

Módulo 2 / Encuentro 11/17

Casos de

prueba 1/2

## 

## 

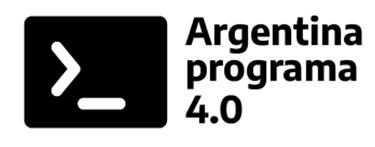
## OBJETIVOS DEL MÓDULO 2

¿Qué habilidades desarrollarás?

* Manejo y priorización de la información
* Introducción al manejo de proyectos
* Análisis y diseño de pruebas

¿Qué herramientas técnicas aprenderás?

* Procesos de Prueba
* Análisis de Requerimientos
* Matriz de Trazabilidad de Requerimientos
* Historias de usuario
* Documentación de pruebas
* Introducción a formularios HTML



Hoy iniciaremos el encuentro mirando un video sobre un ejemplo de un caso de prueba. Este ejemplo es más complejo que el ejemplo que vimos en el encuentro anterior.

|  |  |
| --- | --- |
|  | [**Haz click aquí**](https://youtu.be/qASPwVFGj3E). Este video tiene una duración de 5 minutos. |

# Paso a paso

Ahora veremos paso a paso cómo debemos realizar un plan de pruebas. Los datos necesarios para identificar y armar cada caso de prueba incluyen diversa información, que vimos anteriormente, pero volveremos a repasar aquellas más relevantes:

|  |
| --- |
| **Identificador:** Identificador único para el caso de prueba. Es el código con el objetivo de vincular, por ejemplo, un informe de errores al caso de prueba en el cual ha sido detectado. |
| **Conjunto de prueba:** Identificador de los conjuntos de prueba donde el caso de prueba es usado. |
| **Nombre de la Funcionalidad:** Funcionalidad del IP (Inventario de Prueba) que se testea con este caso de prueba. |
| **Prioridad:** Prioridad asignada al caso de prueba, por ejemplo: 1- alta, 2- media, 3-baja. |
| **Requisitos:** características del objeto de pruebas que el caso de prueba debe evaluar. |
| **Necesidades de ambientes:** si bien no es un requisito obligatorio, debe saberse e indicarse en qué ambiente se van a correr las pruebas. Lo ideal es que haya un ambiente separado de Testing y exclusivo para correr las pruebas, para no ser afectado por que se esté actualizando el código, como ser en el ambiente de desarrollo o por que estén realizando otras pruebas como puede ser en los ambientes de pre-producción donde el cliente prueba para dar su aceptación o en un ambiente de pruebas de automatización donde se pueden afectar los resultados por el uso de dicho ambiente.  **Situación:** puede haber un caso en el que esté probando una aplicación, un desarrollador esté realizando modificaciones en paralelo a la misma aplicación o alguien pueda actualizar la aplicación después de que termine la prueba. Esto conduce a una situación en la que los resultados de su prueba pueden variar con el tiempo.  Por lo tanto, siempre es mejor agregar una marca de tiempo con el nombre del evaluador en los comentarios de la prueba para que el resultado de una prueba (aprobado o reprobado) se pueda atribuir al estado de una aplicación en ese momento en particular. Alternativamente, puede tener una columna '**Fecha de ejecución**' agregada por separado al caso de prueba que identificará explícitamente la marca de tiempo de la prueba. |
| **HW**: Lista de Hardware requerido para ejecutar el caso de prueba. |
| **SW**: Lista de Software requerido para ejecutar el caso de prueba. |
| **Configuración** (setup): Lista de pasos necesarios para comenzar las pruebas. |
| **Retroceder acciones** (cleanup): Lista de pasos necesarios para retroceder al estado previo a la prueba. |
| **Precondiciones**: situación inicial o previa a la ejecución de pruebas o características de Explicación de la acción que se hará con los datos de la situación inicial. Por ejemplo: dar de alta un elemento, agregar un elemento a una lista. |
| **Resultado esperado**: Especificación del resultado esperado de ejecutar las pruebas. |
| **Dependencias** (si hay): orden de ejecución de casos de prueba, razón de las dependencias. |
| **Valores de entrada**: descripción de los datos de entrada de un objeto de pruebas. Puede ser un dibujo o un texto explicando los datos con los que se probará el caso. Debe ser claro y no dejar dudas. |

Los siguientes datos se completan una vez que el caso de prueba se ejecutó:

* **Resultado real**: Especificación de lo que realmente ocurrió: Pasó/Falló/No se pudo ejecutar
* **Post-caso de prueba condiciones:** características de un objeto de prueba tras la ejecución de pruebas, descripción de su situación tras la ejecución de las pruebas. Puede ser un dibujo o texto explicando cómo se espera que los datos iniciales queden luego de ejecutar la acción. Debe ser claro y no dejar dudas.

Ejemplo

· Identificador: 1

· *Nombre:* Alta de elemento al final de una lista ordenada

· Situación inicial: Lista:(3 → 5 → 7 → 15)

· *Acción:* Agregar a la lista el elemento 18

· Resultado esperado: Lista: (3 → 5 → 7 → 15 → 18)

* **Duración**: Tiempo reloj en ejecutar el caso de prueba.
* **Esfuerzo**: Horas-Persona requeridas para configurar y ejecutar el caso de prueba.
* **Fecha**: Fecha en la que se realizó la prueba.
* **Identificación del Defecto**: En caso de que la prueba falle se debe identificar el incidente reportado para su seguimiento.
* **Versión**: Identificador de la versión probada.
* **Tester**: Nombre del tester que ejecutó el caso de prueba.
* **Evidencias**: son los resultados que se adjuntan con foto/s de la/s pantallas con los resultados (screenshot) y pueden ser también con los pasos realizados. También se pueden realizar videos para mostrar el paso a paso y el resultado final.

Cada vez que se ejecuta un caso debe registrarse la información resultante para que quien sea responsable del código probado, sepa si está correcto o se produjo algún error o resultado no deseado.

Aquí se obtiene un resultado real de la ejecución del código que denominamos Resultado Final de la Ejecución del Caso de Prueba.

Las ejecuciones de casos se agrupan en “Corridas” que se deben identificar para saber cuándo (y a veces quién) realizó una ejecución.

**Material de apoyo sugerido para revisar pre o post del encuentro en vivo:**

1. [APRENDE Cómo Diseñar CASOS DE PRUEBA 🚀 (con EJEMPLOS PRÁCTICOS)](https://youtu.be/rQf50sYvgJE)
2. [Cómo Escribir Casos de Pruebas | Paso a paso](https://youtu.be/laawNIdX9js)
3. [Como diseñar y mantener casos de prueba - Jorge Gomez](https://youtu.be/ZEqwbR1YyJU)

## 

## Consideraciones respecto de las pruebas

El fundamento respecto a la Prueba de Software es que no se puede probar completamente un sistema compuesto por varios programas, por lo que en el momento de realizar las pruebas se deben tomar decisiones respecto a cómo se van a diseñar los casos de prueba. Otro punto importante a tener en cuenta es la actitud que debe tener la persona que realiza las pruebas.

La prueba exhaustiva requiere probar el comportamiento de un programa para todas las combinaciones válidas e inválidas de entradas. Además, se debe probar esto en cada punto donde se ingresan datos, bajo cada estado posible del programa en ese punto. Esto no es posible, ya que el tiempo requerido y el costo es prohibitivo.

En un mundo ideal, desearíamos probar cada permutación posible de un programa. Incluso un programa aparentemente simple puede tener centenares o millares de combinaciones posibles de la entrada y de la salida. Crear los casos de prueba para todas estas posibilidades no es económicamente factible. Este problema fundamental, tendrá implicancias para la economía de las pruebas, suposiciones que el tester tendrá que hacer sobre el programa y la manera en el cual los casos de prueba se diseñan.

El objetivo debe ser maximizar la producción de las pruebas, esto es, maximizar el número de los errores encontrados por un número finito de los casos de prueba.

La forma en cómo se seleccionan los casos de prueba es una de las principales decisiones a tomar, teniendo en cuenta las siguientes consideraciones:

Los casos de prueba deben ser revisados regularmente, escribiendo nuevas pruebas que ejerciten diferentes partes del software, con el objetivo de encontrar más defectos

Otro aspecto que se debe considerar al realizar pruebas es decidir cuándo el programa falla para un conjunto de datos de entrada, o sea, conocer cuál es la salida esperada del programa. Este problema es conocido como **el problema del oráculo**.

Un oráculo es cualquier agente (humano o mecánico) que decide si un programa se comportó correctamente en una prueba dada, y produce por consiguiente un veredicto de "pasó" (pass) o de "falló" (fail).

Un oráculo es lo que nos dice qué resultado debemos esperar ante determinadas entradas y condiciones de ejecución.

Básicamente es el mecanismo, ya sea manual o automático, de **verificar si el comportamiento del sistema es el deseado o no**. Para esto, el oráculo deberá comparar el valor esperado contra valor obtenido, el estado final esperado con el estado final alcanzado, el tiempo de respuesta aceptable con el tiempo de respuesta obtenido, etc. Pueden ser:

* personas
* modelos
* Documentos
* productos similares
* normas o estándares

Existen diversas clases de oráculos, y la automatización del oráculo puede llegar a ser muy compleja y costosa. El oráculo más común es el oráculo entrada/salida, que especifica la salida esperada para una entrada específica. Si el programa hace exactamente lo que su especificación dice que debe hacer y no hace nada más, entonces cumple con su especificación.

No es razonable decir que un programa es correcto si cumple con su especificación, ya que las especificaciones pueden tener errores.

En este punto es interesante reflexionar sobre algo a lo que se llama la Paradoja del Pesticida: los insectos (bugs, refiriéndose a fallos) que sobreviven a un pesticida se hacen más fuertes, y resistentes a ese pesticida. O sea, **si diseñamos un conjunto de pruebas probablemente ciertos bugs sobrevivan**. Si luego diseñamos una técnica más completa y llamémosle “exhaustiva”, entonces encontraremos más bugs, pero otros seguirán sobreviviendo. Al fin de cuentas **los que van quedando son los más duros de matar, y se van haciendo resistentes a los distintos pesticidas.**

# Obtención de casos de prueba a partir de requisitos

El diseño de casos de prueba debe ser un proceso controlado.

Los casos de prueba pueden ser creados **formal** o **informalmente**, dependiendo de las características del proyecto y la madurez del proceso en uso.

### Cobertura de pruebas

Es una medida de calidad de las pruebas. Se definen cierto tipo de entidades sobre el sistema, y luego se intenta cubrirlas con las pruebas. Es una forma de indicar cuándo probamos suficiente, o para tomar ideas de qué otra cosa probar (pensando en aumentar la cobertura elegida).

**¿NECESITAS UN EJEMPLO? **

Ejemplo: podríamos decir que la cobertura es como cuando barremos la casa. Barro la cocina, el comedor, el baño y mi dormitorio, pero siempre se me olvida el segundo dormitorio, eso es que en mi barrido no estoy cubriendo este dormitorio. Mide la calidad de mi barrido, y a su vez me da una medida para saber cuándo tengo que terminar de barrer: cuando cubra cada habitación, por ejemplo.

Ahora, lograr el 100% de cobertura con ese criterio, ¿indica que mi casa está limpia? No, porque el garaje, el patio y la lavandería que están en el fondo de la casa, no los he tenido en cuenta. Por lo cual hay que manejar el concepto con cuidado. Tener cierto nivel de cobertura es un indicador de la calidad de las pruebas, pero nunca es un indicador de la calidad del sistema, por ejemplo, ni me garantizará que está todo probado.

¿Entonces para qué me sirve?

* Medida de calidad de cómo barro
* Me indica cuándo parar de barrer
* Me sugiere qué más barrer

Unos criterios pueden ser más fuertes que otros, entonces el conocerlos me puede dar un indicador de qué tan profundas son las pruebas, cuándo aplicar uno y cuándo otro. Se dice que un criterio A incluye a otro criterio B cuando cualquier conjunto de casos de prueba que cubre el criterio A también cubre el criterio B.

Criterio 1: barrer cada habitación.

Criterio 2: barrer cada habitación dentro o fuera de la casa principal (habitaciones, comedor, cocina, baño, garaje, lavandería, etc.).

Criterio 3: barrer cada habitación, incluso en las esquinas, porque ahí hay más posibilidades de que se acumule suciedad.

Criterio 3 incluye al criterio 2, el cual incluye al criterio 1 (y la relación es transitiva, con lo cual el criterio 3 integra al criterio 1).

## Trazabilidad

Las pruebas deben ser trazables: ¿qué casos de prueba han sido incluido en el catálogo o listado de pruebas, basados en qué requisitos?

Las consecuencias de los cambios en los requisitos sobre las pruebas a realizar pueden ser identificadas directamente.

La trazabilidad también ayuda a determinar la cobertura de requisitos.

### Especificación de casos de prueba

Se han desarrollado muchas maneras de escribir y especificar estos Casos. Aquí presentaremos una adaptación que busca simplificar las pruebas:

Ejemplo: si quiero probar si mi solución de agregar un elemento a una lista simplemente vinculada que está ordenada podría plantear las siguientes situaciones o CASOS (estos son sólo algunos ejemplos):

- Agregar un elemento a una lista vacía.

- Agregar un elemento a la lista que tiene un solo elemento.

- Agregar un elemento al principio de la lista.

- Agregar un elemento al medio de la lista.

- Agregar un elemento al final de la lista.

- Agregar un elemento que ya existe en la lista.

**Construcción de Casos de Prueba**: Este paso debe realizarse muy temprano en el proceso de desarrollo, tal como se mencionó anteriormente. El resultado de este paso es una especificación de Casos de Prueba, que pueden tener una estructura como la que proponemos en el punto siguiente (“*Estructura de los Casos de Prueba*”).

**Ejecución de los Casos de Prueba**: *(comúnmente llamada “corrida”)* La ejecución de los casos se realiza una vez terminada la codificación. Esta ejecución consiste en tomar cada especificación de caso de prueba, ejecutar el software que estamos probamos (puede ser una porción de código) y registrar el resultado de la ejecución. Normalmente una corrida implica la ejecución de TODOS los casos de prueba. Un mismo Caso de Prueba se puede ejecutar muchas veces, tantas como corridas se realicen, idealmente al menos cada vez que se modifica algo del código.

### Los 3 problemas más comunes en casos de prueba

#### Pasos compuestos

En primer lugar, ¿qué es un paso compuesto? Es aquel que se puede dividir en varios pasos individuales.

Ejemplo: estás dando instrucciones desde el punto A al punto B: si dices que 'Ve al lugar XYZ y luego al ABC', esto no tendrá mucho sentido, porque aquí nos encontramos pensando: '¿Cómo puedo llegar a XYZ en primer lugar”- en lugar de comenzar con “Gire a la izquierda desde aquí y avance 1 kilómetro, luego gire a la derecha en R número 11 para llegar a XYZ ” podría lograr mejores resultados.

Las mismas reglas exactas se aplican a las pruebas y sus pasos también.

Por ejemplo: escribiendo una prueba para Amazon.com: haga un pedido de cualquier producto. Recuerde, las pruebas siempre tratan sobre 'Cómo' realizar la prueba, por lo que es importante escribir los pasos exactos de 'Cómo pagar y pagar' en su prueba.

1. Dirigirse a: Amazon.com
2. Busque un producto ingresando la palabra clave / nombre del producto en el campo 'Buscar' en la parte superior de la pantalla.
3. De los resultados de búsqueda que se muestran, elija el primero.
4. Haga clic en Agregar al carrito en la página de detalles del producto.
5. Checkout y pago.
6. Consulta la página de confirmación del pedido.

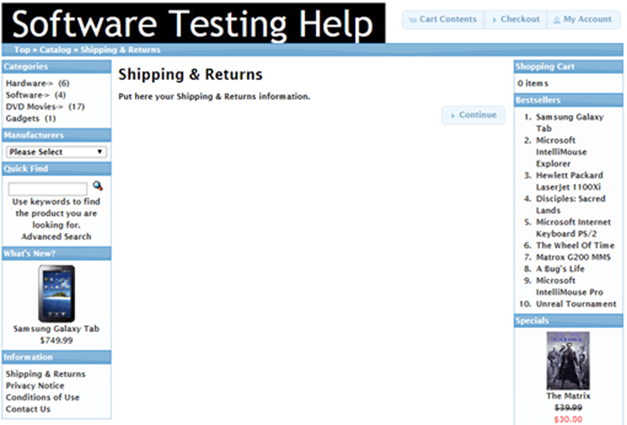
El paso “5” es un paso compuesto, por lo tanto, el caso anterior es más efectivo cuando se escribe de la siguiente manera:

1. Dirigirse a: Amazon.com Busque un producto ingresando la palabra clave / nombre del producto en el campo 'Buscar' en la parte superior de la pantalla.
2. De los resultados de búsqueda que se muestran, elija el primero.
3. Haga clic en Agregar al carrito en la página de detalles del producto.
4. Haga clic en Pagar en la página del carrito de compras.
5. Ingrese la información de CC, envío y facturación.
6. Haga clic en Pagar.
7. Consulta la página de confirmación del pedido.

El comportamiento de la aplicación se toma como comportamiento esperado

Cada vez más proyectos tienen que afrontar esta situación en estos días. La falta de documentación, la programación extrema, los ciclos de desarrollo rápidos son algunas de las razones que nos obligan a confiar en la aplicación (una versión anterior más o menos) para escribir las pruebas o para basar las pruebas en sí.

Ejemplo: si la siguiente es la página para la que está escribiendo / diseñando los pasos de prueba:



Si los pasos de mi caso de prueba son los siguientes:

Caso 1:

1. Inicie el sitio de compras.
2. Haga clic en Envío y devolución. Resultado esperado: la página de envío y devolución se muestra con 'Ponga su información aquí' y un botón 'Continuar'.

Entonces, esto es incorrecto.

Caso 2:

1. Inicie el sitio de compras.
2. Haga clic en Envío y devolución.
3. En el cuadro de texto 'Ingrese el número de pedido' presente en esta pantalla, ingrese el número de pedido.
4. Haga clic en Continuar - Resultado esperado: se muestran los detalles del pedido relacionados con el envío y las devoluciones.

El caso 2 es un mejor caso de prueba porque a pesar de que la aplicación de referencia se comporta de manera incorrecta, solo la tomamos como una guía, **investigamos más y escribimos el comportamiento esperado según la funcionalidad correcta anticipada**.

## Varias condiciones en un caso

Los siguientes son los pasos dentro de una prueba para una función de inicio de sesión.

Ejemplo:

1. Ingrese detalles válidos y haga clic en Enviar.
2. Deje el campo Nombre de usuario vacío. Haga clic en Enviar.
3. Deje el campo de la contraseña vacío y haga clic en Enviar.
4. Elija un nombre de usuario / contraseña que ya haya iniciado sesión y haga clic en Enviar.

Lo que tenían que ser 4 casos diferentes se combinan en uno. Podrías estar pensando: ¿Qué hay de malo en eso? Está ahorrando mucha documentación y lo que puedo hacer en 4, lo estoy haciendo en 1. Bueno, no del todo, ¿por qué?

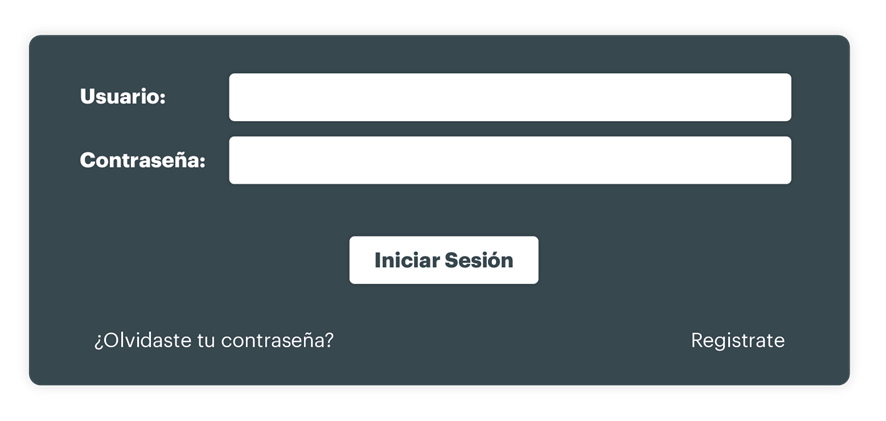
¿Qué pasa si una de las condiciones falla? Tenemos que marcar toda la prueba como 'fallida'. Si marcamos el caso completo como 'fallido', significa que las 4 condiciones no están funcionando, lo cual no es realmente cierto.

Las pruebas deben tener un flujo. Desde la condición previa hasta el paso 1 y todos los pasos. Si sigo este caso, en el paso A, si tiene éxito, iniciaré sesión en la página, donde la opción 'iniciar sesión' ya no está disponible. Entonces, cuando llegue al paso B, ¿dónde ingresará el tester el nombre de usuario? Verás, el flujo está roto.

Por eso, escribir pruebas modulares, parece mucho trabajo, pero todo lo que necesitas es separar las cosas y usar a nuestros mejores amigos Ctrl + C y Ctrl + V para trabajar para nosotros.

Ejemplo:

Veamos cómo escribir casos de prueba de manera eficiente para una pantalla de 'Inicio de sesión' familiar y simple como se muestra en la siguiente figura. Los **enfoques de prueba** será casi el mismo incluso para pantallas complejas con más información y características críticas.



El primer enfoque para cualquier proceso de caso de prueba será obtener un prototipo de pantalla, si está disponible. Es posible que esto no esté disponible para algunas de las funcionalidades y depende de la importancia de diseñar un prototipo en las primeras etapas de desarrollo.

Pero, si un [SRS](https://www.utdallas.edu/~chung/RE/Presentations07S/Team_1_Doc/Documents/SRS4.0.doc) [(Especificación de requisitos de software) documento](https://es.myservername.com/writing-test-cases-from-srs-document) está disponible para el proyecto, la mayoría de los prototipos de pantalla son desarrollados por el equipo del proyecto. Este tipo de pantalla simplifica el trabajo del evaluador y aumenta la eficiencia de las pruebas.

A continuación, las especificaciones de requisitos funcionales. Depende del proceso de organización, estará disponible en un conjunto de múltiples documentos.

Por lo tanto, decida cuál es el mejor documento para redactar casos, ya sea un documento de requisitos del usuario o una especificación de requisitos funcionales (o incluso un documento SRS si el equipo de pruebas lo puede entender cómodamente) que proporcionará un flujo funcional completo de lo seleccionado. característica para ser probado.

Una vez que el prototipo de pantalla y las especificaciones funcionales están en su lugar, el tester debe comenzar a escribir los casos con el siguiente enfoque y criterio.

**Pruebas de IU, Interfaz Usuaria:** los controles / campos que son visibles para el usuario. Hay controles estáticos y dinámicos disponibles para probar la función. Por ejemplo, En la pantalla de inicio de sesión anterior, los textos de 'Nombre de usuario y contraseña' son campos estáticos que no requieren la interacción del usuario, solo para mostrar el texto.

**Casos funcionales:** Por otro lado, el botón Iniciar sesión y los Hipervínculos (¿Olvidó su contraseña? Y Registro) son campos dinámicos que requieren la interacción del usuario haciendo clic en los controles, que realizarán alguna acción después.

**Casos de bases de datos:** Una vez que el usuario ingresa el nombre de usuario y la contraseña, las pruebas se pueden escribir para verificar la base de datos relacionada, si el nombre de usuario y la contraseña están verificados en la base de datos y la tabla correctas y también el usuario tiene permiso para iniciar sesión en la aplicación bajo prueba.

**Pruebas de proceso:** esto está relacionado con el proceso (no las acciones asociadas con los controles visibles disponibles en la pantalla) asociado con la característica y la funcionalidad.

Por ejemplo, Si hace clic en el enlace “Olvidé mi contraseña” en la pantalla de muestra anterior, puede enviar un correo electrónico al usuario. Entonces, tal vez un correo electrónico deba probarse para el proceso y la confirmación adecuados.

Finalmente, mantenga el Enfoque de BAOE (Básico, Alternativo, Opciones y Excepciones) para la cobertura completa del flujo funcional y la característica a probar. Todos los conceptos deben aplicarse a pruebas positivas y negativas.

i) Flujo básico

ii) [Flujo alternativo](https://study.com/academy/lesson/exception-alternate-flow-in-use-case.html)

iii) Opciones

iv) Excepciones

Por ejemplo, Veamos el enfoque BAOE simple para la pantalla de inicio de sesión

Flujo básico: Ingrese la ruta URL del inicio de sesión en cualquier navegador e ingrese la información requerida e inicie sesión en la aplicación.

Flujo alternativo: Instale la aplicación en un dispositivo móvil e ingrese la información requerida e inicie sesión en la aplicación.

Opciones: ¿Cuáles son las opciones disponibles para llegar a la misma pantalla de inicio de sesión?

Ejemplo, después de iniciar sesión en la aplicación, al hacer clic en 'Cerrar sesión' puede aparecer la misma pantalla o, si el tiempo de espera de la sesión o la sesión expiró, el usuario puede acceder a la pantalla de inicio de sesión.

## 

## Excepciones

### ¿Cuáles son las excepciones si mis pruebas son negativas?

Ejemplo si se ingresan credenciales incorrectas en la pantalla de inicio de sesión, si el usuario recibirá un mensaje de error o ninguna acción asociada.

Con toda esta información en la mano, comencemos a escribir los casos de prueba para la pantalla de inicio de sesión, en un formato con la cobertura y trazabilidad completa y con información detallada. La secuencia lógica y la numeración de identificar el ' ID de caso de prueba ' será muy útil para un historial de ejecución de identificación rápida de casos de prueba.

|  |  |  |
| --- | --- | --- |
| Nro | Pasos para reproducir | Comportamiento esperado |
| 1. | Abra un navegador e ingrese la URL para la pantalla de inicio de sesión. | Debería mostrarse la pantalla de inicio de sesión. |
| 2. | Instale la aplicación en el teléfono Android y ábrala. | Debería mostrarse la pantalla de inicio de sesión. |
| 3. | Abra la pantalla de inicio de sesión y compruebe que los textos disponibles estén escritos correctamente. | El texto 'Nombre de usuario' y 'Contraseña' debe aparecer antes del cuadro de texto relacionado. El botón de inicio de sesión debe tener el título 'Iniciar sesión'. '¿Olvidó su contraseña?' Y 'Registro' deberían estar disponibles como enlaces. |
| 4. | Introduzca el texto en el cuadro Nombre de usuario. | El texto se puede ingresar haciendo clic con el mouse o enfocarse usando la pestaña. |
| 5. | Ingrese el texto en el cuadro Contraseña. | El texto se puede ingresar haciendo clic con el mouse o enfocarse usando la pestaña. |
| 6. | Haga clic en ¿Olvidó su contraseña? Enlace. | Hacer clic en el enlace debería llevar al usuario a la pantalla relacionada. |
| 7. | Haga clic en el enlace de registro | Hacer clic en el enlace debería llevar al usuario a la pantalla relacionada. |
| 8. | Introduzca el nombre de usuario y la contraseña y haga clic en el botón Iniciar sesión. | Hacer clic en el botón Iniciar sesión debería llevar a la pantalla o aplicación relacionada. |
| 9. | Vaya a la base de datos y verifique que el nombre correcto de la tabla esté validado con las credenciales de entrada. | El nombre de la tabla debe validarse y debe actualizarse un indicador de estado para un inicio de sesión exitoso o incorrecto. |
| 10. | Haga clic en Iniciar sesión sin ingresar ningún texto en los cuadros Nombre de usuario y Contraseña. | Haga clic en el botón Iniciar sesión para alertar a un cuadro de mensaje 'El nombre de usuario y la contraseña son obligatorios'. |
| 11. | Haga clic en Iniciar sesión sin ingresar texto en el cuadro Nombre de usuario, pero ingrese texto en el cuadro Contraseña. | Haga clic en el botón Iniciar sesión para alertar a un cuadro de mensaje 'La contraseña es obligatoria'. |
| 12. | Haga clic en Iniciar sesión sin ingresar texto en el cuadro Contraseña, pero ingresando texto en el cuadro Nombre de usuario. | Haga clic en el botón Iniciar sesión para alertar a un cuadro de mensaje 'El nombre de usuario es obligatorio'. |
| 13. | Ingrese el texto máximo permitido en los cuadros Nombre de usuario y Contraseña. | Debe aceptar el máximo permitido de 30 caracteres. |
| 14. | Ingrese el nombre de usuario y la contraseña comenzando con los caracteres especiales. | No debe aceptar el texto que comience con caracteres especiales, lo cual no está permitido en el Registro. |
| 15. | Ingrese el nombre de usuario y la contraseña comenzando con espacios en blanco. | No debe aceptar el texto indicando con espacios en blanco, lo cual no está permitido en Registro. |
| 16. | Ingrese el texto en el campo de contraseña. | No debe mostrar el texto real en su lugar debe mostrar el símbolo de asterisco \*. |
| 17. | Actualiza la página de inicio de sesión. | La página debe actualizarse con los campos Nombre de usuario y Contraseña en blanco. |
| 18. | Ingrese el nombre de usuario. | Depende de la configuración de llenado automático del navegador, los nombres de usuario ingresados ​​previamente deben mostrarse como un menú desplegable. |
| 19. | Introduce la contraseña. | Depende de la configuración de autocompletar del navegador, las contraseñas ingresadas anteriormente NO deben mostrarse como un menú desplegable. |
| 20. | Mueva el foco al enlace Olvidé mi contraseña usando Tab. | Deben poder utilizarse tanto el clic del ratón como la tecla Intro. |
| 21. | Mueva el foco al enlace Registro usando Tab. | Deben poder utilizarse tanto el clic del ratón como la tecla Intro. |
| 22. | Actualice la página de inicio de sesión y presione la tecla Intro. | El botón Iniciar sesión debe estar enfocado y la acción relacionada debe activarse. |
| 23. | Actualice la página de inicio de sesión y presione la tecla Tab. | El primer foco en la pantalla de inicio de sesión debe ser el cuadro Nombre de usuario. |
| 24. | Ingrese el usuario y la contraseña y deje la página de inicio de sesión inactiva durante 10 minutos. | La alerta del cuadro de mensaje 'Sesión caducada, ingrese nuevamente el nombre de usuario y contraseña' debe aparecer con los campos de nombre de usuario y contraseña limpios. |
| 25. | Introduzca la URL de inicio de sesión en los navegadores Chrome, Firefox e Internet Explorer. | La misma pantalla de inicio de sesión debe mostrarse sin mucha desviación en el aspecto y la alineación del texto y los controles de formulario. |
| 26. | Ingrese las credenciales de inicio de sesión y verifique la actividad de inicio de sesión en los navegadores Chrome, Firefox e Internet Explorer. | La acción del botón Iniciar sesión debe ser la misma en todos los navegadores. |
| 27. | Verifique que el enlace Olvidé la contraseña y el registro no esté roto en los navegadores Chrome, Firefox e Internet Explorer. | Ambos enlaces deben llevar a las pantallas relativas en todos los navegadores. |
| 28. | Verifique que la funcionalidad de inicio de sesión funcione correctamente en teléfonos móviles con Android. | La función de inicio de sesión debería funcionar de la misma manera que está disponible en la versión web. |
| 29. | Verifique que la funcionalidad de inicio de sesión funcione correctamente en Tab y iPhones. | La función de inicio de sesión debería funcionar de la misma manera que está disponible en la versión web. |
| 30. | Verificar la pantalla de inicio de sesión permite que los usuarios concurrentes del sistema y todos los usuarios obtengan la pantalla de inicio de sesión sin demoras y dentro del tiempo definido de 5-10 segundos. | Esto debe lograrse usando muchas combinaciones de sistema operativo y navegadores, ya sea física o virtualmente, o puede lograrse usando alguna herramienta de prueba de rendimiento / carga. |

## Recopilación de datos de prueba

Cuando se escribe el caso de prueba, la tarea más importante para cualquier tester es recopilar los datos de prueba. Muchos evaluadores omiten esta actividad y la pasan por alto con la suposición de que los casos de prueba se pueden ejecutar con algunos datos de muestra o datos ficticios y se pueden alimentar cuando los datos realmente se requieren. Este es un concepto erróneo crítico de alimentar datos de muestra o datos de entrada desde la memoria mental en el momento de ejecutar casos de prueba.

Si los datos no se recopilan y actualizan en el documento de prueba en el momento de escribir las pruebas, el evaluador pasaría anormalmente más tiempo para recopilar los datos en el momento de la ejecución de la prueba. Los datos de la prueba deben recopilarse tanto para casos positivos como negativos desde la perspectiva del flujo funcional de la característica.

Encuentre un documento de datos de prueba de muestra para las pruebas escritas anteriormente, que a su vez será útil sobre la eficacia con la que podemos recopilar los datos que facilitarán nuestro trabajo en el momento de la ejecución de la prueba.

|  |  |  |
| --- | --- | --- |
| Nro | Propósito de los datos de prueba | Datos de prueba reales |
| 1. | Pruebe el nombre de usuario y la contraseña adecuados | Administrador (admin2015) |
| 2. | Pruebe la longitud máxima del nombre de usuario y la contraseña | Administrador del sistema principal (admin2015admin2015admin2015admin) |
| 3. | Pruebe los espacios en blanco para el nombre de usuario y la contraseña | Ingrese espacios en blanco usando la tecla de espacio para el nombre de usuario y la contraseña |
| 4. | Pruebe el nombre de usuario y la contraseña incorrectos | Admin (activado) (digx ## $ taxk209) |
| 5. | Pruebe el nombre de usuario y la contraseña con espacios no controlados entre ellos. | Admin istrador (admin 2015) |
| 6. | Pruebe el nombre de usuario y la contraseña comenzando con caracteres especiales | $% # @ # $ Administrador (% # \* # \*\* # admin) |
| 7. | Pruebe el nombre de usuario y la contraseña con todos los caracteres pequeños | administrador (admin2015) |
| 8. | Pruebe el nombre de usuario y la contraseña con todos los caracteres en mayúscula | ADMINISTRADOR (ADMIN2015) |
| 9. | Pruebe el inicio de sesión con el mismo nombre de usuario y contraseña con varios sistemas simultáneamente al mismo tiempo. | Administrador (admin2015): para Chrome en la misma máquina y en una máquina diferente con el sistema operativo Windows XP, Windows 7, Windows 8 y Windows Server.  Administrador (admin2015): para Firefox en la misma máquina y en una máquina diferente con el sistema operativo Windows XP, Windows 7, Windows 8 y Windows Server.  Administrador (admin2015): para Internet Explorer en la misma máquina y en una máquina diferente con el sistema operativo Windows XP, Windows 7, Windows 8 y Windows Server. |
| 10. | Pruebe el inicio de sesión con el nombre de usuario y la contraseña en la aplicación móvil. | Administrador (admin2015): para Safari y Opera en móviles, iPhones y tabletas con Android. |

# ¿Qué es una prueba estándar en pruebas web?

Repasemos algunos conceptos que debemos tener presentes:

* Cree casos de prueba que estén completos: pasos, datos, variables, etc. Esto garantizará que los datos / variables (aunque no similares) se puedan reemplazar simplemente cuando se requiera un caso de prueba similar.
* Los criterios de entrada y salida deben definirse adecuadamente.
* Los pasos modificables o la declaración en los pasos deben resaltarse en un color diferente para encontrarlos y reemplazarlos rápidamente.
* El lenguaje utilizado para la creación de casos de prueba estándar debe ser genérico.
* Todas las características de cada sitio web deben cubrirse en los casos de prueba.
* El nombre de los casos de prueba debe ser el nombre de la funcionalidad o la característica que cubre el caso de prueba Esto facilitará mucho la búsqueda del caso de prueba del conjunto.
* Si hay una muestra básica o estándar, un archivo GUI o una captura de pantalla de la función, debe adjuntarse con los pasos correspondientes.

Al usar los consejos anteriores, uno puede crear un conjunto de scripts estándar y usarlos con modificaciones pequeñas o necesarias para diferentes sitios web.

El uso de un conjunto estándar de casos de prueba manuales para diferentes sitios web con modificaciones menores es la mejor manera de realizar una prueba de sitio web. Todo lo que necesitamos es crear y mantener los casos de prueba con los estándares y el uso adecuados.

## Mejoramiento y mantenimiento de los casos de prueba

### Comprobación de los casos de prueba

Una prueba realizada con precisión es fácil de ejecutar lo que significa que, si el tester sigue las instrucciones, el resultado de aprobado o fallido será correcto. Se puede medir fácilmente por medio del tiempo que se tarda en ejecutar la prueba, y si el tester tiene que buscar o no aclaraciones en el proceso de prueba.

### Lenguaje para mejorar la comprobación

Los pasos de los casos de prueba deben ser escritos en forma activa. El tester debe saber qué hacer, y cómo hacerlo.

Por ejemplo, navegar en la página de la tienda online y preparar la lista de lo que va a comprar, para comparar los precios y la variedad con los datos disponibles.

### Controlar longitud para mejorar la comprobación

Es necesario tener en cuenta la longitud de los casos de prueba para saber cuán compleja y precisa es la prueba.

Un buen caso de prueba debe tener entre 8 y 16 pasos ―en el método paso a paso.

Existen varias ventajas en mantener los casos de prueba cortos: se requiere menos tiempo y hay menos posibilidades de cometer errores, de necesitar ayuda o de alguna pérdida de datos.

Con base en la longitud de los casos de prueba es posible estimar con precisión el tiempo y el esfuerzo que se debe invertir en la prueba, lo mismo que sus resultados.

En los casos de prueba de matriz, una buena longitud oscila entre 18 y 20 minutos para la prueba.

## Tipos de Mantenimiento

Si un problema es detectado por el usuario, inmediatamente puede notificarlo al administrador del sistema. Dicha petición debe ser atendida por el administrador y este procederá a diagnosticar de qué tipo de mantenimiento se trata, luego de finalizar las etapas de desarrollo de un sistema.

Atendiendo a estos fines, podemos establecer los siguientes tipos de mantenimiento:

1. **Correctivo.** Cambios precisos para corregir errores del producto de software.
2. **Evolutivo.** Incorporaciones, modificaciones y eliminaciones necesarias en un producto de software para cubrir la expansión o cambio en los requerimientos del usuario.
3. **Adaptativo**. Modificaciones que afectan los entornos en los que el sistema opera, por ejemplo, cambio en las configuraciones del hardware, software de base, gestores de base de datos, comunicaciones, etc.
4. **Perfectivo**. Acciones llevadas a cabo para mejorar la calidad interna de los sistemas en cualquiera de sus aspectos: restructuración de código, definición más clara del sistema y optimización del rendimiento y eficiencia.

Una vez identificado el tipo de mantenimiento y su origen se determina un tiempo razonable para su modificación y prueba, haciéndolo del conocimiento del usuario.

Si se trata de un mantenimiento correctivo, los cuales son más comunes, se verifica y reproduce el problema, o se estudia la viabilidad del cambio propuesto por el usuario. En ambos casos se identifican, según el tipo de mantenimiento de que se trate, cuál es la más adecuada. El plazo y urgencia de la solución a la petición se establece de acuerdo con el estudio anterior.

Las tareas de los procesos de desarrollo que va a ser necesario realizar son determinadas en función de los componentes del sistema actual afectados por la modificación. Estas tareas pertenecen a actividades de los procesos análisis, diseño, desarrollo y pruebas.

Por último, y antes de la aceptación del usuario, es preciso establecer un plan de pruebas de regresión que asegure la integración del sistema de información afectado.

## Revisiones periódicas

El monitoreo permanente del sistema asegura que las necesidades de mantenimiento sean identificadas y satisfechas cuando resulte necesario. Cuando el sistema es de uso prolongado, se puede establecer un mecanismo para recibir retroalimentación de los usuarios como una forma efectiva para determinar las necesidades de mantenimiento y modificación.

A los sistemas se les debe dar mantenimiento para asegurar que continúen operando en el nivel mostrado durante la etapa de prueba. Si los sistemas se deterioran, existe el riesgo de que no se desempeñen conforme a los estándares requeridos.

## Problemas comunes a la hora de hacer casos de prueba

### Demasiado específico: ejecute solo una condición de prueba específica

Los casos de prueba deben considerar una variedad de condiciones que se espera que maneje el software. El caso de prueba debe poder probar exhaustivamente el módulo de software con casi todas las combinaciones posibles de condiciones principales.

Para poder probar exhaustivamente todas las combinaciones de condiciones, el probador de software debe encontrar una manera de presentar estas condiciones de manera que sea fácil de seguir, revisar y modificar para otros si el proceso del mundo real exige tales acciones.

### Cubrir una pequeña parte de la funcionalidad: necesitan probar una parte más grande del sistema

Los casos de prueba a menudo se centran en una función específica. A menudo, esta función está determinada por el diseño técnico interno del software. Tales prácticas a menudo se encuentran en grandes aplicaciones monolíticas como SAP u Oracle ERP, donde un probador de software no siempre puede tener conocimiento de todo el proceso comercial, por lo que el caso de prueba nunca termina reflejando lo que el diseñador de prueba no sabe, pero debería tener. hizo el esfuerzo de comprender.

En cambio, los casos de prueba deben reflejar los patrones y flujos de uso. Cada caso de prueba debe tratar de cubrir la mayor parte del flujo que sea razonablemente posible, cruzando los límites técnicos y modulares de la aplicación.

### Prueba de software creada solo para un rol de usuario específico

A menudo hemos visto casos de prueba escritos para un rol de usuario muy específico, sin tener en cuenta a todos los demás usuarios de la aplicación. Esto limita el alcance de los casos de prueba y, por lo tanto, compromete significativamente su efectividad. Dichos casos de prueba prueban efectivamente solo pequeños elementos de la aplicación, mientras que engañosamente pretenden ser casos de prueba completos y sólidos.

Los casos de prueba que son más efectivos reflejan los patrones de uso, o lo que el mundo Agile denomina viajes de usuario. Una aplicación comercial, por ejemplo, debe probarse con casos de prueba diseñados para probar todo el proceso comercial, cubriendo todos los roles de usuario y todos los sistemas que podrían estar involucrados en el proceso comercial.

### Escrito para demostrar que los casos de uso más comunes están bien cubiertos en la aplicación

Esto es uno de los problemas más comunes y es el resultado de lo que yo llamo un enfoque 'perezoso' para el diseño de pruebas. El diseñador de pruebas simplemente convierte el documento de requisitos en casos de prueba.

En cambio, el diseñador de la prueba debe buscar los "casos de esquina" o las "condiciones de contorno". La mayoría de los desarrolladores pueden escribir fácilmente código para los casos de uso más comunes. Los problemas surgen en el momento en que hay una condición que es incluso ligeramente diferente al caso de uso más común o previsto. Un caso de prueba bien diseñado los detectará de manera fácil y consistente.

### Catalogación de casos de prueba y control de versiones

Cualquier caso de prueba puede volverse completamente inútil si no se cataloga sistemáticamente y se mantiene disponible para su uso. Imagine una biblioteca con libros no catalogados y colocados sistemáticamente en estantes, especialmente después de que varios prestatarios se hayan saciado. Sería imposible usar los libros si no puede encontrarlos con facilidad cuando los necesita.

A menudo, cientos de casos de prueba se escriben con mucho esfuerzo y luego se vuelcan en una estructura de carpetas compartidas. Si bien esto puede funcionar si tiene muy pocos casos de prueba, un sistema mal organizado colapsa en el momento en que la cantidad de casos de prueba aumenta más allá de un puñado.

Por lo tanto, necesita una herramienta de prueba de software que pueda etiquetar y catalogar sistemáticamente los casos de prueba. Luego, su herramienta de prueba de software debería poder "sacar" casos de prueba cuando sea necesario ejecutarlos. Para que todo este proceso sea fluido en todo el equipo de pruebas de software, necesita una poderosa herramienta de pruebas que pueda crear y mantener sin esfuerzo múltiples versiones de casos de prueba.

|  |
| --- |
| **¡Pro tip alert!**  Mejorar la eficiencia de los casos de prueba no es un término simplemente definido, sino que es un ejercicio y se puede lograr a través de un proceso maduro y práctica regular.  El equipo de pruebas no debe cansarse de involucrarse en la mejora de tales tareas, ya que es la mejor herramienta para lograr mayores logros en el mundo de la calidad, esto está probado en muchas de las organizaciones de prueba en todo el mundo en proyectos de misión crítica y aplicaciones complejas. |

# 

# Escenarios de análisis

Te presentamos varios escenarios de análisis en donde veremos aplicados algunos de los conceptos que aprendimos hoy. Te invitamos a que tomes algunos minutos de reflexión con tus compañeros de equipo y discutas cada escenario y su respuesta. Cada uno deberá exponer las razones por las que elige cada opción. Podrán coincidir o no entre ustedes, no se preocupen, intenten llegar a una opinión unánime, pero de no hacerlo cada uno enviará sus propias respuestas.

En el [formulario de fin de guía](https://forms.gle/69ukFQp5JSu6YiZJ6) encontrarás estos mismos escenarios y sus respuestas, envíenlas de forma individual luego de haberlo discutido.

1. Javier es tester. Está convencido que puede realizar un buen caso de prueba para rastrear bugs. Luego, una vez resueltos, hará una prueba más exhaustiva. De esta forma se asegura terminar con todos los bugs existentes en esa funcionalidad.
   1. Si bien la intención de Javier es buena, no es una meta realista terminar con todos los bugs.
   2. Si Javier quiere terminar con todos los bugs, no debe hacer pruebas más exhaustivas sino pruebas distintas.
   3. De acuerdo a la Paradoja del pesticida, al hacer distintas pruebas y más exhaustivas encontraremos más bugs pero sobrevivirán aquellos más fuertes, resistentes a los "pesticidas"
   4. Javier logrará su cometido de esa forma
2. Laura está escribiendo los pasos de un caso de prueba y se ve de esta forma: 1-Ingresar a la URL 2-Utilizar el motor de búsqueda para hallar el producto "X" 3-Seleccionarlo, realizar la compra y pagarlo.
   1. Es correcto y fácilmente comprensible.
   2. El paso 3 de Laura es compuesto. Debería dividirlo en instancias menores.
   3. No cumple con las buenas prácticas pero no es incorrecto
   4. Al ser un caso tan sencillo, puede dejarlo de esa forma
3. Gonzalo está armando sus casos de prueba. Tiene una cita mañana con recursos humanos para solicitar que contraten a otro Tester ya que su diseño incluye una cobertura del 100%. Esta convencido que le aprobarán la solicitud debido a que esta cobertura garantizará una excelente calidad en las pruebas.
   1. Recursos humanos rechazará su petición debido a que debería diseñar pruebas con más criterios que sólo la cobertura.
   2. Recursos humanos aprobará su pedido ya que asegurar la calidad es el objetivo.
   3. Recursos humanos aprobará su pedido pero deberá reportar luego esa calidad de las pruebas.
   4. Recursos humanos rechazará su pedido ya que debería diseñar pruebas que él sólo pueda ejecutar.
4. Raquel esta en la fase de recopilar datos de pruebas. Tiene casos positivos y negativos por ejecutar para un mismo requerimiento. Para ahorrar tiempo, recopilará sólo los datos del escenario positivo.
   1. Es suficiente recopilar información de ese manera, ya que al tratarse de un mismo requerimiento no precisa de ambos datos.
   2. Debería sólo recuperar los datos del escenario negativo para garantizar que el sistema funcione sólo de la manera esperada.
   3. Debería recuperar datos de ambos casos, pero sólo si el caso de prueba falla.
   4. Debería recopilar datos de ambos casos, ya que tener un enfoque de casos positivos y negativos no sólo nos brinda información de que el sistema funciona en un escenario positivo, sino que también garantizamos que no funcione en escenarios negativos.
5. Juan ha reportado que al probar un inicio de sesión, el sistema permite logearse tanto si introduces la contraseña en minúsculas o mayúsculas. El administrador deberá clasificar este fallo como un mantenimiento:
   1. Adaptativo, ya que se deberá hacer una gestión de la base de datos.
   2. Correctivo, ya que es un error del producto
   3. Evolutivo, ya que es un cambio de adaptación a un nuevo requerimiento de usuario.
   4. Perfectivo, ya que es una mejora en el sistema.

# 

# Ejercicios

### Ejercicio #1

De las siguientes afirmaciones: ¿cuáles son verdaderas y cuáles falsas? Marcalas en la siguiente tabla, puedes utilizar [esta plantilla](https://docs.google.com/document/d/1gyrrGKaTYQpG8dTLTQEepqiFwQZCNgSQk0d0raiQbqI/edit?usp=sharing). Luego compara las respuestas con tu equipo.

|  |  |  |
| --- | --- | --- |
| **Afirmaciones** | **V** | **F** |
| La prioridad de un caso de prueba puede ser Crítico, Mayor o Menor |  |  |
| La cobertura de pruebas me permite indicar cuando probamos lo suficiente |  |  |
| Las precondiciones son las situación inicial o previa a la ejecución de pruebas |  |  |
| Todos los casos de prueba tienen una misma cantidad de pasos a seguir |  |  |
| La trazabilidad no ayuda a determinar la cobertura de los requisitos |  |  |
| El oráculo es lo que nos dice que resultados tenemos que esperar de determinadas entradas y condiciones de ejecución |  |  |
| La duración es el tiempo en reloj en ejecutar el caso de prueba |  |  |
| Las evidencias son los resultados esperados de los casos de prueba |  |  |
| El ID (Identificador) puede repetirse en los casos de prueba |  |  |
| Al diseñar las pruebas, todos los bugs deben ser descubiertos y ninguno sobrevive |  |  |
| El diseño de casos de pruebas en un proceso del cual no se puede llevar control |  |  |

### Ejercicio #2

Una empresa está desarrollando una página web, y te contrata como Tester para realizar las pruebas de la siguiente página:

* <http://demo.testim.io/>

Ingrese al siguiente link: <http://demo.testim.io/> (es un sitio armado para pruebas) y realiza lo siguiente:

1. Analiza el sistema, recórrelo e identifique qué funcionalidades que posee
2. Arma una planilla para cargar los casos de prueba
3. Carga los casos de prueba en la planilla
4. Corre los casos de prueba y guarda evidencia de sus resultados. (La evidencia pueden ser capturas de pantalla, GIFs, etc)
5. Comprueba con tu equipo los casos desarrollados y los errores hallados.
6. Arma en equipo una grilla más completa entre todo el equipo

Modelo de un caso de prueba:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Id Caso de Prueba | Caso de Prueba  (Objetivo) | Precondiciones (Requisitos previos) | Procedimiento (Pasos para el caso de prueba) | Datos de prueba | Resultados Esperados | Resultado obtenido | Evidencias |
|  |  |  |  |  |  |  |  |

Puedes utilizar esta [plantilla de ejemplo](https://docs.google.com/document/d/1wz4iZcVEP9ojOeImmb6gYnuvtilqG9q9C4mD-_RZOmc/edit?usp=sharing) para realizar el ejercicio.