**Documentación de Pruebas de Calidad**

**Módulo de creación de Productos - Componente Inventario**

1. **Introducción**

Este documento presenta el plan de pruebas de calidad para el componente Inventario, el cual permite la gestión de productos que ingresan a la barbería. Se detallan los casos de prueba, criterios de aceptación y escenarios de prueba para asegurar la correcta funcionalidad del módulo, incluyendo el registro, edición, visualización y eliminación de productos dentro del sistema

**Componente Analizado:** Inventario

**Tecnologías utilizadas:** React, Axios, SweetAlert2 **Fecha de evaluación:** 20 de mayo de 2025

1. **Descripción del Componente**

El componente Inventario permite gestionar productos de la barberia, incluye las siguientes funciones:

 Añadir productos al inventario con imagen

 Editar productos existentes

 Eliminar productos

 Mostrar tabla con los productos actuales

* Previsualización de imagen del producto
* Mostrar categorías disponibles
* Confirmaciones visuales y alertas de éxito o error

El componente utiliza el token de autenticación almacenado en localStorage para identificar al administrador creador del producto.

1. **Funcionalidades a Probar**

1. **Añadir Producto**

 Ingreso de datos en el formulario  Validación de campos

 Carga y previsualización de imagen

 Envío de datos al servidor

* Manejo de respuestas exitosas y errores

2. **Editar Producto**

* Edición de datos
* Actualización de imagen
* Envío y respuesta del servidor

3. **Eliminar Producto**

* **Confirmación con Sweetalert2**
* **Eliminación Efectiva**

4. **Cargar Inventario**

* **Llamada a API para cargar productos y categorias**
* **Renderizado Correcto**
  1. **Casos de Pruebas Funcionales**

| **ID** | **Caso de Prueba** | **Datos de Entrada** | **Resultado Esperado** | **Criterio de**  **Aceptación** |
| --- | --- | --- | --- | --- |
| CP- 01 | Crear Producto Exitosamente | nombre= Locion,  descripción= humectante, cantidad=10,  categoría=productos de cuidado personal,  proveedor= coca cola,  fecha= 20/05/25 18:00, imágen=archivo.jpg, precio:500.000 | Se guarda el producto, se muestra mensaje de éxito y se reinicia el formulario. | El producto se guarda en la base de datos y se muestra mensaje de exito |
| CP- 02 | Crear producto sin campos obligatorios | nombre= Locion,  descripción= humectante, cantidad=10,  proveedor= coca cola,  fecha= 20/05/25 18:00, imágen=archivo.jpg, precio:500.000  omitimos el campo categoría | El formulario bloquea el envío y marca los campos | No se permite enviar hasta completar todos los campos requeridos |
| CP- 03 | Crear producto con imagen no válida | nombre= Locion,  descripción= humectante, cantidad=10,  proveedor= coca cola,  fecha= 20/05/25 18:00, imágen=archivo.html, precio:500.000  se envia un formato invalido de imagen | El sistema rechaza la imagen | Se muestra alerta de error y no se envia |
| CP- 04 | Editar producto correctamente | nombre= Atomizador,  descripción= Efectivo, cantidad=100,  categoría=productos de cuidado personal,  proveedor=Pepsi,  fecha= 22/05/25 20:00, imágen=archivo.jpg, precio:800.000  Se modificaron los datos ingresados al principio para poder editar | El sistema actualiza el producto y recarga el inventario | Se actualiza y muestra alerta de éxito |
| CP- 05 | Eliminar Producto | Datos válidos, pero servidor no disponible | El sistema maneja el error y muestra mensaje adecuado | Se muestra mensaje de error indicando problema de  conexión |

|  |  |  |  | C |
| --- | --- | --- | --- | --- |

* 1. **Pruebas de Interfaz de Usuario**

| **ID** | **Caso de Prueba** | **Pasos** | **Resultado Esperado** | **Criterio de Aceptación** |
| --- | --- | --- | --- | --- |
| UI- 01 | Previsualización de imagen funciona  correctamente | 1. Seleccionar archivo de imagen2. Observar  área de previsualización | La imagen seleccionada se muestra en la  previsualización | La previsualización muestra correctamente la  imagen seleccionada |
| UI- 02 | Response en dispositivos móviles | Acceder desde un dispositivo móvil | El componente se adapta correctamente | Todos los elementos son  visibles |
| UI- 03 | Resaltado en campos obligatorios | Enviar formulario incompleto | los campos se marcan visualmente | Los campos muestran una alerta que deben de ser completados |
| UI- 04 | SweerAlert2 correctamente mostrado | Completar diferentes acciones que generan  alertas | Las alertas de SweetAlert2 se muestran  correctamente | Las alertas son legibles y contienen la información  adecuada |
| C |  |  |  |  |

1. **4 Pruebas de Seguridad**

| **ID** | **Caso de Prueba** | **Descripción** | **Resultado Esperado** | **Criterio de Aceptación** |
| --- | --- | --- | --- | --- |
| SEC- 01 | Validación de token de usuario | Eliminar el token de localStorage | El sistema detecta si el token es invalido | redirección al login o error visible |
| SEC- 02 | Inyección de código | Intentar ingresar código HTML/JavaScript en  campos de texto | El sistema sanitiza la entrada o rechaza  contenido malicioso | El contenido malicioso no se ejecuta ni almacena |
| SEC- 03 | Acceso sin autenticación | Intentar entrar a una ruta directamente sin el token | El sistema restringe el acceso | Solo el usuario con token válido puede acceder al componente |
| C |  |  |  | C |

1. **Pruebas de Rendimiento**

| **ID** | **Caso de Prueba** | **Descripción** | **Resultado Esperado** | **Criterio de**  **Aceptación** |
| --- | --- | --- | --- | --- |
| PERF-  01 | Tiempo de carga del  componente | Medir tiempo de carga  inicial | El componente se carga en  menos de 1 segundos | Fluidez al entrar al módulo |
| PERF- 02 | Tiempo de respuesta al guardar | Enviar formulario | La respuesta se recibe en menos de 1 segundos | Guardado rapido |
| PERF- 03 | Previsualización de imagen | Cargar imagen al seleccionar archivo | La previsualización aparece en menos de 1 segundo | Fluidez visual al cargar la imagen |
|  |  |  |  |  |

1. **Pruebas de Accesibilidad**

| **ID** | **Caso de Prueba** | **Descripción** | **Resultado Esperado** | **Criterio de Aceptación** |
| --- | --- | --- | --- | --- |
| ACC- 01 | Navegación por teclado | Usar Tab y Enter para moverse | Todos los campos son accesibles mediante teclado | Se puede navegar tranquilamente sin teclado |
| ACC- 02 | lector de pantalla | Probar componente con el lector de pantalla que desee | El lector de pantalla identifica correctamente todos los  elementos | Todos los elementos tienen etiquetas  accesibles |
| ACC- 03 | Contraste de color | Revisar contraste entre texto y fondo | El contraste cumple con estándares WCAG | Proporción de contraste según WCAG AA o  superior |
| C |  |  |  | C |

1. **Pruebas de Integración**

| **ID** | **Caso de Prueba** | **Descripción** | **Resultado Esperado** | **Criterio de Aceptación** |
| --- | --- | --- | --- | --- |
| INT- 01 | Conexión con Backend | Enviar y obtener datos de la API | La API procesa correctamente los datos | Se realiza la operación deseada |
| INT- 02 | Validación de token en el backend | Verificar que el token de  autenticación es utilizado correctamente | El backend válida  correctamente el token | No hay errores de autenticación |
| INT- 03 | Almacenamiento de imágenes | Subir imagen al Inventario | Las imágenes se guardan y se visualizan | Las imágenes se pueden recuperar  posteriormente |
| C |  |  |  | C |

1. **Conclusiones**

El componente Inventario cumple correctamente con todas las funciones, la interfaz es clara y funcional y las respuestas del servidor son manejadas correctamente.

1. **Anexos**
   1. **Diagrama de Flujo del Proceso**

INICIO

INGRESAR AL INVENTARIO

SELECCIONAR AÑADIR / EDITAR / ELIMINAR PRODUCTO

VALIDAR CAMPOS Y CARGAR IMAGEN

ENVIAR FORMULARIO

MOSTRAR ALERTA (ÉXITO / ERROR)

ACTUALIZAR INVENTARIO VISUAL

SI HAS SEGUIDO LOS PASOS CORRECTAMENTE EL RESULTADO DEBE SER DE ÉXITO DE LO CONTRARIO SERÁ DE ERROR

* 1. **Entorno de Pruebas Recomendado**

**** **Navegadores a probar:**

 Google Chrome (última versión)

 Mozilla Firefox (última versión)  Microsoft Edge (última versión)  Brave (última versión)

 **Dispositivos a probar:**

 Desktop (resoluciones 1920x1080, 1366x768)  Tablet (iPad, resolución 768x1024)

 Móvil (iPhone, Galaxy Alcatel, resoluciones 375x667, 414x896)

 **Herramientas recomendadas:**

 React Testing Library para pruebas unitarias  Cypress para pruebas end-to-end

 Lighthouse para evaluación de rendimiento  WAVE o axe para pruebas de accesibilidad

* 1. **Código Fuente del Componente**

import NavbarAdmin from '../../Components/NavbarAdmin'

import SidebarAdmin from '../../Components/SidebarAdmin'

import Swal from 'sweetalert2'

import React, { useEffect, useState } from 'react';

import axios from 'axios'

import { useNavigate } from 'react-router-dom'

export default function Inventario() {

const [imagePreviewEdit, setImagePreviewEdit] = useState('');

const [imagePreview, setImagePreview] = useState('');

const [inventario, setInventario] = useState([]);

const [producto, setProducto] = useState({

nombre: '',

descripcion\_P: '',

cantidad: '',

id\_categoria\_producto: '',

proveedor: '',

fecha\_venta: '',

foto: null,

PrecioUnitario: ''

});

const [productoEditar, setProductoEditar] = useState({

nombre: '',

descripcion\_P: '',

cantidad: '',

id\_categoria\_producto: '',

proveedor: '',

fecha\_venta: '',

foto: null,

PrecioUnitario: ''

});

const [categorias, setCategorias] = useState([]);

const navigate = useNavigate();

// Crear los Productos

const handleSubmit = async (e) => {

e.preventDefault();

try {

const formData = new FormData();

formData.append('nombre', producto.nombre);

formData.append('descripcion\_P', producto.descripcion\_P);

formData.append('cantidad', producto.cantidad);

formData.append('id\_categoria\_producto', producto.id\_categoria\_producto);

formData.append('proveedor', producto.proveedor);

formData.append('fecha\_venta', producto.fecha\_venta);

formData.append('foto', producto.foto);

formData.append('PrecioUnitario', producto.PrecioUnitario);

const res = await axios.post(`http://localhost:8080/CreateInventario`, formData);

if (res.status === 200) {

Swal.fire({

icon: 'success',

title: res.data,

timer: 1000,

customClass: {

popup: "dark-theme-popup bg-dark antonparabackend ",

},

}).then(() => {

window.location.reload(0);

});

}

} catch (err) {

console.log(err);

Swal.fire({

icon: 'error',

title: 'Error',

text: err.response.data,

customClass: {

popup: "dark-theme-popup bg-dark antonparabackend ",

},

});

}

};

const handleSubmitEdit = async (e) => {

e.preventDefault();

try {

const formData = new FormData();

formData.append('nombre', productoEditar.nombre);

formData.append('descripcion\_P', productoEditar.descripcion\_P);

formData.append('cantidad', productoEditar.cantidad);

formData.append('id\_categoria\_producto', productoEditar.id\_categoria\_producto);

formData.append('proveedor', productoEditar.proveedor);

formData.append('fecha\_venta', productoEditar.fecha\_venta);

formData.append('foto', productoEditar.foto);

formData.append('PrecioUnitario', productoEditar.PrecioUnitario);

const res = await axios.put(`http://localhost:8080/UpdateInventario/${productoEditar.id\_producto}`, formData);

if (res.status === 200) {

Swal.fire({

icon: 'success',

title: res.data,

timer: 1000,

customClass: {

popup: "dark-theme-popup bg-dark antonparabackend ",

},

}).then(() => {

window.location.reload(0);

});

}

} catch (err) {

console.log(err);

Swal.fire({

icon: 'error',

title: 'Error',

text: err.response.data,

customClass: {

popup: "dark-theme-popup bg-dark antonparabackend ",

},

});

}

};

const handleFileChange = (e) => {

const selectedFile = e.target.files[0];

if (selectedFile) {

setProducto({ ...producto, [e.target.name]: selectedFile });

setImagePreview(URL.createObjectURL(selectedFile));

} else {

Swal.fire({

icon: 'error',

title: 'Error', text: 'No se pudo subir la imagen',

customClass: {

popup: "dark-theme-popup bg-dark antonparabackend ",

},

});

}

};

// Handle change para editar foto

const handleFileChangeEdit = (e) => {

const selectedFile = e.target.files[0];

if (selectedFile && selectedFile.type.startsWith('image/')) {

setProductoEditar({ ...productoEditar, [e.target.name]: selectedFile });

setImagePreviewEdit(URL.createObjectURL(selectedFile));

} else {

Swal.fire({

icon: 'error',

title: 'Error',

text: 'Por favor, seleccione un archivo de imagen válido.',

customClass: {

popup: "dark-theme-popup bg-dark antonparabackend ",

},

});

}

};

const handleChangeEdit = (e) => {

setProductoEditar(prev => ({ ...prev, [e.target.name]: e.target.value }));

};

// Handle change para añadir

const handleChange = (e) => {

setProducto(prev => ({ ...prev, [e.target.name]: e.target.value }));

};

const DeleteInventario = async (id) => {

try {

const confirm = await Swal.fire({

title: '¿Estas seguro de borrar este producto?',

text: "No podrás revertir esta acción",

icon: 'question',

showCancelButton: true,

confirmButtonColor: '#3085d6',

cancelButtonColor: '#d33',

confirmButtonText: 'Sí, borrar',

customClass: {

popup: "dark-theme-popup bg-dark antonparabackend ",

},

});

if (!confirm.isConfirmed) {

return;

}

const res = await axios.delete(`http://localhost:8080/DeleteInventario/${id}`);

console.log(res);

if (res.status === 200) {

Swal.fire({

icon: 'success',

title: res.data,

customClass: {

popup: "dark-theme-popup bg-dark antonparabackend ",

},

}).then(() => {

navigate(0);

})

}

} catch (error) {

console.log(error);

Swal.fire({

icon: 'error',

title: 'Error al borrar',

text: error.response.data,

customClass: {

popup: "dark-theme-popup bg-dark antonparabackend ",

},

});

}

};

useEffect(() => {

const fetchData = async () => {

try {

const [inventarioRes, categoriasRes] = await Promise.all([

axios.get(`http://localhost:8080/GetInventario`),

axios.get(`http://localhost:8080/categorias`),

]);

setInventario(inventarioRes.data);

setCategorias(categoriasRes.data);

} catch (error) {

console.log(error);

}

};

fetchData();

}, []);

const openEditModal = (item) => {

setProductoEditar(item);

};

return (

<div>

<NavbarAdmin />

<SidebarAdmin />

<div className='mt-5'>

<div className='contenido ' id='Inventario'>

<p className='text-center text-white mt-5 display-6 bebas mx-3 '>HOLA, <span className='text-danger'>ADMINISTRADOR</span>| ESTE ES EL INVENTARIO DE LOS PRODUCTOS QUE ENTRAN A LA BARBERIA</p>

<div className="d-flex justify-content-end mx-5 mt-5 ">

<button type="button" class="btn btn-danger .col-md-4" data-bs-toggle="modal" data-bs-target="#AñadirModal" data-bs-whatever="@mdo" >Añadir</button>

</div>

<div className='container text-center'>

<div className="table-responsive">

<table class="table table-bordered table-responsive table-dark table-hover mt-4 container p-5">

<thead>

<tr>

<th scope="col" className=' bebas p-4 text-danger'>ID Producto</th>

<th scope="col" className=' bebas p-4 fs-3'>Nombre</th>

<th scope="col" className=' bebas p-4 fs-3'>Descripcion</th>

<th scope="col" className=' bebas p-4 fs-3'>Cantidad</th>

<th scope="col" className=' bebas p-4 fs-3'>Categoria</th>

<th scope="col" className=' bebas p-4 fs-3'>Proveedor</th>

<th scope="col" className=' bebas p-4 fs-3'>Fecha Y Hora</th>

<th scope="col" className=' bebas p-4 fs-3'>Imagen</th>

<th scope="col" className=' bebas p-4 fs-3'>Precio</th>

<th scope="col" className=' bebas p-4 text-warning fs-3'>Acciones</th>

</tr>

</thead>

<tbody className='p-5'>

{inventario.map((item) => (

<tr key={item.id\_producto}>

<th className='text-center'>{item.id\_producto}</th>

<td className='text-center'>{item.nombre}</td>

<td className='text-center'>{item.descripcion\_P}</td>

<td className='text-center'>{item.cantidad}</td>

<td className='text-center'>{categorias.find(c => c.id\_categoria\_producto === item.id\_categoria\_producto).categoria}</td>

<td className='text-center'>{item.proveedor}</td>

<td className='text-center'>{item.fecha\_venta}</td>

<td className='text-center'><img src={`http://localhost:8080/ImagesInventario/${item.Foto}`} className='img-fluid zoomhover2' style={{ width: '150px', height: '150px', objectFit: 'cover' }} /></td>

<td className='text-center'>{item.PrecioUnitario}</td>

<td>

<div className="d-flex">

<button type="button" className="btn btn-outline-warning me-3" onClick={() => openEditModal(item)} data-bs-toggle="modal" data-bs-target="#exampleModal" data-bs-whatever="@mdo">

<i className='bi bi-pencil-fill text-white'></i>

</button>

<button className='btn btn-outline-danger' onClick={() => DeleteInventario(item.id\_producto)}>

<i className="bi bi-trash-fill" ></i>

</button>

</div>

</td>

</tr>

))}

</tbody>

</table>

</div>

</div>

{/\* MODAL EDITAR \*/}

<div class="modal fade" id="exampleModal" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content bg-dark">

<form onSubmit={handleSubmitEdit}>

<div class="modal-header">

<h1 class="modal-title fs-3 text-white bebas" id="exampleModalLabel">EDITAR</h1>

<button type="button" class="btn-close bg-white" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<div className="d-flex justify-content-center ">

<div className="card bg-dark" style={{ width: '10rem' }}>

<img src={imagePreviewEdit || `http://localhost:8080/ImagesInventario/${productoEditar.foto}`} style={{ width: '100%', height: '100%', objectFit: 'cover' }} className='img-fluid text-white rounded' alt="Imagen" />

</div>

</div>

<div class="modal-body">

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend ">Nombre:</label>

<input type="text" value={productoEditar.nombre} class="form-control bg-dark text-white" id="recipient-name" name='nombre' onChange={handleChangeEdit} />

</div>

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Descripcion:</label>

<input type="text" value={productoEditar.descripcion\_P} class="form-control bg-dark text-white" id="recipient-name" name='descripcion\_P' onChange={handleChangeEdit} />

</div>

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Cantidad:</label>

<input type="text" value={productoEditar.cantidad} class="form-control bg-dark text-white" id="recipient-name" name='cantidad' onChange={handleChangeEdit} />

</div >

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Categoria:</label>

<select name="id\_categoria\_producto" value={productoEditar.id\_categoria\_producto} class="form-select bg-dark text-white" id="" onChange={handleChangeEdit}>

<option selected disabled>Seleccione una categoria</option>

{categorias.map((item) => (

<option key={item.id\_categoria} value={item.id\_categoria\_producto}>{item.categoria}</option>

))}

</select>

</div>

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Proveedor:</label>

<input type="text" value={productoEditar.proveedor} class="form-control bg-dark text-white" id="recipient-name" name='proveedor' onChange={handleChangeEdit} />

</div>

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Fecha Y Hora De Venta</label>

<input type="datetime-local" class="form-control bg-dark text-white" id="recipient-name" name='fecha\_venta' onChange={handleChangeEdit} />

</div>

<p className="text-white antonparabackend">Editar Imagen</p>

<div className="input-group">

<input

name="foto"

accept="image/\*"

type="file"

className="form-control bg-dark text-white"

id="inputGroupFile04"

onChange={handleFileChangeEdit}

/>

</div>

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Costo de venta:</label>

<input value={productoEditar.PrecioUnitario} type="text" class="form-control bg-dark text-white" id="recipient-name" name='PrecioUnitario' onChange={handleChangeEdit} />

</div>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal" >Cerrar</button>

<button type="sumbit" class="btn btn-danger ">Editar</button>

</div>

</form>

</div>

</div>

</div>

{/\* MODAL DE AÑADIR \*/}

<div class="modal fade" id="AñadirModal" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content bg-dark">

<form onSubmit={handleSubmit}>

<div class="modal-header">

<h1 class="modal-title fs-3 text-white text-white bebas" id="exampleModalLabel" >AÑADIR</h1>

</div>

<div className="modal-body d-flex justify-content-center ">

<div className="card bg-dark" style={{ width: '10rem' }}>

<img src={imagePreview || ''} style={{ width: '100%', height: '100%', objectFit: 'cover' }} className='img-fluid text-white rounded' alt="Imagen" />

</div>

</div>

<div class="modal-body">

<form>

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Nombre</label>

<input type="text" class="form-control bg-dark text-white" id="recipient-name" name='nombre' onChange={handleChange} />

</div>

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Descripcion</label>

<input type="text" class="form-control bg-dark text-white" id="recipient-name" name='descripcion\_P' onChange={handleChange} />

</div>

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Cantidad</label>

<input type="text" class="form-control bg-dark text-white" id="recipient-name" name='cantidad' onChange={handleChange} />

</div >

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Categoria</label>

<select name="id\_categoria\_producto" class="form-select bg-dark text-white" id="" onChange={handleChange}>

<option selected disabled>Seleccione una categoria</option>

{categorias.map((item) => (

<option key={item.id\_categoria} value={item.id\_categoria\_producto}>{item.categoria}</option>

))}

</select>

</div>

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Proveedor</label>

<input type="text" class="form-control bg-dark text-white" id="recipient-name" name='proveedor' onChange={handleChange} />

</div>

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Fecha Y Hora</label>

<input type="datetime-local" class="form-control bg-dark text-white" id="recipient-name" name='fecha\_venta' onChange={handleChange} />

</div>

<p className="text-white antonparabackend">Imagen Del Producto</p>

<div className="input-group">

<input

required

name="foto"

accept="image/\*"

type="file"

className="form-control bg-dark text-white "

id="inputGroupFile04"

onChange={handleFileChange}

style={{ width: '50f%', height: '50%', objectFit: 'cover' }}

/>

</div>

<div class="mb-3">

<label for="recipient-name" class="col-form-label text-white antonparabackend">Costo de venta</label>

<input type="text" class="form-control bg-dark text-white" id="recipient-name" name='PrecioUnitario' onChange={handleChange} />

</div>

</form>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Cerrar</button>

<button type="submit" class="btn btn-danger" onClick={handleSubmit}>Añadir</button>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

</div>

)

}