Descripciones técnicas del proyecto *Agending System*

DISEÑO DESARROLLO E IMPLEMENTACIÓN DE SOFTWARE

Tabla de contenido

[***1*** ***Introducción.*** 3](#_Toc66683946)

[***1.1*** ***Alcancé del documento.*** 4](#_Toc66683947)

[***1.2*** ***Objetivo del documento.*** 4](#_Toc66683948)

[***2*** ***Tipos de pruebas de software.*** 4](#_Toc66683949)

[***2.1*** ***Pruebas funcionales.*** 4](#_Toc66683950)

[***2.1.1*** ***Pruebas Exploratorias*** 4](#_Toc66683951)

[***2.1.2*** ***Pruebas de Regresión*** 5](#_Toc66683952)

[***2.1.3*** ***Pruebas de Compatibilidad de entorno*** 5](#_Toc66683953)

[***2.1.4*** ***Pruebas Libres*** 5](#_Toc66683954)

[***2.1.5*** ***Pruebas de Humo*** 5](#_Toc66683955)

[***2.1.6*** ***Pruebas de Mono*** 6](#_Toc66683956)

[***2.1.7*** ***Pruebas de Sanidad*** 6](#_Toc66683957)

[***2.2*** ***Pruebas no funcionales.*** 6](#_Toc66683958)

[***2.2.1*** ***Pruebas de Carga o Rendimiento*** 6](#_Toc66683959)

[***2.2.2*** ***Pruebas de Recuperación o vuelta atrás*** 6](#_Toc66683960)

[***2.2.3*** ***Pruebas de Instalación*** 6](#_Toc66683961)

[***2.2.4*** ***Pruebas Estructurales*** 6](#_Toc66683962)

[***2.2.5*** ***Pruebas de configuración*** 7](#_Toc66683963)

[***3*** ***¿Por qué son necesarias las pruebas?*** 7](#_Toc66683964)

[*3.1* ***QA le ahorra dinero y muchos dolores de cabeza.*** 7](#_Toc66683965)

[*3.2* ***QA previene emergencias corporativas catastróficas*** 7](#_Toc66683966)

[*3.3* ***QA inspira confianza del cliente*** 8](#_Toc66683967)

[*3.4* ***QA mantiene una gran experiencia de usuario*** 8](#_Toc66683968)

[*3.5* ***QA trae más ganancias*** 8](#_Toc66683969)

[*3.6* ***QA aumenta la satisfacción del cliente*** 8](#_Toc66683970)

[*3.7* ***QA promueve la organización, la productividad y la eficiencia*** 8](#_Toc66683971)

[***4*** ***Referencias.*** 9](#_Toc66683972)

# ***Introducción.***

## ***Alcancé del documento.***

Presentar los tipos de pruebas que se deben realizar al momento de desarrollar un sistema de información para su correcto funcionamiento y que deberán ser implementadas durante el desarrollo de la herramienta *Agending System* - sistema de agendamiento de citas.

## ***Objetivo del documento.***

Construir un documento con las definiciones de los diferentes tipos de pruebas de software que se deben realizar a un sistema durante su etapa de desarrollo, las ventajas y desventajas de realizar pruebas de software y la determinación de cuales serian las mas adecuadas para aplicar durante el desarrollo de la herramienta Agending *System.*

# ***Tipos de pruebas de software.***

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.

## ***Pruebas funcionales.***

Las pruebas funcionales se definen teniendo como fuente los requisitos del sistema, estas pruebas validan y verifican que el producto cumple con lo especificado y hace lo que debe y cómo lo tiene que hacer dando también una idea del grado de calidad del software.

### ***Pruebas Exploratorias***

El término “[testing exploratorio](https://www.softwaretestinghelp.com/exploratory-testing-tours/" \t "_blank)” fue introducido por **Cem Kaner**, consiste en **ejecutar las pruebas a medida que se piensa en ellas**, sin gastar demasiado tiempo en prepararlas o explicarlas, confiando en los instintos.

La metáfora del turista.

*Donde se compara el testing exploratorio con el proceso que sigue un turista que decide visitar una ciudad “por su cuenta” y que, a pesar de no contar con una guía detallada, al final de su visita conocerá la ciudad, sabrá qué lugares evitar, cuáles repetir y por dónde comenzar a explorar nuevas partes en el futuro.*

### ***Pruebas de Regresión***

Las pruebas de regresión están pensadas para evitar el **efecto onda** en un producto estable en el momento de introducir un cambio.

Es decir, **evita que al introducir nuevos cambios en un software se obtengan comportamientos no deseados** o defectos en otros módulos no modificados.

Sería ideal poder ejecutar este tipo de pruebas sobre un producto cada vez que se realice un cambio de funcionalidad en él o se introduzca código para solucionar un error (pruebas de confirmación).

El caso ideal sería poder automatizar todo este tipo de pruebas ya que según su naturaleza están pensadas para ejecutarse repetidamente en la vida del producto.

### ***Pruebas de Compatibilidad de entorno***

Las pruebas de compatibilidad son pruebas en las que se ejecuta el **mismo producto en diferentes entornos**, para chequear que funcionalmente se comportan igual.

Como puede ser: ejecutar una web en diferentes navegadores o un programa de escritorio en diferentes sistemas operativos.

### ***Pruebas Libres***

Son las pruebas que se ejecutan **sin un Test Plan determinado**. Comparte la misma filosofía que el testing exploratorio.

Son un complemento ideal para la fase final de ejecución de un proyecto donde el tester ya ha ejecutado el test plan y puede probar fuera de un guión lo que él considere menos maduro o más propenso a fallo sin atender tanto a la burocracia de la ejecución del test plan.

### ***Pruebas de Humo***

Este tipo de pruebas es una **revisión rápida inicial** de la versión de software entregada por desarrollo donde se verificará de forma general sin entrar en detalle las principales funcionalidades del mismo y se asegurará que no tiene defectos que interrumpa el funcionamiento básico del mismo, para que el equipo de testing pueda seguir probando entrando más en detalle.

El nombre viene de las pruebas que se hacen en las viviendas relacionadas con la instalación de las tuberías de agua en las que se le aplica una corriente de humo para ver si hubiera alguna fuga.

### ***Pruebas de Mono***

Son las pruebas que se hacen sin atender mucho al funcionamiento teórico del producto, simplemente consiste en **navegar por los distintos caminos del software sin un orden determinado** e intentando ejecutar todas las opciones posibles.

El comportamiento esperado de estas pruebas no es más que la robustez del programa.

El objetivo final es comprobar que con este comportamiento de usuario loco el programa tiene una respuesta adecuada y no se cuelga o presenta un fallo grave.

### ***Pruebas de Sanidad***

La idea principal de las pruebas de sanidad es parecida a las de regresión, pero suelen ser menos exhaustivas.

Se refieren a un conjunto de **pruebas que se ejecutan para comprobar que todo funciona correctamente** después de alguna intervención o modificación.

## ***Pruebas no funcionales.***

Las pruebas No Funcionales se centran en aspectos muy importantes del comportamiento del producto pero que no están relacionados con las funciones que realiza el sistema.

### ***Pruebas de Carga o Rendimiento***

Verifica el comportamiento del **sistema frente a un crecimiento** de carga de consultas, accesos, etc…

En este tipo de pruebas se centra en el comportamiento del producto frente a diferentes escenarios de niveles de uso y ocupación de HW.

### ***Pruebas de Recuperación o vuelta atrás***

Verifica los procedimientos de **recuperación del sistema ante fallos**.

### ***Pruebas de Instalación***

Se verifica todo lo relacionado con el **despliegue del producto**: documentación, instalación de software, configuración pos-instalación, etc.

### ***Pruebas Estructurales***

Se realizan para ejecutar la estructura interna del producto intentando ejecutar todos los caminos posibles del código. Se refieren a las pruebas de **caja blanca**.

### ***Pruebas de configuración***

Consiste en la realización de pruebas sobre el **producto frente a diferentes configuraciones HW o SW** para comprobar que su comportamiento es el esperado.

# ***¿Por qué son necesarias las pruebas?***

Querámoslo o no, actualmente el uso de software es pan nuestro de cada día. En momento, mientras escribo tengo en la pantalla de mi computadora 3 escritorios y 15 pestañas abiertas con diferentes herramientas de software. Además, tengo al lado mi teléfono móvil con aplicaciones que uso todo el tiempo, y una Tablet que me sirve para leer y ver mis series favoritas de Netflix. En fin, puedo decir que mi vida productiva depende enteramente de software que funcione. Y sé que no estoy sólo en esto.  
  
¿Acaso no es frustrante cuando el software que está usando de pronto no funciona? Personalmente si un sitio web o una aplicación móvil tarda 3 segundos más en cargarse, me molesto y me distraigo. Seamos sinceros, vivimos en una época en la que nadie es paciente con software de mala calidad.  
  
Es por eso que el aseguramiento de la calidad (QA) debe ser parte integral en cualquier proyecto de software. La ciencia detrás de las pruebas QA está en identificar con precisión la calidad del software con el objetivo de asegurar que el software funcione como se espera que funcione en **todo momento**. El término se refiere a diferentes métodos y procesos para probar software y garantizar su calidad.  
Por eso las pruebas QA son su mejor aliado en la relación costo beneficio, y para poder garantizarse software de excelente calidad.

## ***QA le ahorra dinero y muchos dolores de cabeza.***

¿Cuánto dinero le cuesta un proyecto de software defectuoso? Les cuesta usuarios y a los clientes. Y es bien sabido que cuanto más tiempo pasa inadvertido un error en su software, más difícil y costoso se vuelve solucionarlo. Al emplear pruebas de aseguramiento de la calidad durante todo el proceso de desarrollo del software, ahorrará tiempo y dinero después de la implementación.

## ***QA previene emergencias corporativas catastróficas***

Con el software corporativo, lo que está en juego es aún más importante. Los errores en el software corporativo pueden ocasionar apagones del sistema, datos faltantes y fallas de comunicación. Si va a utilizar software en una empresa o para manejar información sensible, entonces debe asegurarse de que el software funcionará exactamente como debe funcionar. No hay margen de error.

## ***QA inspira confianza del cliente***

Al hacer que el Software Testing para Aseguramiento de la Calidad tenga una clara prioridad para el desarrollo de software, está enviando un mensaje a sus clientes indicando que desea que su software tenga el mayor éxito posible. Esto es extraordinariamente importante cuando se trata de ofrecer calidad y forjar relaciones a largo plazo.

## ***QA mantiene una gran experiencia de usuario***

Cada vez es más claro que la experiencia del usuario creará o romperá un producto. Si el software está fallando o es lento, entonces interrumpe la experiencia del usuario con el producto. La mala experiencia del usuario resulta en insatisfacción y frustración. Una buena experiencia de usuario, lo que obtiene  
cuando ha probado meticulosamente un producto de software, resulta en un usuario satisfecho, que es mucho más probable que recomiende el producto y su negocio a otros.

## ***QA trae más ganancias***

Si está creando un software que va a comercializar o vender, entonces invertir en QA significará que puede vender su producto a una tasa mayor. No hay nada peor que un usuario enojado que pagó por un producto que no funciona bien.

## ***QA aumenta la satisfacción del cliente***

En relación con el primer punto, este sexto beneficio se centra en la reputación que la satisfacción del cliente aporta a su empresa, no solo la ganancia. Al ofrecer un software de calidad que funciona cuándo y cómo desea que funcione, aumentará su reputación al producir clientes satisfechos. No ponga a prueba la paciencia de sus clientes con software defectuoso que tiene que arreglar constantemente.  
Proporcione calidad desde el principio y le recompensarán con su lealtad.

## ***QA promueve la organización, la productividad y la eficiencia***

Usted definitivamente no quiere lidiar con un caos por software defectuoso, comunicación frenética, ni correcciones apresuradas. Organizarse con pruebas de control de calidad desde el comienzo de su estrategia de desarrollo le permitirá trabajar en paz y ser más productivo con su tiempo. Al utilizar metodologías ágiles, donde los desarrolladores de software crean y entregan pequeños fragmentos de  
un producto en una línea de tiempo clara, se puede comenzar a probar el software a medida que se crea, en lugar de esperar siempre hasta el final. Cuando las pruebas de software son una parte integral de su estrategia de software, usted gana, su cliente gana y sus usuarios ganan.

# ***Referencias.***

<https://es.wikipedia.org/wiki/Pruebas_de_software>

<https://testerhouse.com/teoria-testing/pruebas-funcionales/#:~:text=Pruebas%20No%20Funcionales&text=Pero%20una%20divisi%C3%B3n%20muy%20com%C3%BAn,el%20uso%20final%20del%20entregable>.

<http://www.avantica.com/es/blog/7-beneficios-de-hacer-software-testing-y-pruebas-de-calidad>