# Pruebas Auditoría Lighthouse

# **Información general**

## **Lighthouse**

Lighthouse es una herramienta que facilita la verificación del rendimiento de páginas Web progresivas, midiendo así su capacidad, rendimiento y otros factores para facilitarle al desarrollador un buen manejo de su página web.

## **Performance**

Es el que se encarga de verificar el rendimiento de carga de la página web, con ayuda de algunos conceptos tales como:

**First Contentful Paint**: indica cuánto tarda en mostrarse la primera imagen o el primer texto completo en la página web.

**First Meaningful Paint**: indica en qué momento se pueden visualizar íntegramente los contenidos principales de la página.

**Speed Index:** indica cuánto tardan los contenidos de la página web en visualizarse.

**Time To Interactive:** indica cuánto tarda la página en cargar con plena capacidad de interacción.

**First CPU Idle:** indica el momento en el que la actividad de los subprocesos principales es suficientemente reducida como para poder procesar entradas del usuario por primera vez.

**Estimated Input Latency:** indica una estimación de cuántos milisegundos necesita la página o aplicación web para reaccionar a una entrada del usuario en el margen de los cinco segundos con mayor carga mientras se descarga la página. Si la latencia es superior a 50 milisegundos, es muy frecuente que los usuarios consideren la página o app como lenta.

## **Accesibility**

Es la que se encarga de verificar si el aplicativo o página web es fácil de utilizar. De una u otra forma comprueba elementos clave en la página y determina si es sencillo e intuitivo para los usuarios que tengan limitaciones, o para los que no.   
  
 Por lo general se tiene en cuenta aspectos tales como el color, tamaño y función de los elementos. Sin embargo esta función ayuda al desarrollador a ver los elementos que no está necesitando o que simplemente sean de más en su aplicativo, para ello con esta función se citará el elemento en conjunto a su línea de código que no haga falta poner.

## **Best practices**

Esta función se encarga de analizar e implementar aspectos de seguridad con respecto a tu página teniendo en cuenta el uso de bibliotecas y otros recursos integrados a tú aplicativo con la finalidad de verificar de ver si vienen de una fuente medianamente segura teniendo en cuenta su uso. Para esto se tienen en cuenta algunos aspectos tales como el origen de Links, librerías desconocidas, entre otros errores de consola.

## **SEO**

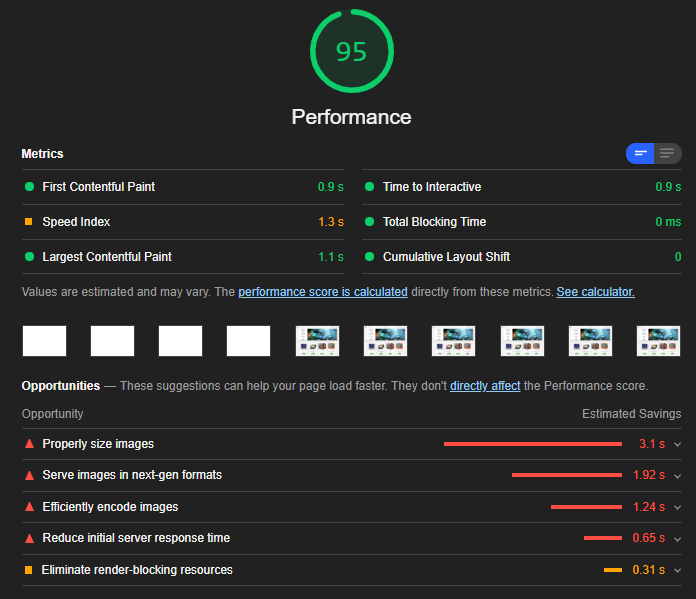
El SEO es el que se encarga de manejar el nivel de búsqueda hacia tu página, con base a esto se debe de tener en cuenta; La versión en donde estás elaborando internamente el aplicativo web, algo a destacar es que a esto se le tiene en cuenta un aspecto primordial como lo es el uso de palabras clave, mientras más detallado sean estas palabras, más se ampliarán las posibilidades de que los buscadores te posicionen mejor con respecto a lo que está buscando el usuario referente a tu aplicativo.   
 En definitiva, el SEO verifica y puntúa de 0 a 100 con respecto a que tan bien esté tu página o aplicativo con respecto a los aspectos ya mencionados anteriormente

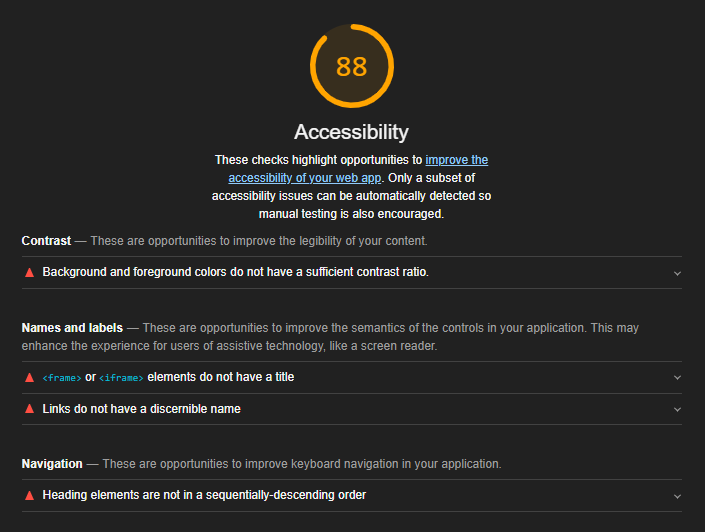
# **Capítulo 2**

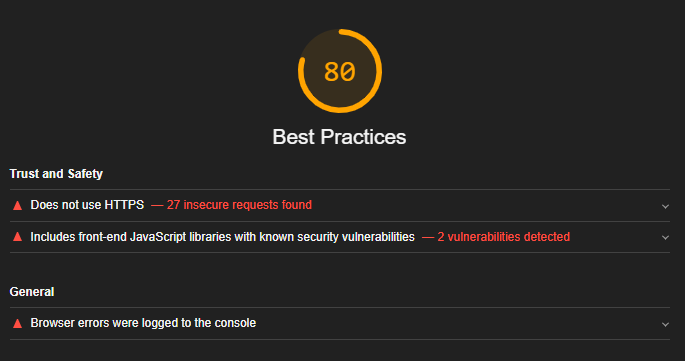
# **Auditoría ComponenTech**

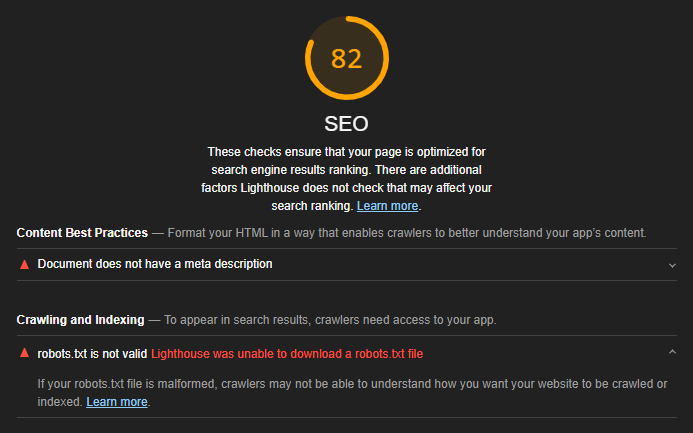
## **Página principal**

Esta es la primera puntuación que recibimos con respecto a nuestra interfaz inicial, la cual contiene inicialmente una lista de categorías, un espacio en donde están los productos. En las oportunidades de cambio, Lighthouse nos está sugiriendo utilizar imágenes menos pesadas, saber manejar los formatos de imagen para la implementación.

****

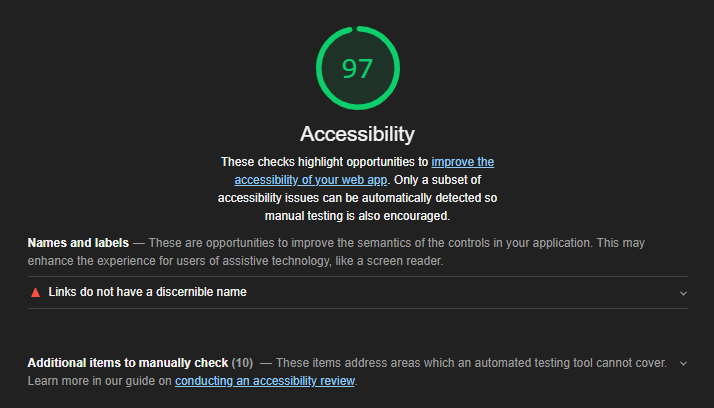
****

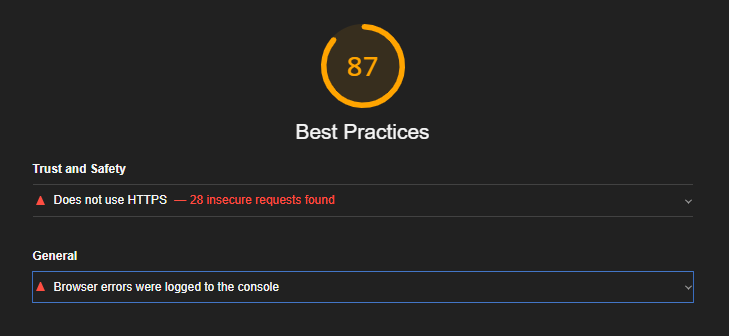
****

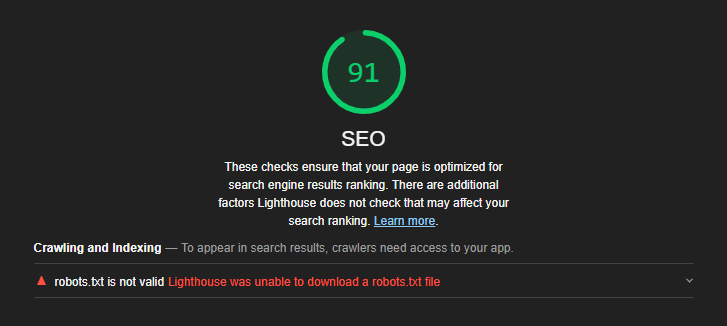


Después de aplicar los cambios



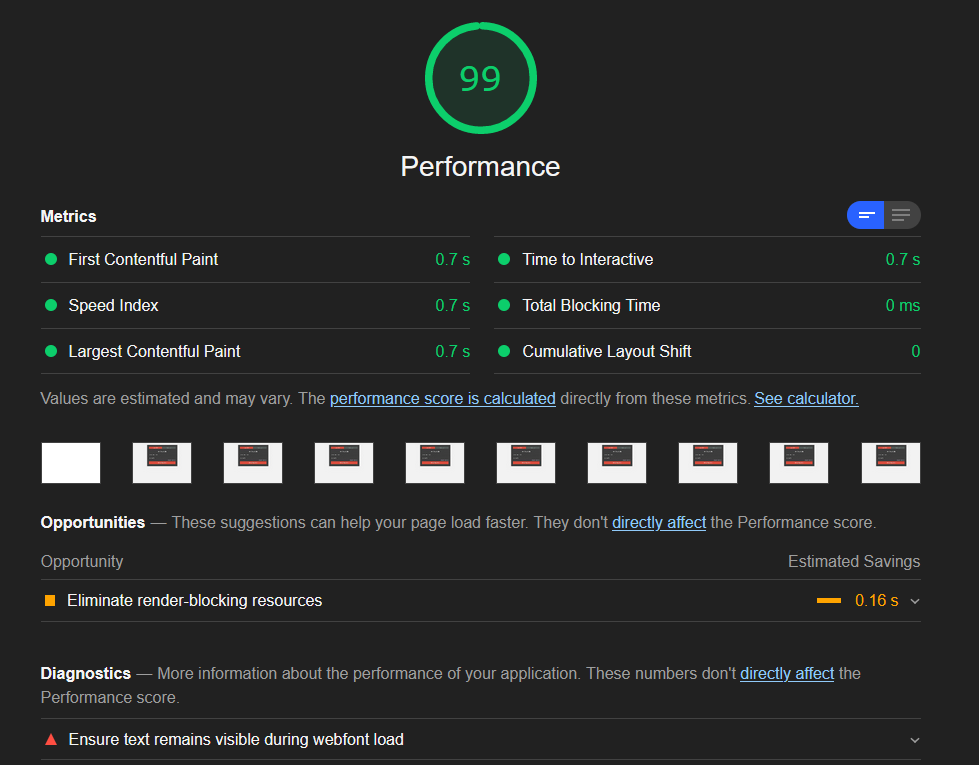


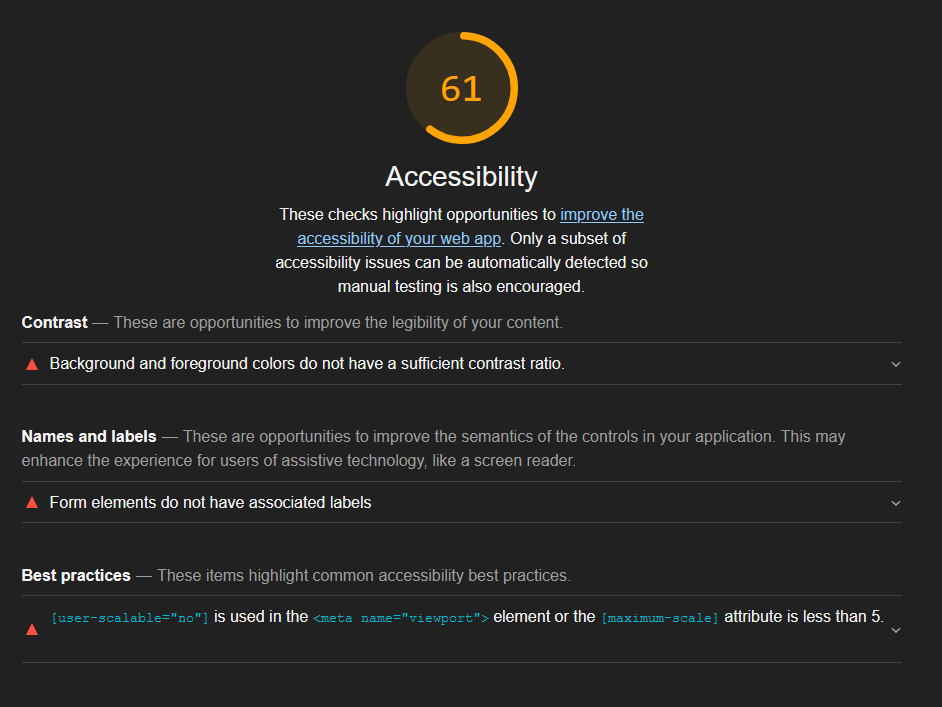


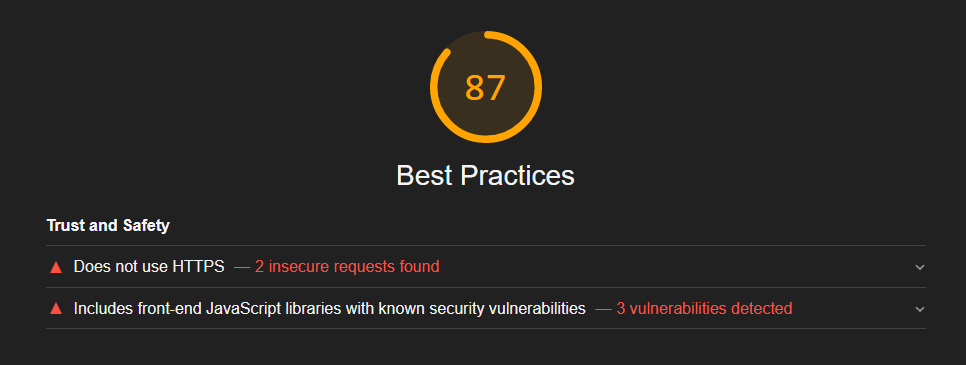


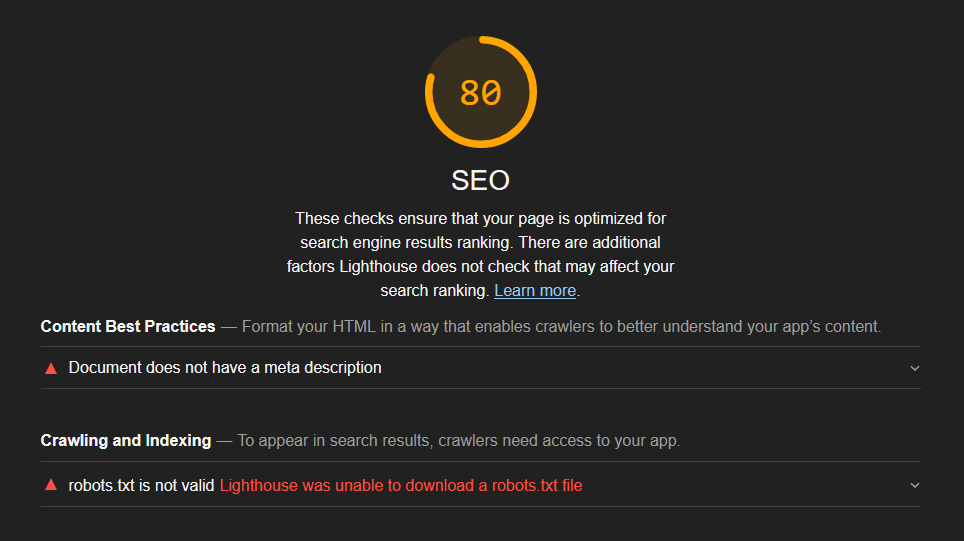
Aquí está la última puntuación que recibió la interfaz principal de nuestro aplicativo, en donde se le hicieron cambios respectivos tales como eliminar algunas librerías que no estaban en uso, disminuir el peso de algunas imágenes utilizadas, modificar la fuente de algunos elementos para que al usuario le sea más sencillo verlos.

## **Inicio de sesión y registro**



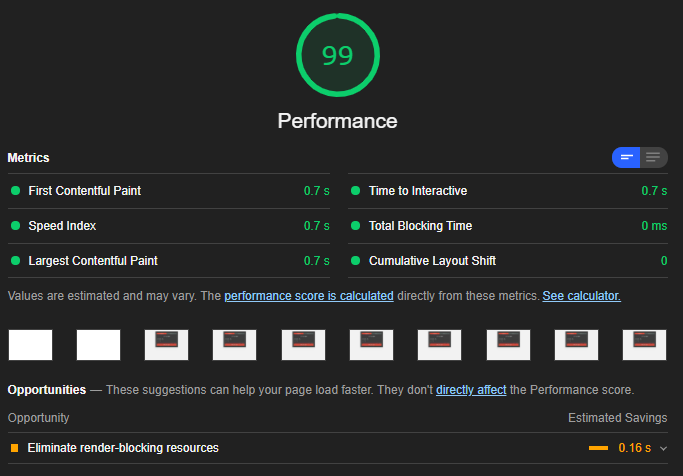


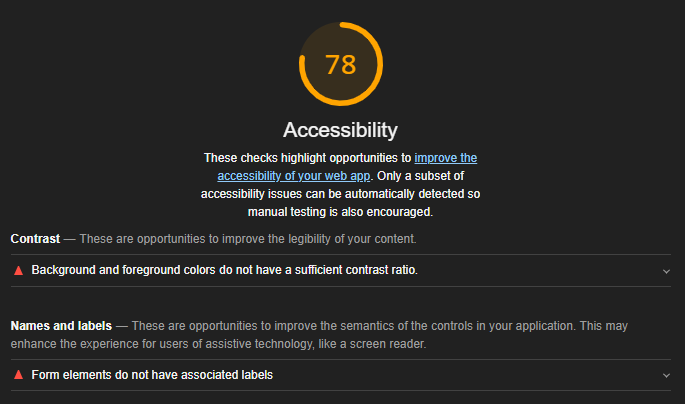


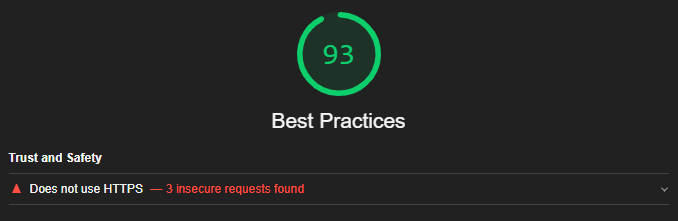


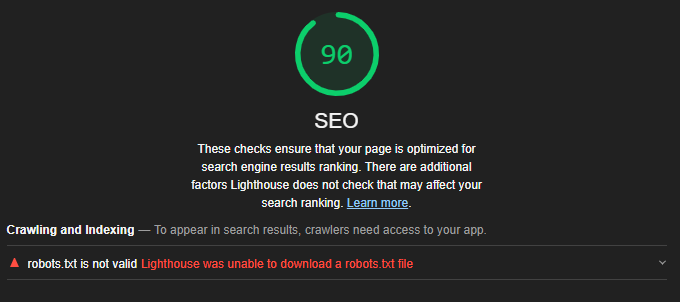
Esta es la puntuación que nos arrojó Lighthouse con la interfaz que le permite al usuario Iniciar sesión o registrarse si es que no tiene una cuenta registrada en el sistema Componen Tech.

Después de aplicar los cambios



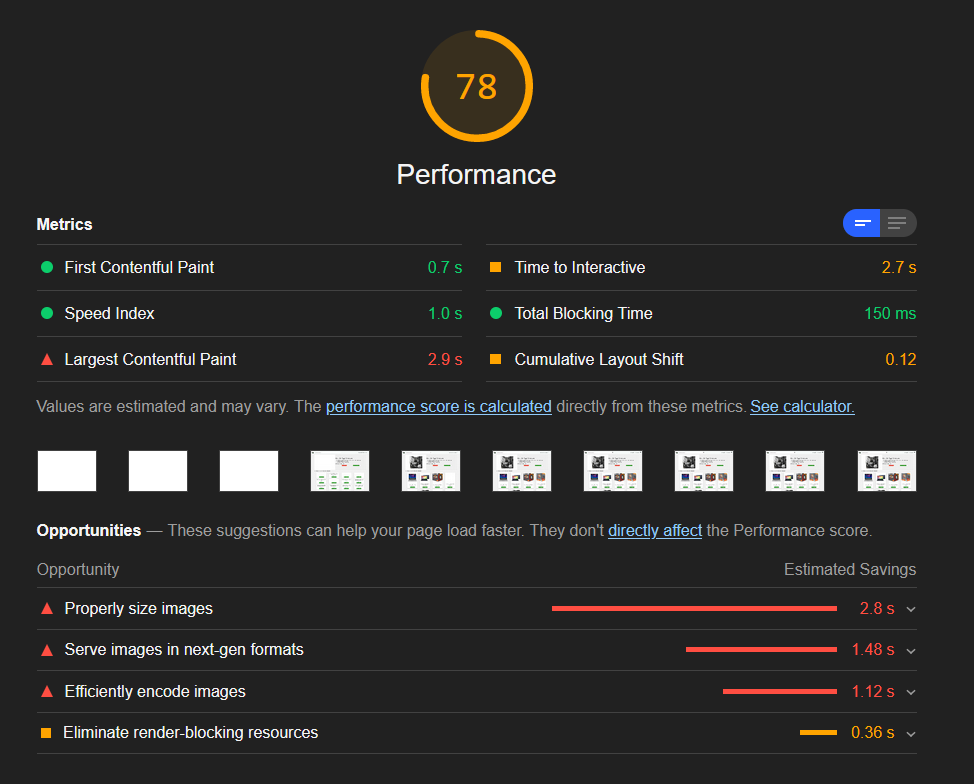


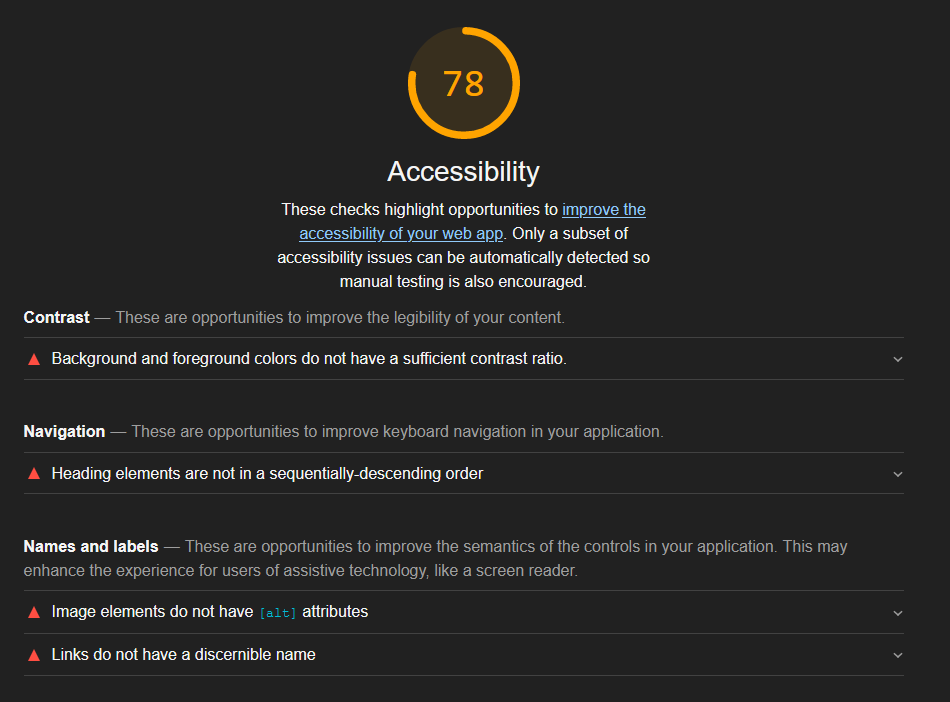


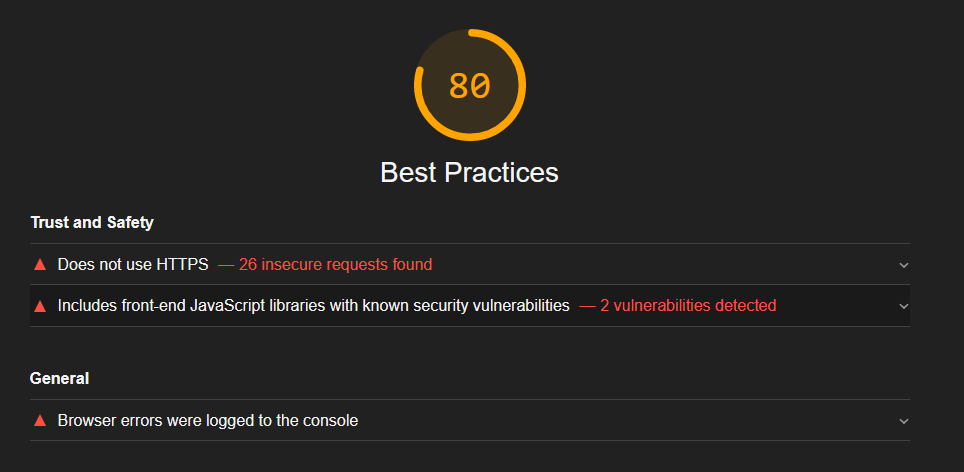


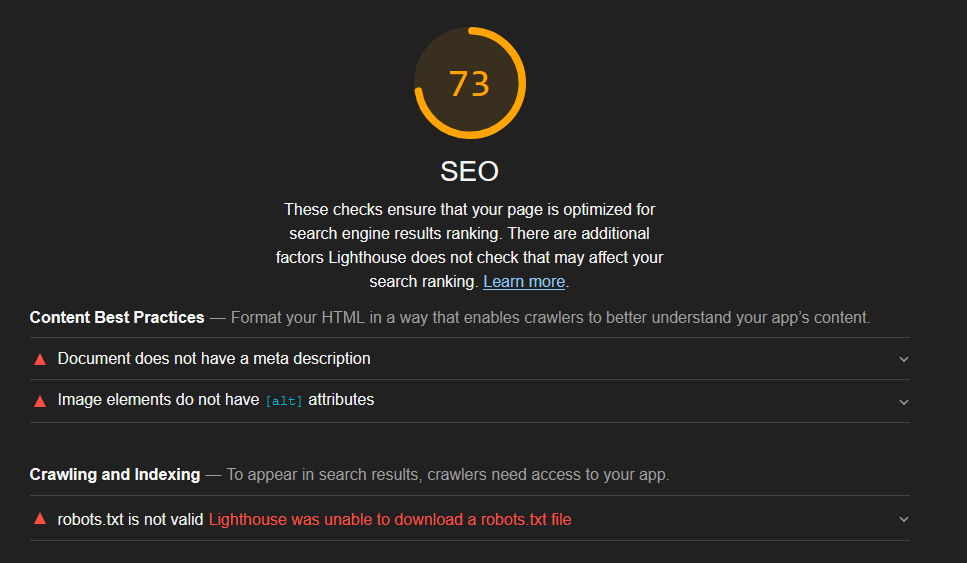
Finalmente, aquí está la puntuación de la misma interfaz con los cambios recomendados por la misma aplicación de auditoría, la cual mejora mucho más el tema del rendimiento, renderizado y otros aspectos que estuvo requiriendo en la primera evaluación.

## **Ver productos**



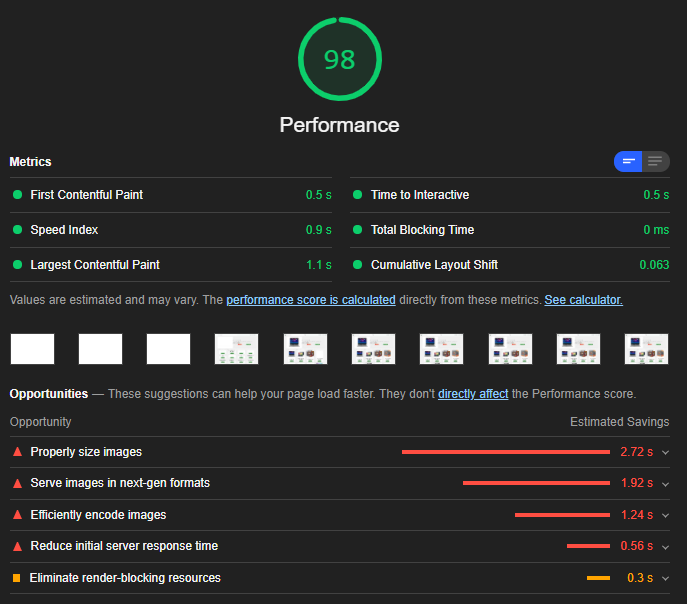


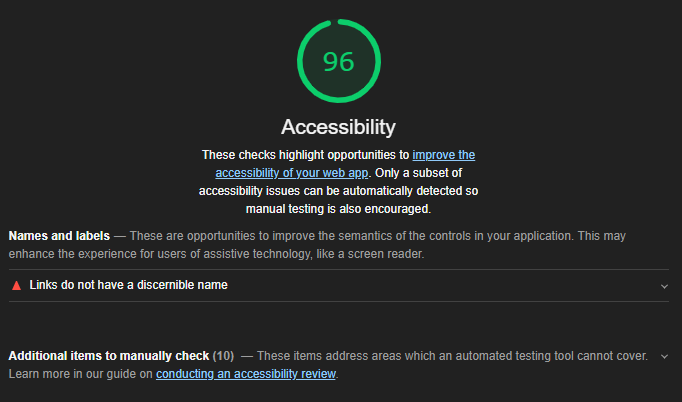


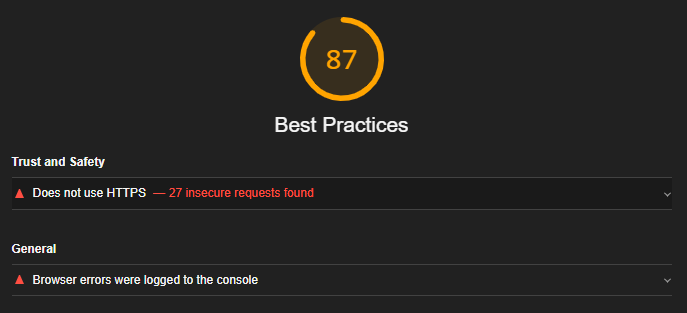


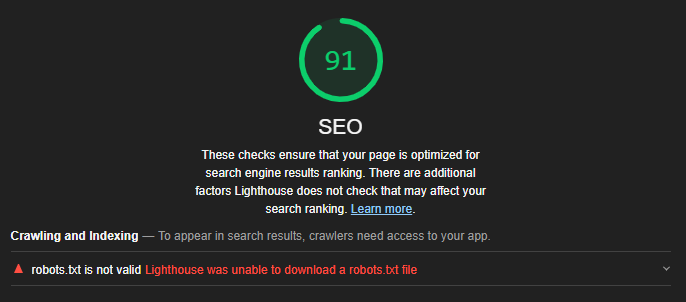
Esta es la puntuación que recibió la interfaz de vista previa de productos por medio de Lighthouse, por la cual los usuarios pueden ver productos que le llamen la atención.

Después de aplicar los cambios



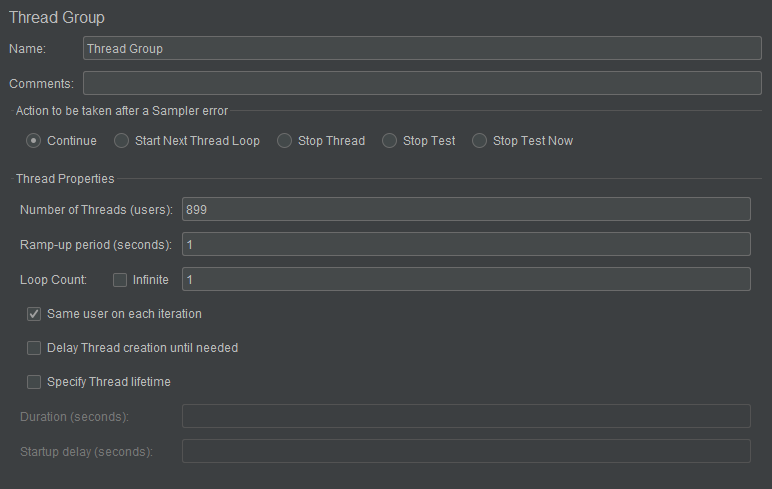


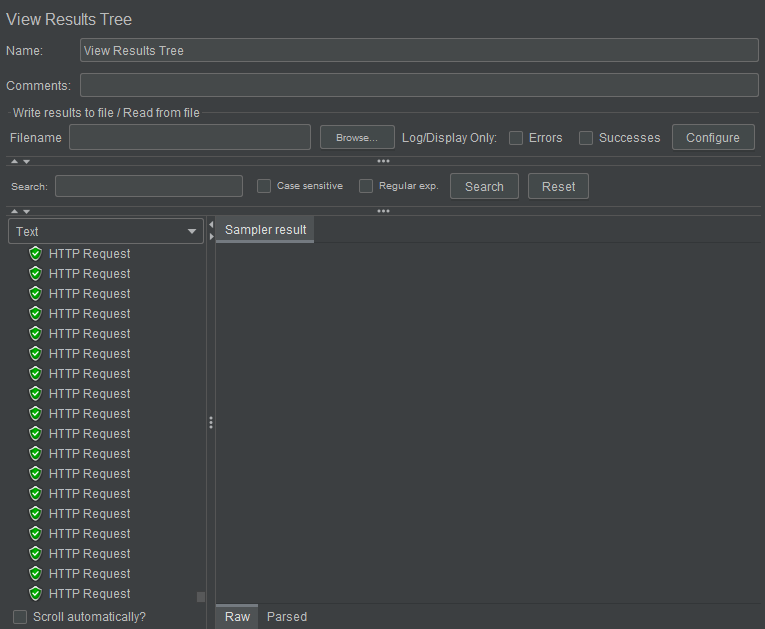




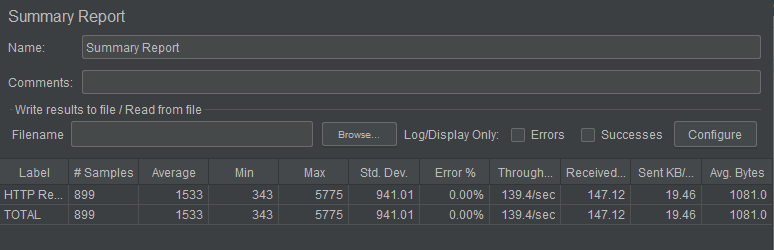
Aquí está la última evaluación que se le hizo a la misma interfaz con los cambios que nos requirió la auditoría de Lighthouse, en la cual nos pedía disminuir el peso de las imágenes, especificar algunas referencias de las rutas de imagen al igual que cambiarle un poco las dimensiones de las mismas.

Pruebas de Carga y Stress

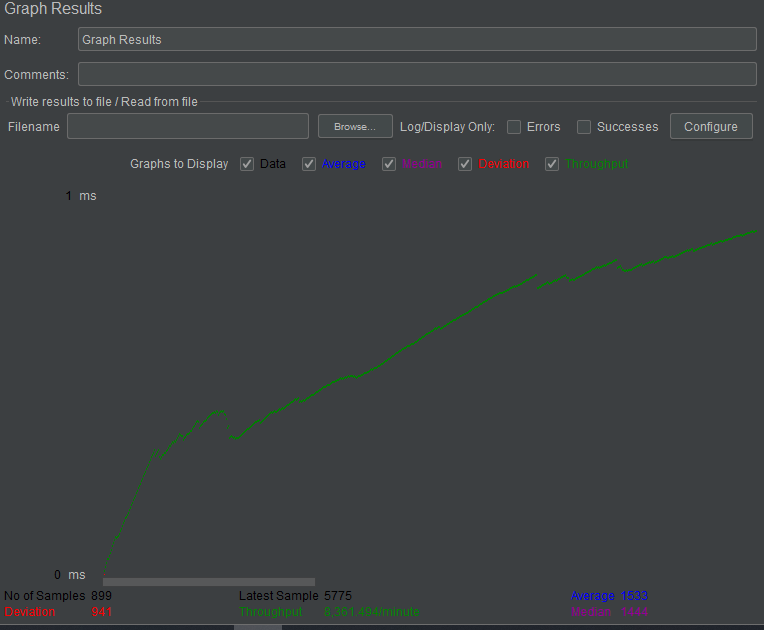
Hicimos una ardua búsqueda la cual iba ligada al número de peticiones que podría leer la interfaz de nuestra página en un tiempo determinado, con latencia mínima de un segundo. Concluimos que el número de peticiones óptimo que aguanta nuestra interfaz es de **899**, cuando se pasa de esa cifra ya empieza a fallar las peticiones y no las va a leer. 



Estos serían los resultados con el número que colocamos en las peticiones ya que como se puede ver en la foto saldrán todos en verde y para confirmar si hay algún error.



Esta es la tabla de los resultados ya que nos traería el reporte sobre todas las peticiones hechas en jmeter y los varios reportes que tenga para así podernos dar a conocer esos reportes.



Este es el resultado después de haber realizado la carga de stress que nos arrojaría una gráfica comprobando todo lo sucedido con la prueba.

# y

# Pruebas de caja blanca y caja negra

# 

# 

|  |  |
| --- | --- |
| **Descripción prueba de caja blanca** | |
| **Datos de entrada:** Código fuente | **Resultado:** Código fuente sin documentación. |
| **Tipo de flujo de datos:** Archivo - Pantalla | |
| **La estructura que viaja con el flujo:** CÓDIGO FUENTE - CONDICIONAL | |
| **Comentarios:** Se evidencia código fuente sin documentar, lo cual hace más complejo el hecho de comprenderlo y desarrollarlo. | |
| La acción varía dependiendo de si los datos ingresados son correctos a el igual si están almacenados en la base de datos. | |

# 

|  |  |
| --- | --- |
| **Descripción prueba de caja blanca** | |
| **Datos de entrada:** Código fuente | **Resultado:** Archivos CSS separados y ordenados. |
| **Tipo de flujo de datos:** Archivo - Pantalla | |
| **La estructura que viaja con el flujo:** CÓDIGO FUENTE - ORDEN | |
| **Comentarios:** Se evidencia archivos CSS correctamente separados en hojas para más orden a la hora de estructurar y maquetar. | |
| Código fuente aislado y ordenado. | |

# 

|  |  |
| --- | --- |
| **Descripción prueba de caja blanca** | |
| **Datos de entrada:** Código fuente HTML | **Resultado:** Código HTML especificado. |
| **Tipo de flujo de datos:** Archivo - Pantalla | |
| **La estructura que viaja con el flujo:** CÓDIGO FUENTE - DETALLADO | |
| **Comentarios:** Se evidencia Código HTML (img) especificado con respecto a su ruta Alt. | |
| Código html detallado y especificado. | |

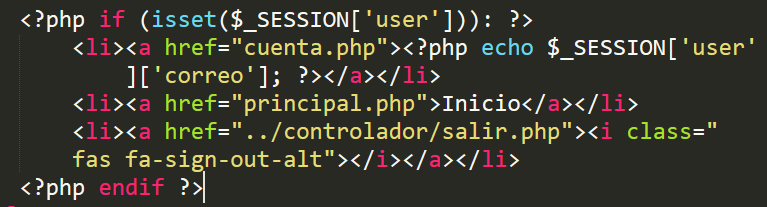
# 



# 

|  |  |
| --- | --- |
| **Descripción prueba de caja blanca** | |
| **Datos de entrada:** Código fuente HTML | **Resultado:** Código HTML indentado. |
| **Tipo de flujo de datos:** Archivo - Pantalla | |
| **La estructura que viaja con el flujo:** CÓDIGO FUENTE - Indentado | |
| **Comentarios:** Se evidencia Código HTML indentado para más orden de líneas y ahorro de las mismas. | |
| Código html indentado y organizado. | |

# 



# 

|  |  |
| --- | --- |
| **Descripción prueba de caja blanca** | |
| **Datos de entrada:** Código fuente PHP | **Resultado:** Código PHP fuera de estándar. |
| **Tipo de flujo de datos:** Archivo - Pantalla | |
| **La estructura que viaja con el flujo:** CÓDIGO PHP- fuera de estándar | |
| **Comentarios:** Se evidencia Código PHP fuera de estándar a la hora de culminar un condicional. | |
| Código php fuera del estándar de condición. | |

# 



# 

|  |  |
| --- | --- |
| **Descripción prueba de caja blanca** | |
| **Datos de entrada:** Código fuente Js | **Resultado:** Código Js sin separar de código HTML |
| **Tipo de flujo de datos:** Archivo - Pantalla | |
| **La estructura que viaja con el flujo:** CÓDIGO Js- sin aislar | |
| **Comentarios:** Se evidencia Código Js sin separar de Código HTML, esto causa menos ahorro de líneas por cada hoja al igual que hace que sea menos ordenado. | |
| Código Js sin aislar. | |

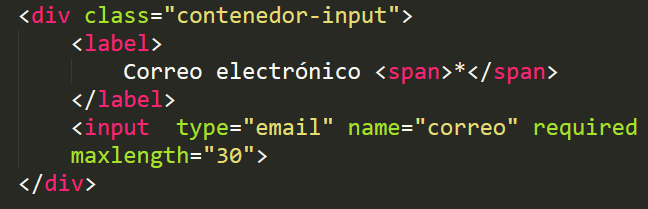
# 



# 

|  |  |
| --- | --- |
| **Descripción prueba de caja blanca** | |
| **Datos de entrada:** Librerías | **Resultado:** Librerías sin utilizar. |
| **Tipo de flujo de datos:** Archivo - Pantalla | |
| **La estructura que viaja con el flujo:** Librerías sin utilizar. | |
| **Comentarios:** Se evidencian algunas librerías sin utilizar en el código HTML. | |
| Librerias externas sin utilizar en el Código fuente. | |

# 



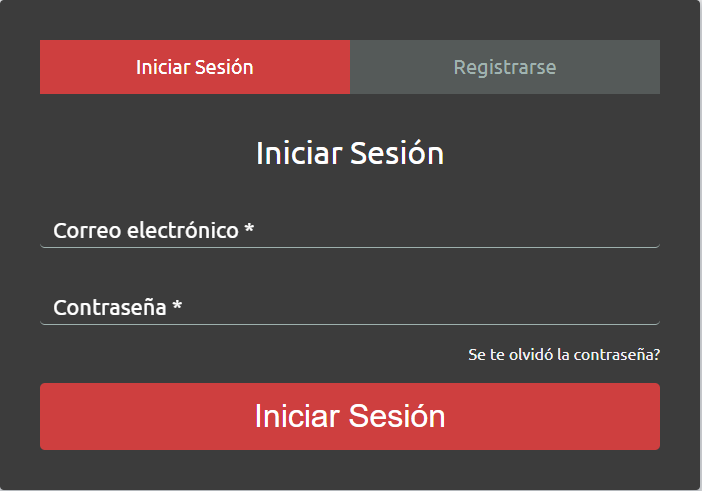
# 

|  |  |
| --- | --- |
| **Descripción prueba de caja blanca** | |
| **Datos de entrada:** Código fuente | **Resultado:** Código fuente. |
| **Tipo de flujo de datos:** Archivo - Pantalla | |
| **La estructura que viaja con el flujo: Codigo fuente.** | |
| **Comentarios:** Se evidencian algunos fallos con respecto a los términos predeterminados para rellenar cada uno de los campos. El usuario podría darle a “Inspeccionar elemento” y modificar ya sea el maximo o minimo de caracteres para rellenar el campo y causarnos inconvenientes a la hora del almacenamiento en la base de datos. | |
| /\* | |

# 

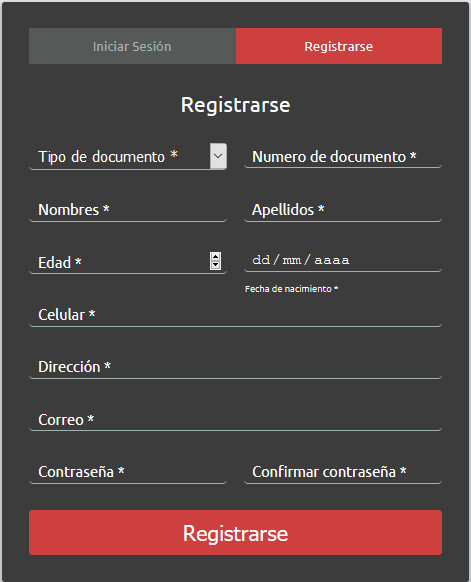
# **Capítulo 2 Pruebas de caja negra**

## **Inicio de sesión**



|  |  |
| --- | --- |
| **Descripción prueba de caja negra** | |
| **Datos de entrada**  Correo electrónico - Contraseña | **Resultado**  Inicio de sesión |
| **Tipo de flujo de datos**  Archivo – Index – Formulario – Controlador - Pantalla principal de productos | |
| **La estructura que viaja con el flujo**  Correo electrónico - Contraseña | |
| **Comentarios:** el usuario completa los campos requeridos, el controlador verifica la existencia del usuario y verifica que la contraseña sea la correcta | |
| Inicio de sesión: ok | |

## **Registro**



|  |  |
| --- | --- |
| **Descripción prueba de caja negra** | |
| **Datos de entrada**  Tipo de documento - Número de documento - Nombres - Apellidos - Edad - Fecha de nacimiento - Celular - Dirección - | **Resultado**  Registro de cliente |
| **Tipo de flujo de datos**  Archivo – Index – Formulario – Controlador - Index | |
| **La estructura que viaja con el flujo**  Los mismos datos de entrada | |
| **Comentarios:** el usuario completa los campos requeridos, el controlador verifica la inexistencia del número de documento, verifica la inexistencia del correo electrónico y verifica que las contraseñas sean idénticas. | |
| Inicio de sesión: ok | |

## **Insertar usuarios**



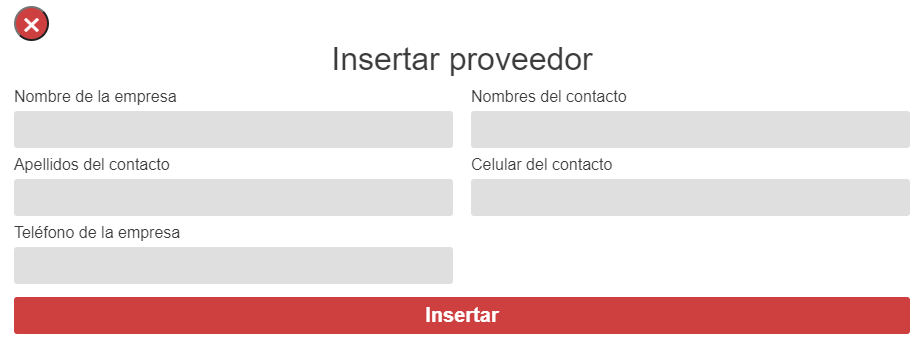
|  |  |
| --- | --- |
| **Descripción prueba de caja negra** | |
| **Datos de entrada**  Tipo de documento - Número de documento - Nombres - Apellidos - Edad - Fecha de nacimiento - Celular - Dirección - contraseña y confirmación de contraseña | **Resultado**  Registro de cliente desde la administración |
| **Tipo de flujo de datos**  Administración - Tabla de clientes - Formulario - Controlador - Tabla de clientes | |
| **La estructura que viaja con el flujo**  Los mismos datos de entrada | |
| **Comentarios:** el usuario administrador completa los campos requeridos, el controlador verifica la inexistencia del número de documento, verifica la inexistencia del correo electrónico y verifica que las contraseñas sean idénticas. | |
| Registro: ok | |

## **Insertar operadores**



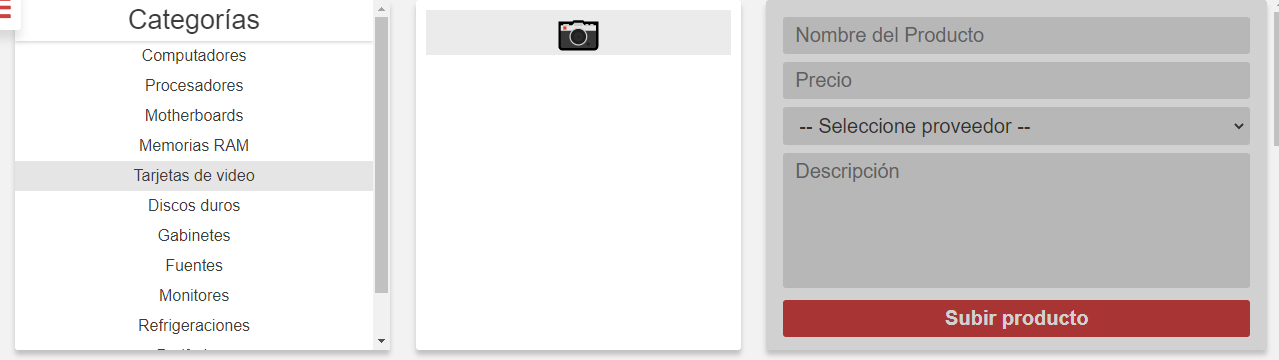
|  |  |
| --- | --- |
| **Descripción prueba de caja negra** | |
| **Datos de entrada**  Tipo de documento - Número de documento - Nombres - Apellidos - Edad - Fecha de nacimiento - Celular - Dirección - contraseña y confirmación de contraseña | **Resultado**  Registro de operador |
| **Tipo de flujo de datos**  Administración - Tabla de operadores - Formulario - Controlador - Tabla de operadores | |
| **La estructura que viaja con el flujo**  Los mismos datos de entrada | |
| **Comentarios:** el usuario administrador completa los campos requeridos, el controlador verifica la inexistencia del número de documento, verifica la inexistencia del correo electrónico y verifica que las contraseñas sean idénticas. | |
| Registro: ok | |

## **Insertar proveedores**



|  |  |
| --- | --- |
| **Descripción prueba de caja negra** | |
| **Datos de entrada**  Nombre de la empresa . Nombres del contacto - Apellidos del contacto - Celular del contacto - Teléfono de la empresa. | **Resultado**  Registro de un proveedor |
| **Tipo de flujo de datos**  Administración - Tabla de proveedores - Formulario - Controlador - Tabla de proveedores | |
| **La estructura que viaja con el flujo**  Los mismos datos de entrada | |
| **Comentarios:** el usuario operador o administrador completa los campos requeridos, el controlador inserta los datos | |
| Insertar proveedores: ok | |

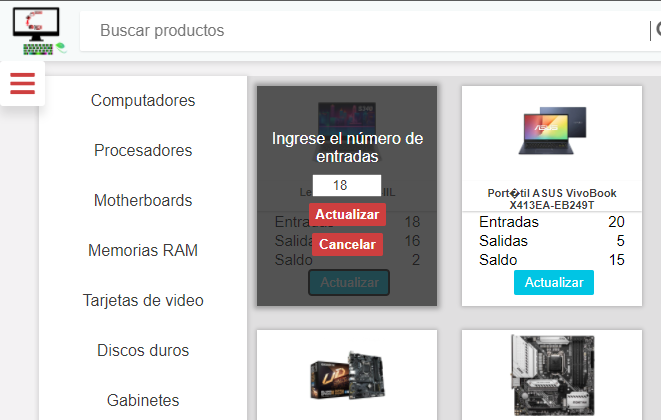
## **Insertar producto**



|  |  |
| --- | --- |
| **Descripción prueba de caja negra** | |
| **Datos de entrada**  Categoría, foto, nombre del producto, precio, proveedor, descripción. | **Resultado**  Inserción de un producto |
| **Tipo de flujo de datos**  Administración - Sección de productos - Formulario - Controlador - sección de productos | |
| **La estructura que viaja con el flujo**  Categoría - foto - nombre del producto - precio - proveedor - descripción - estado | |
| **Comentarios:** el usuario operador o administrador completa los campos requeridos, el controlador guarda la imagen, si se guarda correctamente, se insertan los datos. | |
| Insertar Producto: ok | |

## 

## **Modificar inventario**



|  |  |
| --- | --- |
| **Descripción prueba de caja negra** | |
| **Datos de entrada**  Número de entradas del producto | **Resultado**  Actualización del inventario |
| **Tipo de flujo de datos**  Administración - Inventario - formulario - controlador - inventario | |
| **La estructura que viaja con el flujo**  Número de entradas, salidas y producto | |
| **Comentarios:** el administrador o el operador agregan entradas del producto, el controlador recibe las entradas y calcula las entradas con las salidas para obtener el saldo y actualizar el producto en la base de datos | |
| Actualizar productos: ok | |

## 

|  |  |
| --- | --- |
| **Nombre del proyecto: componen tech** | |
| **Caso de prueba** | |
| **ID Caso de prueba:** REGISTRO | **Prueba diseñada por:** MJ, YP, KC, JM |
| **Prioridad de la prueba:** Establecer un buen almacenamiento de información a la hora de registrarnos. | **Fecha de diseño de la prueba: -- -- --** |
| **Nombre del módulo:** Registro (Usuarios) | **Prueba ejecutada por:** Marlon Jaramillo |
| **Título de la prueba:** Parámetros de insersión | **Fecha de ejecución de la prueba: -- -- --** |
| **Descripción:** Esta prueba consiste en intentar comprender hasta donde está funcionando la ejecución que lleva la información a la base de datos con respecto a los parámetros en cada uno de los campos a llenar. | |
| **Precondiciones:** Debe saltar un error a la hora de intentar modificar algún parámetro desde el código fuente.  **Dependencias:** | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Paso** | **Paso de la prueba** | **Datos de prueba** | **Resultado esperado** | **Resultado actual** | **Estatus** | **Notas** |
| 1 | Seleccionar el campo a llenar, y darle clic derecho “Inspeccionar elemento” |  |  |  |  | . |
| 2 | Deberás de eliminar algún Required o en lo posible un “Maxlength” que muy normalmente debe de tener tu campo. |  | El campo quedará de una manera neutra capaz de recolectar cualquier información sin importar sus caracteres |  |  | No deberás de actualizar la página luego de hacer este paso. |
| 3 | Por último, deberás de ingresar datos hacia la base de datos, enfocándote en dicho campo el cual le editaste los parámetros desde el código fuente. |  | Si tienes parámetros elaborados internamente desde Js, te saltará un error con respecto a ese campo que modificaste. | A la hora de ingresar información y excederme con los caracteres, se envía sin ningún fallo, con la posibilidad de saturar la base de datos teniendo en cuenta su limite de caracteres de raíz. |  | Es muy recomendable establecer los parámetros con js para evitar estos inconvenientes, y limitar las posibilidades de que cualquier usuario logre modificar los parámetros predeterminadamente creados desde el código fuente |

|  |
| --- |
| **Postcondiciones:** Se establecerá forzosamente una inserción a la base de datos intentando modificar los parámetros por medio del código fuente. |

**Información adicional.**

Con respecto a la inserción de datos externo (Login/Register) hay ciertos fallos con respecto a la estructura de seguridad para insertar nuevos datos al registrarse, esto puede suceder ya que los requisitos de cada campo están visibles a la hora de ver el código fuente, por lo cual cualquier usuario puede modificarlos e insertar los datos medianamente errados, saturando la base de datos e incluso colapsando con datos aleatorios.  
  
Otra situación que se llega a presentar es a la hora de establecer un límite con respecto a los parámetros, como se mencionó anteriormente, esto sucede sin necesidad de ingresar a modificar desde el código fuente, algunos campos están mal estructurados a la hora de ingresar la información.