la calidad?**

Los atributos pueden estar organizados en dos grupos: *funcionales y no funcionales*.

Dentro de los *atributos funcionales* encontramos dos características sumamente importantes: correctitud y completitud.

La correctitud implica que la funcionalidad satisface correctamente los atributos requeridos. Mientras que la completitud significa que la funcionalidad del software satisface todos los requisitos que el cliente ha pedido.

Dentro de los *atributos no funcionales* encontramos muchos más atributos: fiabilidad (confiable en el tiempo), usabilidad (facilidad de uso / uso intuitivo), portabilidad (fácil de instalar / desinstalar y fácil de transferir a otro entorno de trabajo), eficiencia (requiere de un mínimo de recursos para realizar la tarea específica) y mantenibilidad (medida del esfuerzo que se requiere para realizar cambios en mi sistema).

Estos atributos son más difíciles de lograr porque no están perfectamente definidos o bien establecidos.

1. **¿Por qué son necesarias las pruebas?**

A lo largo del tiempo, el desarrollo de los lenguajes de programación ha ido creciendo en forma notable. Como así también, las técnicas y estrategias para el trabajo en equipo y la estructuración de sistemas, esto ha posibilitado asegurar la calidad del software.

Durante el desarrollo de diferentes tipos de sistemas o productos de software encontramos ejemplos que si bien pueden parecer casos extremos, permiten demostrar por qué es necesario realizar el proceso de testing.

A lo largo de la historia espacial o armamentística, encontramos problemas en el trabajo de los desarrolladores que generaron grandes gastos de dinero adicional y pudieron causar desastres naturales, como así también en la medicina nuclear, donde por errores humanos se vio afectada la salud de otras personas (control de radiación por ejemplo).

*En síntesis:*

* El software es probado por personas y las personas nos equivocamos
* A medida que se incrementa la presión en las entregas, muchas veces, no alcanza el tiempo para verificar el funcionamiento.
* Es una forma de demostrar que el software no tiene fallas y hace lo que tiene que hacer.
* Por último, por una cuestión de costos, es más barato encontrar los fallos en el momento de desarrollo que cuando está funcionando en el cliente.

1. **Objetivos de las pruebas**

* Adquirir conocimiento sobre el funcionamiento del sistema
* Confirmación de la funcionalidad
* Generación de información sobre posibles riesgos antes de ser entrega al usuario
* Generar confianza a partir de la validación del funcionamiento

1. **Duración del proceso de testing**

Es imposible revisar y chequear todas las opciones, por lo tanto, podemos preguntarnos ¿cuánto testing es necesario?

Para ello es necesario evaluar riesgos (de quien es la responsabilidad en caso de fallos) y prioridades y depende de cada proyecto.

**Felicitaciones! Nos vemos en el foro!**