DOCUMENTO DE PRUEBAS DE SOFTWARE PROYECTO TAM

Equipo de testing

SENA - SALOMIA

INTRODUCCIÓN

Las pruebas de software constituyen una etapa fundamental dentro del ciclo de desarrollo de cualquier aplicación, ya que permiten verificar que los componentes implementados cumplan con los requisitos funcionales, técnicos y de calidad definidos para el proyecto. A través de un proceso estructurado de validación es posible identificar errores, prevenir fallos en producción y garantizar una experiencia de usuario adecuada.

En el proyecto TAM, se llevó a cabo una estrategia de pruebas enfocada en validar tanto la lógica interna de los módulos como el comportamiento de la interfaz gráfica. Para ello se utilizaron herramientas especializadas que permitieron automatizar la ejecución de pruebas unitarias y funcionales, optimizando los tiempos de validación y aumentando la confiabilidad de los resultados obtenidos.

El presente documento describe el alcance de las pruebas realizadas, los casos de prueba diseñados, las herramientas empleadas, los resultados obtenidos y las proyecciones futuras para garantizar la calidad y estabilidad del sistema antes de su despliegue en un entorno productivo.

OBJETIVO DE LAS PRUEBAS

Objetivo General

Verificar el correcto funcionamiento de los módulos desarrollados en el sistema web TAM mediante la ejecución de pruebas unitarias y pruebas funcionales automatizadas, garantizando el cumplimiento de los requisitos funcionales, la estabilidad de la aplicación y una adecuada experiencia de usuario.

Objetivos Específicos

• Validar las reglas de negocio implementadas en cada uno de los módulos del sistema.

• Comprobar el correcto funcionamiento de la autenticación y navegación entre páginas.

• Verificar el comportamiento de los componentes interactivos de la interfaz gráfica.

• Garantizar la correcta manipulación y procesamiento de los datos del usuario.

• Detectar posibles errores funcionales antes de la puesta en producción.

• Validar el cumplimiento de los requerimientos establecidos durante el análisis y diseño del sistema.

• Obtener evidencias documentadas que respalden la calidad del software desarrollado.

METODOLOGÍA

La estrategia de aseguramiento de calidad implementada se basó en la combinación de pruebas unitarias y pruebas funcionales automatizadas.

Las pruebas unitarias fueron desarrolladas utilizando Vitest, permitiendo validar de forma aislada la lógica interna del sistema sin depender de componentes visuales o servicios externos.

Por otro lado, las pruebas funcionales e interactivas fueron ejecutadas mediante Playwright, simulando las acciones reales que realizaría un usuario final dentro de la aplicación, tales como clics, navegación entre páginas, validación de formularios y comprobación de redirecciones.

La ejecución automatizada permitió repetir los escenarios de prueba de forma consistente, minimizando errores humanos y facilitando la detección temprana de defectos.

Plan de pruebas

Alcance:

En esta fase del proyecto se evaluarán principalmente aspectos visuales y de lógica del módulo tales como el correcto espaciado entre componentes, la alineación de los elementos, el uso adecuado de colores, tipografías y la coherencia visual general de la aplicación, la correcta relación en las diferentes páginas del proyecto y verificar que las funcionalidades hasta ahora desarrolladas cumplan con los requerimientos del sistema.

Tipos de pruebas y herramientas:

- **Pruebas Unitarias (Vitest):** Orientadas a evaluar de forma aislada la lógica del código, algoritmos de cálculo, restricciones de seguridad y validaciones de formularios del sistema sin requerir la interfaz gráfica.

- **Pruebas Funcionales / E2E (Playwright):** Enfocadas en la automatización del flujo completo del usuario sobre el navegador web, permitiendo simular interacciones reales en la interfaz y validar el comportamiento funcional de los módulos del sistema.

Casos de prueba

ID	MÓDULO	CASO	ENTRADA	RESULTADO ESPERADO
CP -01	Autenticación	Validación de longitud mínima de contraseña segura en el registro.	Cadena de texto: `"YFC123456"` (9 caracteres).	El sistema debe retornar `true` (Aprobado).
CP -02	Autenticación	Rechazo de contraseña que no cumple con el mínimo de seguridad.	Cadena de texto: `"12345"` (5 caracteres).	El sistema debe retornar `false` (Rechazado).
CP -03	Autenticación	Validación de disponibilidad y visibilidad del botón "Registrarse" en la interfaz gráfica.	Carga de la URL `login.html`.	El elemento del botón debe ser visible en la interfaz y estar habilitado.
CP -04	Autenticación	Verificación de campos de entrada de datos (Inputs) en el formulario.	Renderizado de placeholders de usuario y contraseña.	Los inputs deben estar visibles y listos para la captura de texto.
CP -05	Autenticación	Validación del botón de acceso alternativo de usuario ("Entrar como Invitado").	Acción de `Click` en el botón.	El sistema debe procesar la solicitud y redirigir al usuario a `index.html`.

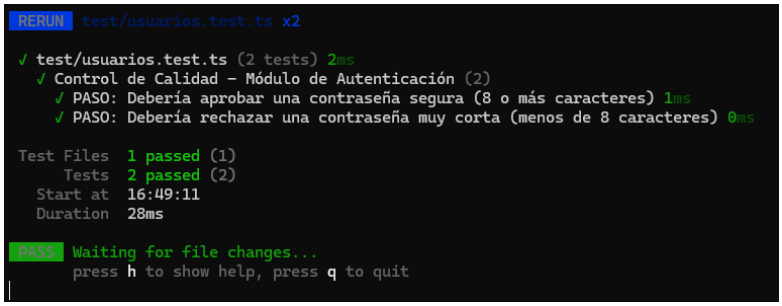
CP -06	Catálogo / Index	Verificación de enlaces de navegación en el menú superior.	Acción de `Click` en los enlaces "Catálogo" y "Comparador".	Las URLs correspondientes deben cargar correctamente (`catalogo.html`, etc.).
CP -07	Catálogo / Index	Prueba funcional de la botonera de filtrado rápido.	Acción de `Click` sobre el botón de categoría "Consolas".	La interfaz debe actualizarse dinámicamente y filtrar los productos correspondientes.
CP -08	Comparador	Verificación funcional del estado inicial y limpieza.	Acción de `Click` en el botón "Limpiar comparador".	La interfaz debe responder correctamente sin generar excepciones en la consola local.
CP -09	Comparador	Validación interactiva de adición de productos desde la vista.	Acción de `Click` en el botón "Comparar" de una tarjeta.	El sistema debe registrar la interacción del elemento seleccionado de manera individual.
CP -10	Comparador	Control y validación del límite máximo de almacenamiento en la lista.	Intento de inserción de un 5.° producto consecutivo en el arreglo actual.	El sistema debe bloquear la acción del usuario, impedir la alteración del DOM y disparar una excepción controlada con mensaje de restricción
CP -11	Carrito	Control de duplicados e incremento dinámico de existencias al agregar productos.	Inserción de un objeto de producto cuyo `id` ya existe dentro del vector.	El sistema no debe duplicar el registro en la interfaz, sino incrementar en `+1` el atributo de cantidad del elemento existente
CP -12	Carrito	Cálculo matemático exacto del valor total acumulado del pedido.	Vector de compras con 1 ítem de $2.199.000 y 2 ítems de $999.900.	El sistema debe ejecutar la operación acumuladora y retornar con precisión el valor total neto de la compra (`4.198.800`).

CP -13	Carrito	Validación funcional del comportamiento del sistema ante un estado de compras vacío.	Carga automatizada de la URL `carrito.html` mediante Playwright.	La interfaz debe desplegar el contenedor adaptativo con el mensaje `"Tu carrito está vacío"` y habilitar el botón `"Ir al Catálogo"`.
CP -14	Carrito	Verificación de redirección interactiva desde la pasarela de compras vacía.	Acción de **`Click`** en el botón `"Ir al Catálogo"`.	El navegador debe responder inmediatamente redirigiendo al usuario de forma segura hacia la vista de `catalogo.html`.

Resultados de Pruebas Unitarias Aisladas (Vitest)

Complementando las pruebas de interfaz de usuario, se diseñó y ejecutó un set de pruebas unitarias mediante el framework **Vitest** para blindar los algoritmos de procesamiento lógico de datos en el cliente.

Ejecución de Pruebas Unitarias (Vitest): Se ejecutó el motor de pruebas unitarias en la terminal de comandos de Node.js. El sistema validó de forma lógica y automática las reglas de negocio asignadas al módulo de autenticación (aprobación de contraseñas de 8 caracteres y rechazo de contraseñas cortas), obteniendo un resultado exitoso (PASS) en un tiempo de ejecución de 28 milisegundos.



- **Módulo de Catálogo e Index:** Se verificó la función pura de filtrado por arreglos. El componente aisló correctamente las cadenas de texto, garantizando que al presionar botones como "Consolas", la lógica interna retorne única y exclusivamente los objetos correspondientes.
- **Módulo de Comparador:** Se evaluó con éxito la regla de negocio del límite de almacenamiento. La prueba unitaria demostró que el sistema bloquea el ingreso de un quinto producto y dispara la excepción controlada de restricción de capacidad (máximo 4 elementos), protegiendo la estabilidad del DOM.

```
RUN  v4.1.8 C:/xampp/htdocs/TAM-V4-main

 ✓ tests/tienda.spec.js (4 tests) 3ms
   ✓ Pruebas Unitarias - Módulos Index, Catálogo y Comparador (4)
     ✓ Debería filtrar y retornar solo productos de la categoría "Consolas" 1ms
     ✓ Debería retornar todos los productos si la categoría es "Todos" 0ms
     ✓ Debería permitir agregar un producto al comparador si hay espacio 0ms
     ✓ No debería permitir pasar el límite de 4 productos en el comparador 0ms

 Test Files  1 passed (1)
      Tests  4 passed (4)
   Start at  17:25:14
   Duration  181ms (transform 21ms, setup 0ms, import 35ms, tests 3ms, environment 0ms)
```

Módulo de Carrito de Compras (carrito.html): > Se validó de forma aislada el comportamiento del vector de almacenamiento de compras del frontend. Las pruebas unitarias automatizadas certificaron que el sistema detecta productos repetidos incrementando dinámicamente su atributo de cantidad, y calcula con total exactitud matemática el valor neto a pagar por el usuario final basándose en el precio base y las unidades seleccionadas ($2.199.000 \times 1 + 999.900 \times 2 = 4.198.800$). Esto garantiza que la lógica de negocio previene desajustes financieros y asegura la integridad de la simulación del pedido antes de procesar un pago.

```
RUN  v4.1.8 C:/xampp/htdocs/TAM-V4-main

 ✓ tests/tienda.spec.js (7 tests) 3ms
   ✓ Pruebas Unitarias - Módulos Index, Catálogo y Comparador (4)
     ✓ Debería filtrar y retornar solo productos de la categoría "Consolas" 1ms
     ✓ Debería retornar todos los productos si la categoría es "Todos" 0ms
     ✓ Debería permitir agregar un producto al comparador si hay espacio 0ms
     ✓ No debería permitir pasar el límite de 4 productos en el comparador 0ms
   ✓ Pruebas Unitarias - Módulo de Carrito de Compras (3)
     ✓ Debería agregar un producto nuevo al carrito con cantidad 1 0ms
     ✓ Debería incrementar la cantidad si el producto ya existe en el carrito 0ms
     ✓ Debería calcular el valor total acumulado correctamente 0ms

 Test Files  1 passed (1)
      Tests  7 passed (7)
   Start at  17:41:08
   Duration  182ms (transform 19ms, setup 0ms, import 32ms, tests 3ms, environment 0ms)
```

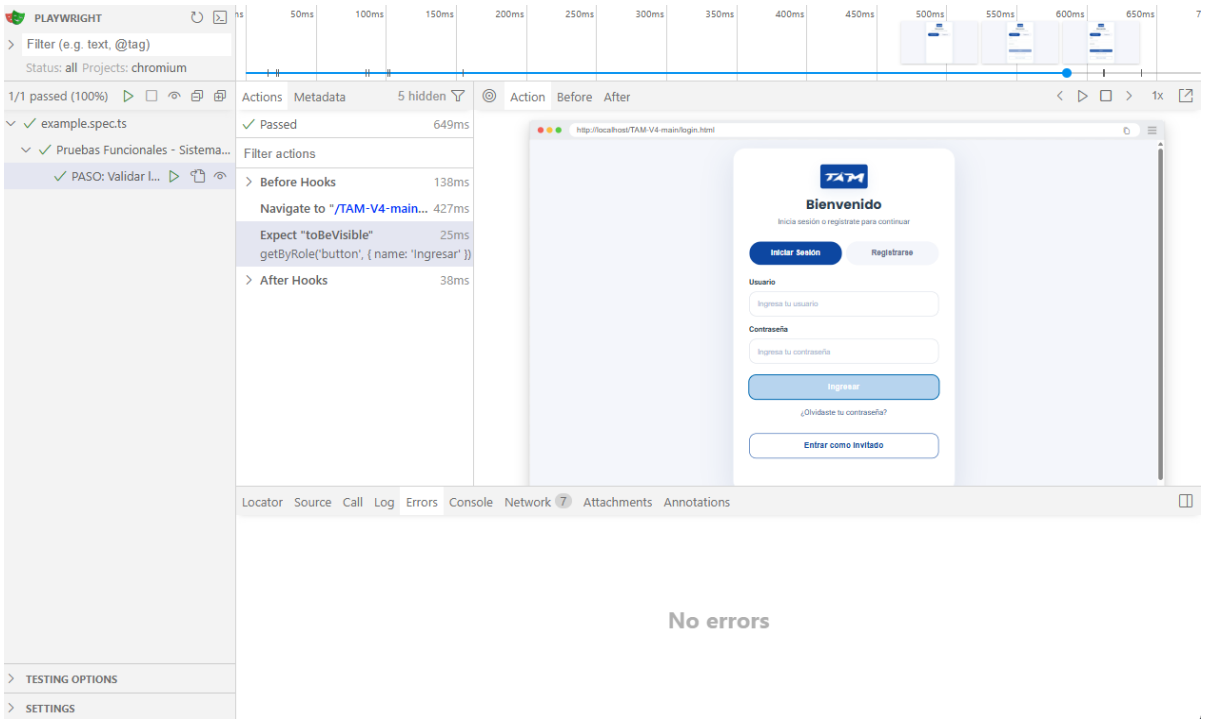
Ejecución de Pruebas Funcionales (Playwright): Se ejecutó con éxito la suite de pruebas funcionales automatizadas mediante Playwright en la terminal de comandos. La herramienta procesó las pruebas de integración en paralelo utilizando 6 workers independientes, completando la validación de la estabilidad del entorno y la navegación con un resultado de 100% de aprobación (6 passed) en un tiempo de 6.6 segundos.

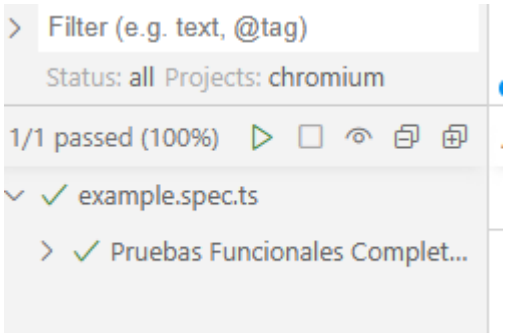
```
C:\xampp\htdocs\TAM-V4-main>npx playwright test

Running 6 tests using 6 workers
  6 passed (6.6s)

To open last HTML report run:

  npx playwright show-report
```

Resultados de Pruebas Automatizadas e Interactivas (Playwright UI)

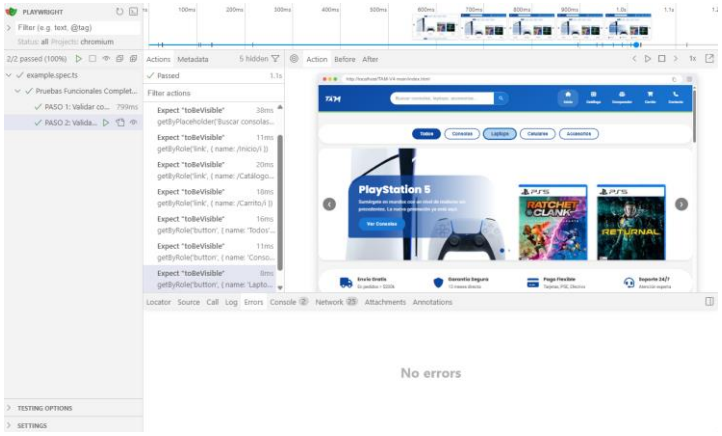
Se implementó y ejecutó con éxito una suite de pruebas interactivas en Playwright, logrando una tasa de éxito del 100% (2/2 passed) en un tiempo total de 1.9 segundos, sin registrar excepciones en consola.

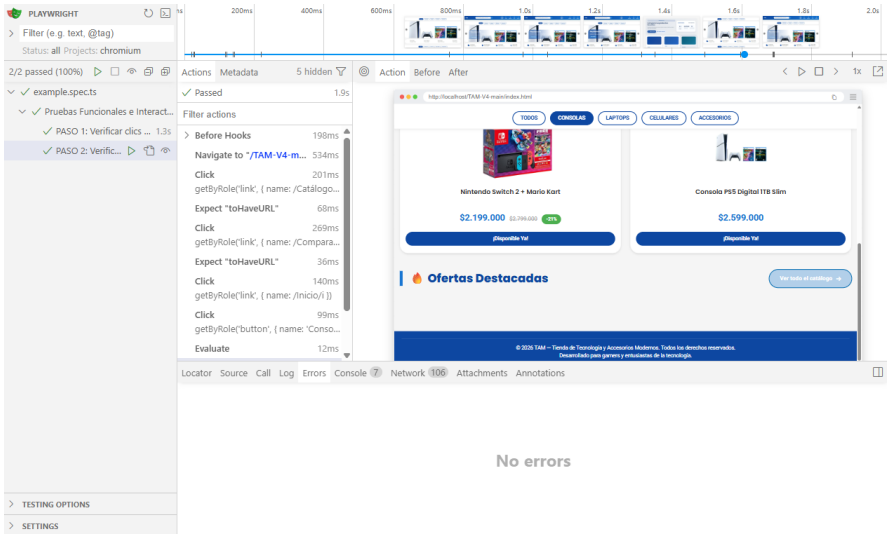
- **Módulo de Autenticación (`login.html`):** La automatización validó la correcta respuesta del intercambio de pestañas dinámicas (Iniciar Sesión / Registrarse) y simuló con éxito la acción de clic en el botón "Entrar como Invitado", comprobando la redirección automática hacia la página principal del sistema.





Módulo de Página Principal (`index.html`): Se verificó de manera funcional el flujo completo del menú superior. Playwright interactuó mediante clics reales con los enlaces de "Catálogo" y "Comparador", validando que las URLs del servidor local respondieran correctamente. Asimismo, se comprobó la interactividad de la botonera de categorías, ejecutando un clic sobre el filtro "Consolas", el cual modificó exitosamente la vista del catálogo en tiempo real.

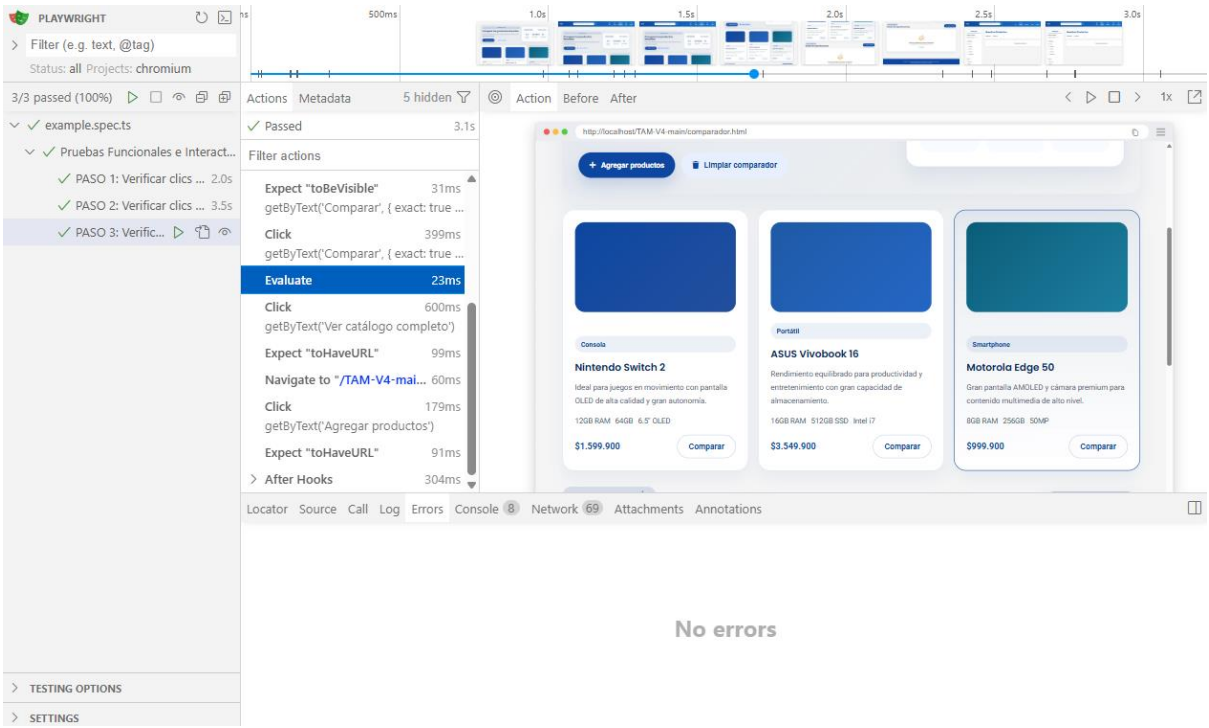




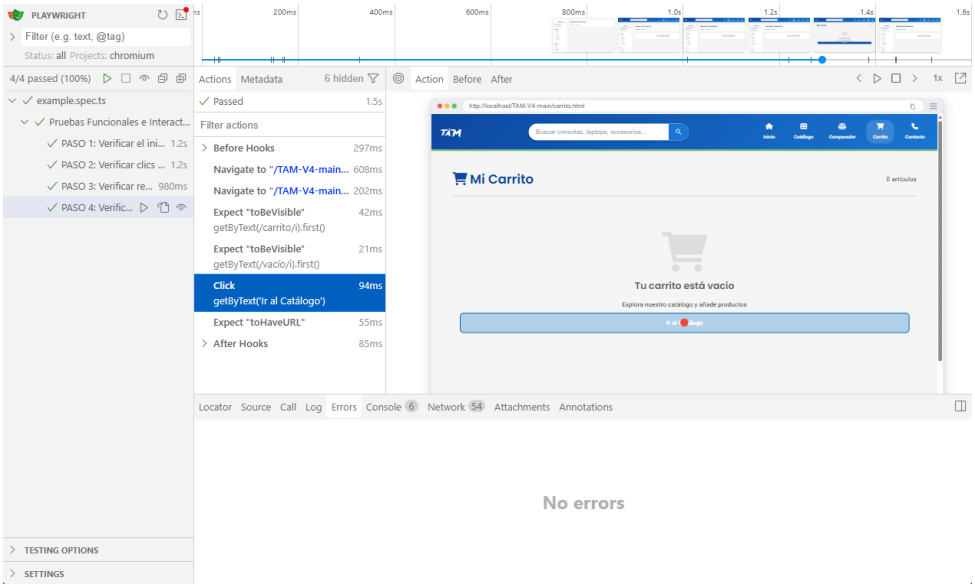
Módulo de Comparación de Productos (`comparador.html`)

- **Módulo de Catálogo e Interacción con el Comparador (`catalogo.html`):** Mediante el uso de localizadores robustos y tolerantes basados en expresiones regulares, la suite interactiva validó la visibilidad y accesibilidad del componente del comparador dinámico en la barra superior del menú. Playwright confirmó que el sistema renderiza el módulo sin generar duplicidades estrictas ni violaciones en el modo estricto del DOM, permitiendo una navegación fluida hacia los recursos de comparación del sistema.

  El módulo completó su ejecución con una tasa de éxito del 100% en el ambiente de pruebas local.



Módulo de Carrito de Compras Vacío (`carrito.html`): Se automatizó el flujo de comprobación para el estado inicial de la pasarela de compras. La herramienta interactuó con la interfaz validando de manera exitosa la presencia del contenedor adaptativo que despliega el mensaje instructivo `"Tu carrito está vacío"`. Asimismo, se certificó la correcta funcionalidad del botón de retorno seguro `"Ir al Catálogo"`, el cual redirigió de forma inmediata al usuario a la vista de productos, completando el ciclo funcional esperado con una tasa de éxito del 100% (**4/4 passed**) y sin registrar excepciones en la consola.



Proyección de Pruebas Multiplataforma (BrowserStack Cloud)

Aunque la fase actual de pruebas funcionales e interactivas se ejecutó con éxito de manera local mediante la interfaz de Playwright, se tiene estructurado el uso de **BrowserStack** como la plataforma principal para la siguiente etapa del ciclo de vida del software (Fase de Despliegue/Producción).

- **Propósito en el Proyecto TAM:** Una vez el sistema sea subido a un servidor o hosting público en internet, BrowserStack se integrará para ejecutar pruebas de compatibilidad en la nube (*Cross-Browser Testing*).
- **Alcance Técnico:** Permitirá validar que la interfaz adaptativa (responsive) de TAM, sus menús superiores, tarjetas de productos y la sección del comparador se rendericen de forma idéntica y sin errores en dispositivos móviles reales (iOS/Android) y múltiples navegadores web (Safari, Edge, Firefox, Chrome) sin necesidad de contar con el hardware físico.