



ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE
CIENCIAS DE LA COMPUTACIÓN

INGENIERÍA DE SOFTWARE

SEXTO NIVEL

*“SISTEMA DE GESTIÓN DE INVENTARIOS PARA
EL EMPRENDIMIENTO BLOZ CELL”.*

Integrantes:

Jurado Junior
Jonathan Lituma
Ivette Román

Tutor:

Ing. Jenny Ruiz

Sangolquí, 2023

Historial de Revisiones

Fecha	Versión	Descripción	Autor
08/02/2023	1.0	Revisión documental	Jurado Junior Lituma Jhonatan Román Ivette

Contenido

RESUMEN EJECUTIVO	4
INTRODUCCIÓN	6
CAPÍTULO I	7
1.1 Título del Proyecto	7
1.2. Sistema de Objetivos	7
1.2.1 Objetivo General	7
1.2.2 Objetivos Específicos	7
1.3 Alcance	7
1.4 Definición y Justificación del Problema	8
1.5 Presupuesto	9
1.5.1 Recursos Hardware y Software	9
1.5.2 Recursos Humanos	10
1.5.3 Factibilidad Económica	10
CAPÍTULO II	11
2.1 Modelamiento del Negocio y sus Entregables	11
2.1.1 Documentos de Caso de Uso Historias de Usuario	11
2.2 Definición de Requerimientos	14
2.2.1 Especificación de Requerimientos de Software	14
Requisitos Funcionales	23
2.2.2 Especificación de Casos de Uso	23
Figura 11. Diagrama de casos de uso Gestionar Categorías	23
Figura 12. Diagrama de casos de uso Gestionar Productos	24
2.3 Análisis y Diseño	24
2.3.1 Modelo Conceptual	24
2.3.2 Modelo Lógico	24
2.3.3 Modelo Físico	24
2.3.4 Script de la Base de Datos	25
2.4 Implementación	31
2.4.1 Modelo de Arquitectura General	31
2.4.2 Modelo de Arquitectura Cliente-Servidor	33
CAPÍTULO III	33
3.1 Pruebas	33
3.1.1 Prueba de Caja Negra	33
3.1.2 Prueba de Caja Blanca	33

3.1.3 Documentación de Informe de Errores	33
3.1.3.1 Enlace: video demostrativo de evaluación por pares	34
3.1.4 Documentación Técnicas Caja Blanca y Caja Negra	34
3.1.4.1 TÉCNICA DE CAJA BLANCA: Cobertura de caminos	34
3.1.4.2 TÉCNICA DE CAJA NEGRA: Clases equivalentes	49
CAPÍTULO IV	55
4.1 Conclusiones	55
4.2 Recomendaciones.	55

RESUMEN EJECUTIVO

El objetivo del proyecto es mejorar la eficiencia y efectividad en la gestión de inventarios de Bloz Cell, mediante la implementación de un sistema de gestión basado en la metodología ágil SCRUM. Con esto se busca involucrar al usuario en cada etapa del desarrollo, a fin de garantizar un software de alta calidad y que cumpla con las expectativas del cliente.

Para lograr esto, se llevará a cabo un análisis y diseño de un software de gestión de inventarios, que incluirá la identificación de los requisitos de negocio, la planificación, el diseño, la implementación y la realización de pruebas exhaustivas. También se presentará la documentación generada en cada iteración del proyecto, para garantizar la transparencia y el seguimiento del progreso.

Se espera que este proyecto mejore la precisión de los inventarios, reduzca los costos de almacenamiento, aumente la eficiencia en la toma de decisiones y mejore la satisfacción del cliente.

INTRODUCCIÓN

El sistema de inventarios es un aspecto crítico en cualquier emprendimiento que desea mantener su competitividad y eficiencia operativa. Un buen sistema de inventarios permite a los empresarios controlar y gestionar sus activos de manera efectiva, maximizando la rentabilidad y minimizando los costos.

El objetivo principal de un sistema de inventarios es mantener un registro preciso y actualizado de la cantidad y el valor de los bienes de la empresa. Esto incluye desde materias primas hasta productos terminados y suministros de oficina. Un sistema de inventarios eficaz permite a los empresarios determinar cuándo realizar pedidos y cuánto ordenar, evitando la escasez o la acumulación excesiva de inventario.

Además, un sistema de inventarios también ayuda a los empresarios a identificar las áreas de la empresa que pueden requerir mejoras, tales como la optimización de los procesos de producción o la implementación de mejores prácticas de almacenamiento. También puede ser utilizado para analizar el desempeño financiero de la empresa y para tomar decisiones estratégicas informadas.

Es por ello que se pretende el desarrollo de un sistema que ayude a gestionar los inventarios de la empresa Bloz Cell, el cual facilite el desarrollo de la misma y ayude a llevar un control exhaustivo de los diferentes productos dentro del emprendimiento.

CAPÍTULO I

1.1 Título del Proyecto

Sistema de inventarios para el emprendimiento Bloz Cell

1.2. Sistema de Objetivos

1.2.1 Objetivo General

Desarrollar un sistema de inventario para el emprendimiento Bloz Cell, aplicando los diferentes conocimientos adquiridos en las diferentes materias del desarrollo de software. Con el fin de permitir que el administrador lleve un control de las entradas y salidas de los productos del emprendimiento.

1.2.2 Objetivos Específicos

- Realizar la matriz de historias de usuario (HU), mediante la implementación de la técnica de las 5W y 2H.
- Realizar casos de prueba y reporte de errores.
- Implementar el patrón de diseño acorde a los requisitos funcionales planteados en el documento de especificación de requisitos de software.
- Proponer un cronograma con las actividades a realizar durante el desarrollo del proyecto.

1.3 Alcance

Las fases de Análisis y Diseño de Software que se considerarán para el desarrollo del sistema de inventarios son:

Etapas de Análisis

En esta fase se va a definir el problema que se plantea resolver, es importante que todos los integrantes del equipo de desarrollo conozcan el dominio del problema y cuáles son las necesidades del usuario.

Etapas de Diseño

Con ayuda de la información obtenida se va a desarrollar un modelo que se adapte a las necesidades y requerimientos.

Etapas de Desarrollo

Se deben aplicar los conocimientos en programación para el desarrollo del sistema. Se plantea el desarrollo de un aplicativo web con el fin de que sea de fácil acceso para los trabajadores del emprendimiento y se les facilite la tarea de la gestión de inventarios es decir que logren agregar, modificar, actualizar y eliminar artículos. Por ellos se pretende el desarrollo de 2 principales funcionalidades las cuales son:

- Gestionar productos: el usuario puede agregar, modificar, actualizar, eliminar y observar productos.
- Gestionar categorías: usuario puede registrar, editar, modificar, eliminar y enlistar las categorías.

Etapas de Pruebas

En esta etapa se debe comprobar que el sistema cumpla con la especificación de requisitos, con la ayuda de los casos de prueba.

1.4 Definición y Justificación del Problema

La gestión de inventarios es esencial para las organizaciones, ya que proporciona un mayor control sobre la continuidad de las actividades, el uso racional de los recursos disponibles, el mantenimiento de un nivel satisfactorio de servicio a los clientes y la garantía de la independencia de otras actividades y de las funciones que deben desempeñarse en ellas.

Para Bloz Cell los bienes destinados a la venta incluye celulares y toda clase de accesorios de los mismos por tal motivo la empresa, se ven en la necesidad de poseer mecanismos o fuentes de información que registren de una manera más eficiente todos los costos y gastos relacionados con la compra y adquisición de la Mercancía.

El sistema de inventario tiene una importancia trascendental, ya que afecta tanto al balance como a la cuenta de resultados, y es representativo de la eficiencia y la eficacia de la prestación del servicio. También es útil para mejorar la administración e identificar las necesidades actuales, lo que le guiará para satisfacer sus propias necesidades y las de sus clientes.

1.5 Presupuesto

1.5.1 Recursos Hardware y Software

Recurso	Herramienta	Valor (USD)
Sistema operativo	Windows 10	0
Programa de ofimática	G Suite de Google	0
Herramienta para el modelado de la base de datos	PowerDesigner	0
Herramienta para los diferentes diagramas	Lucid Chart	0
Herramienta para el diseño de la arquitectura	Visual Paradigm	0
IDE para desarrollo	Visual Studio Code	0
		0

*Tabla 1. Recursos Software
Tabla Elaborada por los Autores*

Recurso	Requisitos de los estudiantes	Valor (USD)
Computador personal primer estudiante	DELL	900
Computador personal segundo estudiante	DELL	900
Computador personal tercer estudiante	HP	900
		2700

*Tabla 2. Recursos Hardware
Tabla Elaborada por los Autores*

1.5.2 Recursos Humanos

Nombre	Cargo	Funciones	Periodo de tiempo
Jurado Junior	Analista	Encargada de redactar, investigar, analizar las necesidades del usuario. Redactar la documentación detrás del desarrollo.	9 de noviembre del 2022 hasta la finalización, calificación y presentación del documento el 27 de febrero de 2023
Lituma Jhonatan	Desarrollador	Encargado de analizar los requerimientos obtenidos, y desarrollar la aplicación funcional basada en la documentación realizada.	9 de noviembre del 2022 hasta la finalización, calificación y presentación del documento el 27 de febrero de 2023
Román Ivette	Líder de proyecto	Encargada de redactar, investigar, analizar las necesidades del usuario. Redactar la documentación detrás del desarrollo. Además de verificar que se cumpla el plazo en cuanto al tiempo.	9 de noviembre del 2022 hasta la finalización, calificación y presentación del documento el 27 de febrero de 2023

*Tabla 3. Recursos Humanos
Tabla Elaborada por los Autores*

1.5.3 Factibilidad Económica

	Primer Mes	Segundo Mes	Tercer Mes	Cuarto Mes	Total
Computadores	2 700	0	0	0	2 700
Recursos software	0	0	0	0	0
Desarrollador de software	360	360	360	360	1 800
Total					4 500

*Tabla 4. Factibilidad económica
Tabla Elaborada por los Autores*

CAPÍTULO II

2.1 Modelamiento del Negocio y sus Entregables

2.1.1 Documentos de Caso de Uso Historias de Usuario

HISTORIA DE USUARIO (HU)			
ITEM	USUARIO	STATUS	
REQ001	Administrador	Terminado	
TIEMPO	PRIORIDAD	PROG. RESP	
6	Alta	Jhonatan Lituma	
QUE	Registrar Productos	PARA QUE	Para agregar nuevos productos al aplicativo
COMO	1.- El aplicativo despliega los atributos descritos que el administrador debe llenar para crear un nuevo producto. 2.- El administrador asigna una categoría al producto.		
NOMBRE HISTORIA	Registrar productos		
PRUEBA	Crear una prueba unitaria para conocer si el registro procedio con normalidad	COMENTARIOS	Sin comentarios

Figura 1. Historia de Usuario Registrar Productos

HISTORIA DE USUARIO (HU)			
ITEM	USUARIO	STATUS	
REQ002	Administrador	Terminado	
TIEMPO	PRIORIDAD	PROG. RESP	
3	Alta	Jhonatan Lituma	
QUE	Enlistar productos	PARA QUE	Para mostrar al administrador los productos existentes
COMO	1.- El administrador ingresa al sistema. 2.- Click en la seccion "Productos". 3.- El aplicativo enlista toda la información de los productos que se haya registrado.		
NOMBRE HISTORIA	Enlistar productos		
PRUEBA	Verificando que los productos se muestren de forma correcta, mediante la creacion de una prueba unitaria.	COMENTARIOS	Sin comentarios

Figura 2. Historia de Usuario Listar Productos

HISTORIA DE USUARIO (HU)			
ITEM	USUARIO	STATUS	
REQ003	Administrador	Terminado	
TIEMPO	PRIORIDAD	PROG. RESP	
4	Alta	Jhonatan Lituma	
QUE	Editar productos	PARA QUE	Para que el administrador puede corregir errores en el producto existente
COMO	1.- Dar click en el boton "Editar". 2.- Cambiar los parametros que desea editar. 3.- Dar click en el boton "Guardar"		
NOMBRE HISTORIA	Editar productos		
PRUEBA	Crear una prueba unitaria para conocer si la edicion de los parametros del producto se actualizo en la base de datos ademas observar en el aplicativo si estos datos se muestran de manera correcta.	COMENTARIOS	Sin comentarios

Figura 3. Historia de Usuario Editar Productos


HISTORIA DE USUARIO (HU)					
ITEM	USUARIO	STATUS			
REQ004	Administrador	Terminado			
TIEMPO	PRIORIDAD	PROG. RESP			
4	Alta	Jhonatan Lituma			
QUE	Eliminar productos		PARA QUE	Poder sacar de circulación ciertos productos agotados o que no se van a vender más	
			COMO	1.- Seleccionar el producto. 2.- Dar clic en el botón "Eliminar". 3.- Dar clic en el botón "Aceptar". Muestra un mensaje que se quitara el producto de la base de datos.	
NOMBRE HISTORIA			Eliminar productos		
PRUEBA		Crear una prueba unitaria para conocer si la eliminación procedió con normalidad y si el producto fue removido de la base de datos.		COMENTARIOS	Sin comentarios

Figura 4. Historia de Usuario Eliminar Productos

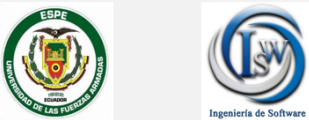
HISTORIA DE USUARIO (HU)					
ITEM	USUARIO	STATUS			
REQ005	Administrador	En proceso			
TIEMPO	PRIORIDAD	PROG. RESP			
4	Alta	Jhonatan Lituma			
QUE	Acceso al sistema		PARA QUE	Pueda acceder a la funcionalidades del sistema como la gestión de productos y categorías.	
			COMO	1.- Dar clic en el botón "Iniciar Sesión". 2.- Ingresar el usuario y contraseña. 3.- Dar clic en iniciar	
NOMBRE HISTORIA			Iniciar Sesión		
PRUEBA		Crear una prueba unitaria para verificar que el sistema valida correctamente el usuario y la contraseña.		COMENTARIOS	Sin comentarios

Figura 5. Historia de Usuario Acceso al Sistema


HISTORIA DE USUARIO (HU)					
ITEM	USUARIO	STATUS			
REQ006	Administrador	Terminado			
TIEMPO	PRIORIDAD	PROG. RESP			
6	Alta	Jhonatan Lituma			
QUE	Registrar Categoría		PARA QUE	Para agregar categorías	
			COMO	1.- El aplicativo despliega los atributos descritos que el administrador debe llenar para crear una nueva categoría. 2.- El aplicativo validará que todos los datos solicitados estén llenos.	
NOMBRE HISTORIA			Registrar categorías		
PRUEBA		Crear una prueba unitaria para conocer si el registro procedió con normalidad		COMENTARIOS	Sin comentarios

Figura 6. Historia de Usuario Registrar Categorías



HISTORIA DE USUARIO (HU)				
ITEM	USUARIO	STATUS		
REQ007	Administrador	Terminado	 	
TIEMPO	PRIORIDAD	PROG. RESP		
3	Alta	Jhonatan Lituma		
QUE	Enlistar Categoría	PARA QUE	Para mostrar al administrador las categorías registradas	COMO
				1.- El administrador ingresa al sistema. 2.- Click en la sección "Categorías". 3.- El aplicativo enlista toda la información de las categorías registradas.
NOMBRE HISTORIA		Enlistar categorías		
PRUEBA	Verificando que los productos se muestren de forma correcta, mediante la creación de una prueba unitaria.		COMENTARIOS	Sin comentarios

Figura 7. Historia de Usuario Enlistar Categorías



HISTORIA DE USUARIO (HU)				
ITEM	USUARIO	STATUS		
REQ008	Administrador	Terminado	 	
TIEMPO	PRIORIDAD	PROG. RESP		
4	Alta	Jhonatan Lituma		
QUE	Editar Categoría	PARA QUE	Para que el administrador puede editar información	COMO
				1.- Dar click en el botón "Editar". 2.- Cambiar los parámetros que desea editar. 3.- Dar click en el botón "Guardar"
NOMBRE HISTORIA		Editar categorías		
PRUEBA	Crear una prueba unitaria para conocer si la edición de los parámetros del producto se actualiza en la base de datos además observar en el aplicativo si estos datos se muestran de manera correcta.		COMENTARIOS	Sin comentarios

Figura 8. Historia de Usuario Editar Categorías



HISTORIA DE USUARIO (HU)				
ITEM	USUARIO	STATUS		
REQ009	Administrador	Terminado	 	
TIEMPO	PRIORIDAD	PROG. RESP		
4	Alta	Jhonatan Lituma		
QUE	Eliminar Categoría	PARA QUE	Poder eliminar categorías no usadas	COMO
				1.- Seleccionar el producto. 2.- Dar clic en el botón "Eliminar". 3.- Dar click en el botón "Aceptar". Muestra un mensaje que se quitara el producto de la base de datos.
NOMBRE HISTORIA		Eliminar categorías		
PRUEBA	Crear una prueba unitaria para conocer si la eliminación procedió con normalidad y si el producto fue removido de la base de datos.		COMENTARIOS	Sin comentarios

Figura 9. Historia de Usuario Eliminar Categorías

2.2 Definición de Requerimientos

2.2.1 Especificación de Requerimientos de Software

ACTORES DEL SISTEMA

ACT-01	Administrador
Descripción	Este actor representa al administrador del sistema de inventario (Tienen acceso a todo).
Comentarios	Ninguno

Tabla 5. Actores del sistema

REQUISITOS DEL SISTEMA

REQ001	Registrar producto	
Actor	• ACT-01	
Descripción	El caso de uso empieza cuando un producto nuevo ingresa a la empresa y se desea registrar el producto en la base de datos del sistema de gestión de inventarios.	
Precondición	Iniciar sesión como administrador. Debe de existir la categoría del producto.	
Secuencia normal	Paso	Acción
	1	El actor solicita al sistema iniciar el proceso para registrar un producto.
	2	El sistema despliega los atributos descritos en la tabla de atributos productos que el actor debe llenar para crear un nuevo producto.
	3	El actor asigna una categoría e inserta los datos requeridos del producto. Los datos deben ser ingresados de forma completa a los solicitados por el sistema.
	4	El sistema validará que todos los datos solicitados estén llenos y cumplan con los estándares de validación especificados.
	5	El actor guarda la información en una base de datos segura.
	6	El sistema notificará al actor que el producto ha sido guardado correctamente.
Postcondición	El producto queda registrado en el sistema.	
Excepciones	Paso	Acción
	3	El sistema informa al actor en caso de que no se hayan llenado los datos obligatorios.
	3	Si la categoría del producto a asignar no se encuentra registrada, se debe realizar el caso de

	uso registrar categoría .
3	El sistema deberá validar que el precio ingresado por el actor no supere los 1000 dólares. Además, se deberá validar que la fecha de adquisición del producto no sea más antigua de 1 mes a la fecha en la que se va a registrar el producto.
3	Si el producto ya existe, el sistema emitirá un mensaje de error "Error. El producto ya se encuentra registrado".
4	Si el actor no cumple con los estándares de validación, el sistema emitirá un mensaje de alerta "Los datos ingresados no son correctos" y no le permitirá continuar con el proceso de registro.
6	Si se presentaron problemas de comunicación con la base de datos el sistema notificará "Error. Datos no guardados", a continuación este caso de uso termina.
Importancia	Alta

Tabla 6. Requisito Registrar Producto

REQ002	Enlistar producto	
Objetivos asociados	<ul style="list-style-type: none"> • OBJ-01 Gestionar productos 	
Requisitos asociados	<ul style="list-style-type: none"> • REQ007 Registrar producto 	
Actor	<ul style="list-style-type: none"> • ACT-01 	
Descripción	El caso de uso le permite al actor enlistar los productos que se encuentran registrados en el sistema de gestión de inventarios.	
Precondición	Acceder al sistema como administrador.	
Secuencia normal	Paso	Acción
	1	El actor solicita al sistema iniciar el proceso para listar un producto.
	2	El sistema muestra una ventana para ingresar los datos correspondientes para poder generar la lista.
	3	El actor ingresa los datos solicitados en la ventana del sistema y procede a enviar los datos.
	4	El sistema procesa y valida los datos ingresados anteriormente para generar la lista.
	5	El sistema enlista toda la información de los

		productos que se haya registrado en la base de datos.
	6	El sistema mostrará en una tabla todos los atributos descritos en la tabla de atributos productos , y las operaciones que se pueden realizar con los productos listados.
Postcondición	Se podrán realizar acciones como editar o eliminar productos.	
Excepciones	Paso	Acción
	1	Si todavía no existen productos registrados, se debe realizar el caso de uso registrar producto , a continuación, este caso de uso termina.
	4	Si el sistema no valida
Importancia	Alta	

Tabla 7. Requisito Enlistar Producto

REQ003	Editar producto	
Requisitos asociados	<ul style="list-style-type: none"> • REQ007 Registrar producto 	
Actor	<ul style="list-style-type: none"> • ACT-01 	
Descripción	Este proceso le permite al actor actualizar información de un producto que se encuentra registrado en el sistema de gestión de inventario.	
Precondición	Acceder al sistema como administrador. Conocer la información del producto que se desea editar.	
Secuencia normal	Paso	
	1	El usuario solicita al sistema comenzar el proceso de editar producto.
	2	El administrador selecciona el producto para editar la tabla de atributos productos .
	3	El sistema muestra un mensaje de confirmación para actualizar la información del producto.
	4	Si acepta la confirmación de actualizar información el sistema guardará estos cambios en la base de datos.
	5	Se le notificará al actor que los cambios se realizaron correctamente.
Postcondición	Se obtienen los datos del producto actualizados en el inventario y estarán disponibles los nuevos datos del producto en el stock del sistema.	
Excepciones	Paso	

	2	Si el administrador no selecciona la cédula del usuario debe dar un error y no permitirá continuar con el proceso.
	3	El mensaje de confirmación será “¿Está seguro de actualizar la información?”, mientras que el mensaje de confirmación del vendedor será “¿Está seguro de actualizar su contraseña?”, si el administrador o vendedor no responden o responden con un no el sistema notificará “Su solicitud fue cancelada correctamente ”, a continuación este caso de uso termina.
	5	Si se presentaron problemas de comunicación con la base de datos el sistema notificará “Error. Datos no guardados”, a continuación este caso de uso termina.

Tabla 8. Requisito Editar Producto

REQ004	Eliminar producto	
Requisitos asociados	<ul style="list-style-type: none"> • REQ007 Registrar producto 	
Actor	<ul style="list-style-type: none"> • ACT-01 	
Descripción	Este proceso le permite al actor eliminar un producto de la base de datos del sistema de gestión de inventario	
Precondición	Acceder al sistema como administrador. Conocer el producto que se desea eliminar.	
Secuencia normal	Paso	Acción
	1	El administrador solicita al sistema comenzar el proceso de eliminación de producto.
	2	El administrador selecciona en el menú de administrador el producto a eliminar.
	3	El sistema muestra un mensaje de confirmación para eliminar el producto.
	4	Si acepta la confirmación de eliminar el producto el sistema guardará estos cambios en la base de datos.
	5	Se le notificará al actor que los cambios se realizaron correctamente.
	6	El usuario seleccionado ya no tendrá acceso al sistema.

Postcondición	Se eliminará el producto del inventario y el producto ya no estará disponible en el stock del sistema.	
Excepciones	Paso	Acción
	2	Si el administrador no selecciona la cédula del usuario y el estado el sistema notificará "Error. Seleccione la cédula del usuario" o "Error. Seleccione el estado".
	3	El mensaje de confirmación será "¿Está seguro de eliminar este usuario?", si el administrador no responde o responde con un no el sistema notificará "Su solicitud fue cancelada correctamente", a continuación, este caso de uso termina.
	5	Si se presentaron problemas de comunicación con la base de datos el sistema notificará "Error. Datos no guardados", a continuación, este caso de uso termina.

Tabla 9. Requisito Eliminar Producto

REQ005	Acceder al sistema	
Objetivos asociados	<ul style="list-style-type: none"> • OBJ-03 Gestionar usuarios 	
Actor	<ul style="list-style-type: none"> • ACT-01 	
Descripción	El sistema permitirá el acceso, para empezar el usuario debe ingresar su nombre de usuario y su contraseña. Una vez que el usuario, contraseña y su estado sean validados se le otorga el ingreso al sistema como administrador.	
Precondición	El usuario debe estar registrado previamente en el sistema de gestión de inventarios y tener los privilegios ya sea de administrador.	
Secuencia normal	Paso	Acción
	1	El usuario solicita el acceso al sistema de gestión de inventario.
	2	El sistema solicita al usuario que ingrese su nombre de usuario y la contraseña.
	3	El usuario proporciona al sistema su nombre de usuario y la contraseña.
	4	El sistema valida si el nombre de usuario y la contraseña son correctas, es decir que se estos datos ingresados se encuentren en la base de datos. Además de verificar si el usuario está activo o no.

	5	Si el nombre de usuario y la contraseña son correctas y se encuentra activo el sistema permite el acceso al usuario.
Postcondición		
Excepciones	Paso	Acción
	3	Si el usuario ha intentado tres veces acceder sin éxito ya sea por contraseña o usuario incorrecto, el sistema rechaza el acceso del usuario, a continuación, este caso de uso termina.
	4	Si el nombre de usuario no existe en la base de datos el sistema notificará "Error. Usuario no registrado". Si el usuario se encuentra inactivo el sistema notificará "Error. Usuario inactivo". Si el nombre de usuario es incorrecto se notificará "Error. Nombre de usuario incorrecto". Si la contraseña es incorrecta se notificará "Error. Contraseña incorrecta". Si el nombre de usuario y contraseña son incorrectos se notificará "Error. El nombre de usuario y contraseña son incorrectos".

Tabla 10. Requisito Acceso al sistema

REQ006	Registrar Categoría	
Requisitos asociados	<ul style="list-style-type: none"> REQ001 Registrar Producto 	
Actor	<ul style="list-style-type: none"> ACT-01 	
Descripción	Este proceso permite registrar una nueva categoría en la base de datos del sistema de gestión de inventarios.	
Precondición	Acceder al sistema como administrador. La categoría no se encuentra registrada en el sistema	
Secuencia normal	Paso	Acción
	1	El actor solicita al sistema comenzar el proceso para registrar categoría.
	2	El sistema despliega los atributos código categoría y nombre categoría que el actor debe llenar para crear una nueva categoría.
	3	El actor inserta los datos requeridos de la categoría. Los datos deben ser ingresados de forma completa a los solicitados por el sistema.

	4	El sistema validará que todos los datos solicitados estén llenos y cumplan con los estándares de validación especificados.
	5	El actor guarda la información en una base de datos segura.
	6	El sistema notificará al actor que la categoría ha sido guardada correctamente.
Postcondición		La categoría queda registrada en el sistema.
Excepciones	Paso	Acción
	3	El sistema informa al actor en caso de que no se hayan llenado los datos obligatorios.
	3	Si la categoría ya existe, el sistema emitirá un mensaje de error "Error". La categoría ya se encuentra registrada".
	4	Si el actor no cumple con los estándares de validación, el sistema emitirá un mensaje de alerta "Los datos ingresados no son correctos" y no le permitirá continuar con el proceso de registro.
	6	Si se presentaron problemas de comunicación con la base de datos el sistema notificará "Error. Datos no guardados", a continuación, este caso de uso termina.

Tabla 11. Requisito Registrar Categoría

REQ007	Enlistar Categoría	
Requisitos asociados	<ul style="list-style-type: none"> REQ006 Registrar Categoría 	
Actor	<ul style="list-style-type: none"> ACT-01 	
Descripción	Este proceso le permite al actor enlistar las categorías que se encuentran registradas en la base de datos del sistema de gestión de inventario.	
Precondición	Acceder al sistema como administrador.	
Secuencia normal	Paso	Acción
	1	El actor solicita al sistema iniciar el proceso para enlistar una categoría.

	2	El sistema enlista toda la información de las categorías que se haya registrado en la base de datos.
	3	El sistema mostrará en una tabla todas las categorías con sus atributos, nombre, código y las operaciones que se pueden realizar con las categorías listadas.
Postcondición	Se podrá realizar acciones como editar o eliminar categorías.	
Excepciones	Paso	Acción
	1	Si todavía no existen categorías registradas, se debe realizar el caso de uso registrar categoría , a continuación, este caso de uso termina.

Tabla 12. Requisito Enlistar Categoría

REQ008	Enlistar Categoría	
Requisitos asociados	<ul style="list-style-type: none"> • IRQ-02 Información sobre nuevas categorías 	
Actor	<ul style="list-style-type: none"> • ACT-01 	
Descripción	Este proceso le permite al actor enlistar las categorías que se encuentran registradas en la base de datos del sistema de gestión de inventario.	
Precondición	Acceder al sistema como administrador.	
Secuencia normal	Paso	Acción
	1	El actor solicita al sistema iniciar el proceso para enlistar una categoría.
	2	El sistema enlista toda la información de las categorías que se haya registrado en la base de datos.
	3	El sistema mostrará en una tabla todas las categorías con sus atributos, nombre, código y las operaciones que se pueden realizar con las categorías listadas.
Postcondición	Se podrá realizar acciones como editar o eliminar categorías.	
Excepciones	Paso	Acción
	1	Si todavía no existen categorías registradas, se debe realizar el caso de uso registrar categoría , a continuación, este caso de uso termina.

Tabla 13. Requisito Eliminar Categoría

REQ009	Editar Categoría
---------------	-------------------------

Objetivos asociados	<ul style="list-style-type: none"> ● OBJ-02 Gestionar categorías 	
Requisitos asociados	<ul style="list-style-type: none"> ● IRQ-02 Información sobre nuevas categorías 	
Actor	<ul style="list-style-type: none"> ● ACT-01 	
Descripción	Este proceso le permite al actor editar una categoría que se encuentra registrada en la base de datos del sistema de gestión de inventario.	
Precondición	Acceder al sistema como administrador. Conocer la información de la categoría que se desea editar.	
Secuencia normal	Paso	Acción
	1	El actor selecciona una categoría para actualizar su información.
	2	El sistema despliega la información de la categoría seleccionada, con la información de este para que pueda ser modificada.
	3	El actor inserta los datos que quiere actualizar de la categoría escogida.
	4	El sistema valida que los datos ingresados sean correctos.
	5	El administrador presiona el botón de guardar para almacenar los cambios realizados.
	6	El sistema notificará al actor que la categoría ha sido actualizada correctamente.
Postcondición	La categoría actualizada queda registrada en el sistema.	
Excepciones	Paso	Acción
	1	Debe existir la categoría a ser modificada o actualizada.
	3	El sistema informa al actor en caso de que no se hayan llenado los datos obligatorios.
	6	Si los datos de categoría no se guardaron correctamente, el sistema emitirá un mensaje de error "Error. La categoría no se actualizó, intente de nuevo".

Tabla 14. Requisito Editar Categoría

Requisitos Funcionales

2.2.2 Especificación de Casos de Uso

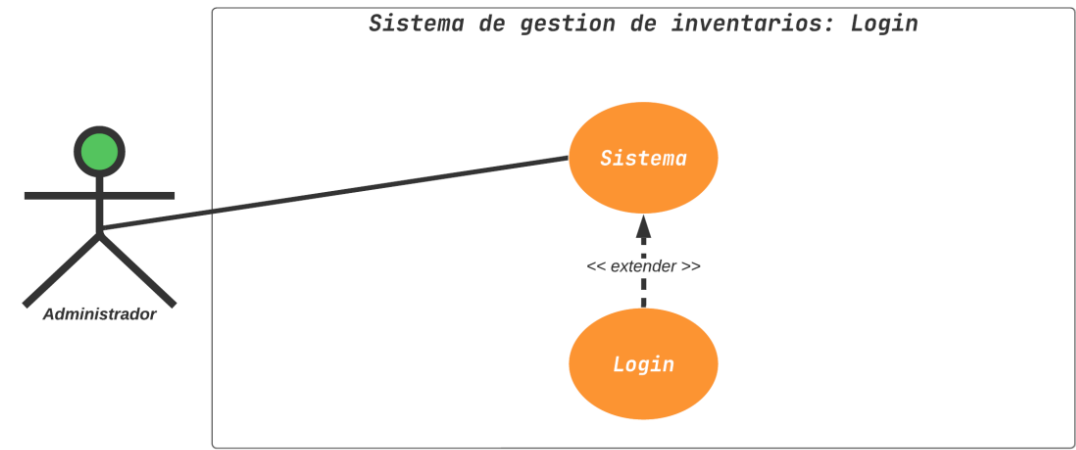


Figura 10. Diagrama de casos de uso Login
Diagrama Elaborado por los Autores

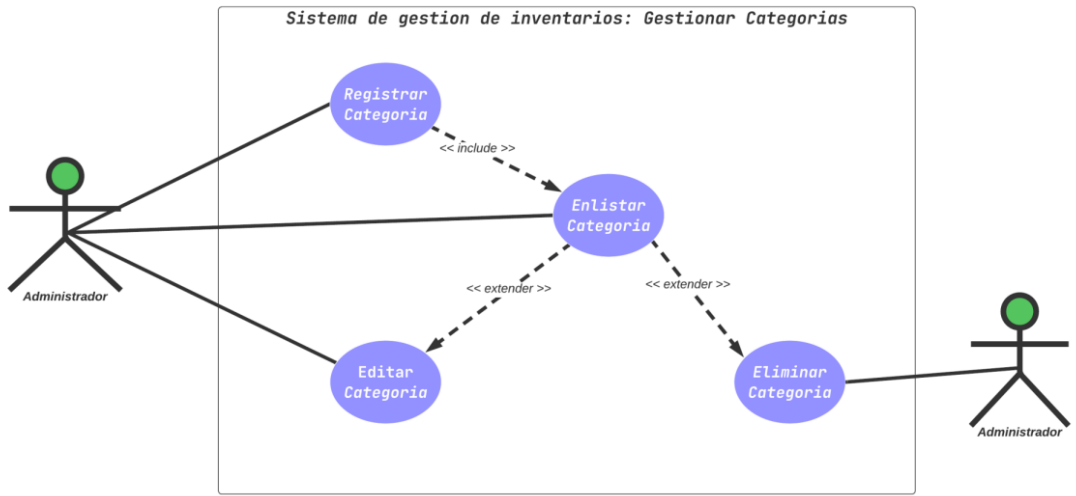


Figura 11. Diagrama de casos de uso Gestionar Categorías
Diagrama Elaborado por los Autores

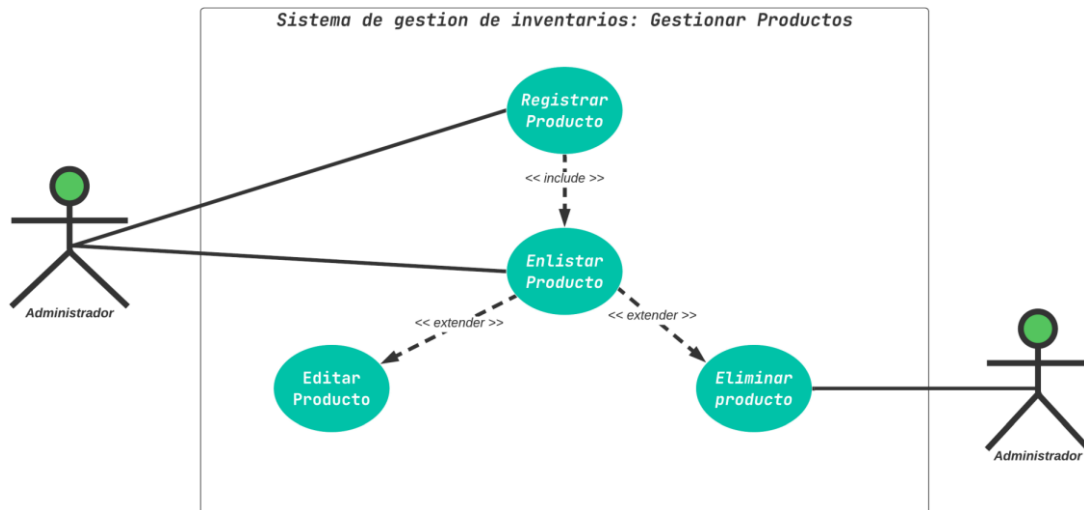


Figura 12. Diagrama de casos de uso Gestionar Productos
Diagrama Elaborado por los Autores

2.3 Análisis y Diseño

2.3.1 Modelo Conceptual

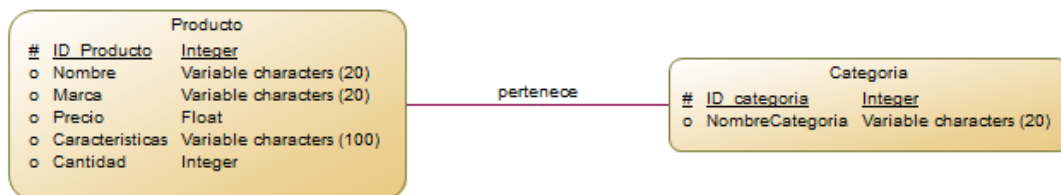


Figura 13. Diagrama Conceptual
Diagrama Elaborado por los Autores

2.3.2 Modelo Lógico

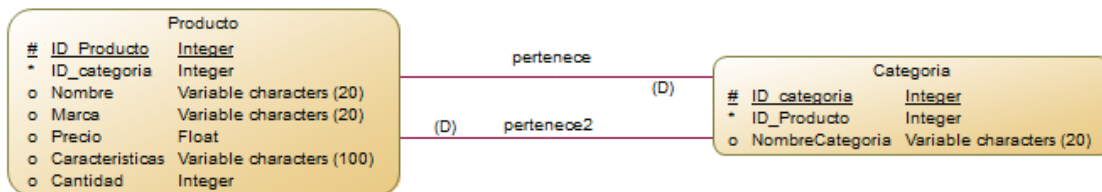


Figura 14. Diagrama Lógico
Diagrama Elaborado por los Autores

2.3.3 Modelo Físico

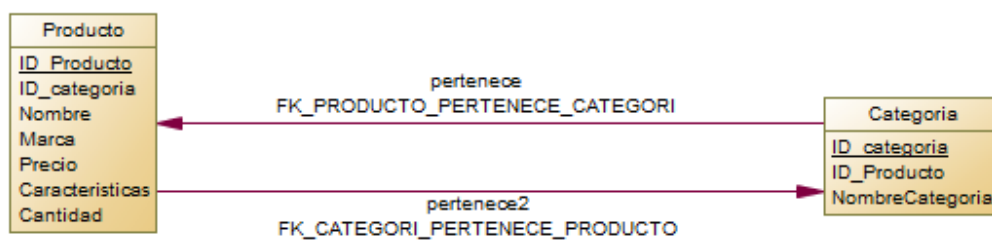


Figura 15. Diagrama físico
Diagrama Elaborado por los Autores

2.3.4 Script de la Base de Datos

Conexión con MongoDB

```
const mongoose = require("mongoose");

const databaseConnection = () => {
  const DB_URI = process.env.DB_URI;
  mongoose.connect(
    DB_URI,
    {
      useNewUrlParser: true,
      useUnifiedTopology: true,
    },
    (error, res) => {
      if (!error) {
        console.log("CONEXION CORRECTA");
      } else {
        console.log("ERROR");
      }
    }
  );
};

module.exports = databaseConnection;
```

CRUD Producto

```
const { matchedData } = require("express-validator");
const { handleHttpError } = require("../utils/handleErrors");
const { productsModel } = require("../models");

/**
 * Obtener lista de la base de datos
 * @param {*} req
 * @param {*} res
 */
const getItems = async (req, res) => {
  try {
    const data = await productsModel.find({});
    res.send({ data });
  } catch (error) {
    handleHttpError(res, error);
  }
};

/**
 * Obtener detalle
 * @param {*} req
 * @param {*} res
 */
const getItemByID = async (req, res) => {
```

```

    try {
      const data = await productsModel.findOne({
        idProducto: req.params.idProducto,
      });
      res.send({ data });
    } catch (error) {
      handleHttpError(res, error);
    }
  };

  /**
   * Crear registro
   * @param {*} req
   * @param {*} res
   */
  const createItem = async (req, res) => {
    try {
      const body = req.body;
      console.log(body);
      const data = await productsModel.create(body);
      res.send({ data });
    } catch (e) {
      handleHttpError(res, e);
    }
  };

  /**
   * Actualizar registro
   * @param {*} req
   * @param {*} res
   */
  const updateItem = async (req, res) => {
    try {
      const { idProducto, ...updatedProductData } = req.body;

      const data = await productsModel.findOneAndUpdate(
        { idProducto: req.params.idProducto },
        { ...updatedProductData, idProducto: req.params.idProducto }
      );
    }
  };

```

```

        res.send({ data });
    } catch (e) {
        handleHttpError(res, e);
    }
};

/**
 * Eliminar registro
 * @param {*} req
 * @param {*} res
 */
const deleteItem = async (req, res) => {
    try {
        const { idProducto } = req.params;
        const data = await productsModel.deleteOne({ idProducto });
        res.send({ data });
    } catch (e) {
        handleHttpError(res, e);
    }
};

/**
 * Eliminar registro (logico)
 * @param {*} req
 * @param {*} res
 */
const deleteItemLogic = async (req, res) => {
    req = matchedData(req);
    const { id } = req;
    const data = await productsModel.delete({ _id: id });
    res.send({ data });
};

module.exports = {
    getItems,
    getItemByID,
    createItem,

```

```

    updateItem,

    deleteItem,

    deleteItemLogic,
  };

```

CRUD Categoría

```

const { matchedData } = require("express-validator");
const { handleHttpError } = require("../utils/handleErrors");
const { categoryModel } = require("../models");

/**
 * Obtener lista de la base de datos
 * @param {*} req
 * @param {*} res
 */
const.getItems = async (req, res) => {
  try {
    const data = await categoryModel.find({});
    res.send({ data });
  } catch (error) {
    handleHttpError(res, error);
  }
};

/**
 * Obtener detalle por ID
 * @param {*} req
 * @param {*} res
 */
const.getItemByID = async (req, res) => {
  try {
    const data = await categoryModel.findOne({
      idCategoría: req.params.idCategoría,
    });
    res.send({ data });
  } catch (error) {
    handleHttpError(res, error);
  }
};

/**
const getItemByName = async (req, res) => {
  try {
    const data = await categoryModel.findOne({ name:
req.params.name });
    res.send({ data });
  } catch (error) {
    handleHttpError(res, error);
  }
};

```

```

*/
/**
 * Crear registro
 * @param {*} req
 * @param {*} res
 */
const createItem = async (req, res) => {
  try {
    const body = req.body;
    console.log(body);
    const data = await categoryModel.create(body);
    res.send({ data });
  } catch (e) {
    handleHttpError(res, e);
  }
};

/**
 * Actualizar registro
 * @param {*} req
 * @param {*} res
 */
const updateItem = async (req, res) => {
  try {
    const updatedCategory = {
      idCategoria: req.body.idCategoria,
      nombre: req.body.nombre,
    };

    const data = await categoryModel.findOneAndUpdate(
      { idCategoria: req.params.idCategoria },
      updatedCategory,
      { new: true }
    );
    res.send({ data });
  } catch (error) {
    res.status(500).send({ error: error.message });
  }
};

/**
 * Eliminar registro
 * @param {*} req
 * @param {*} res
 */
const deleteItem = async (req, res) => {
  try {
    const data = await categoryModel.deleteOne({
      idCategoria: req.params.idCategoria,
    });
    res.send({ data });
  } catch (e) {

```

```
    handleError(res, e);
  }
};

/**
 * Eliminar registro (logico)
 * @param {*} req
 * @param {*} res
 */
const deleteItemLogic = async (req, res) => {
  req = matchedData(req);
  const { id } = req;
  const data = await categoryModel.delete({ _id: id });
  res.send({ data });
};

module.exports = {
  getItems,
  getItemByID,
  createItem,
  updateItem,
  deleteItem,
  deleteItemLogic,
};
```

2.4.1 Modelo de Arquitectura General

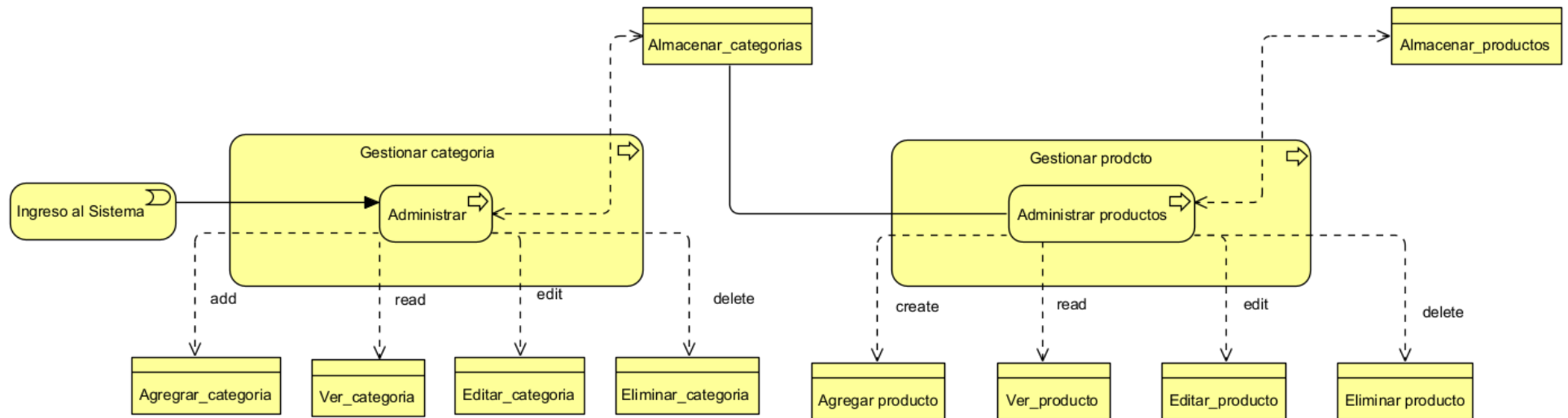


Figura 16. Diagrama de arquitectura del sistema de inventarios Bloz Cell
Diagrama Elaborado por los Autores

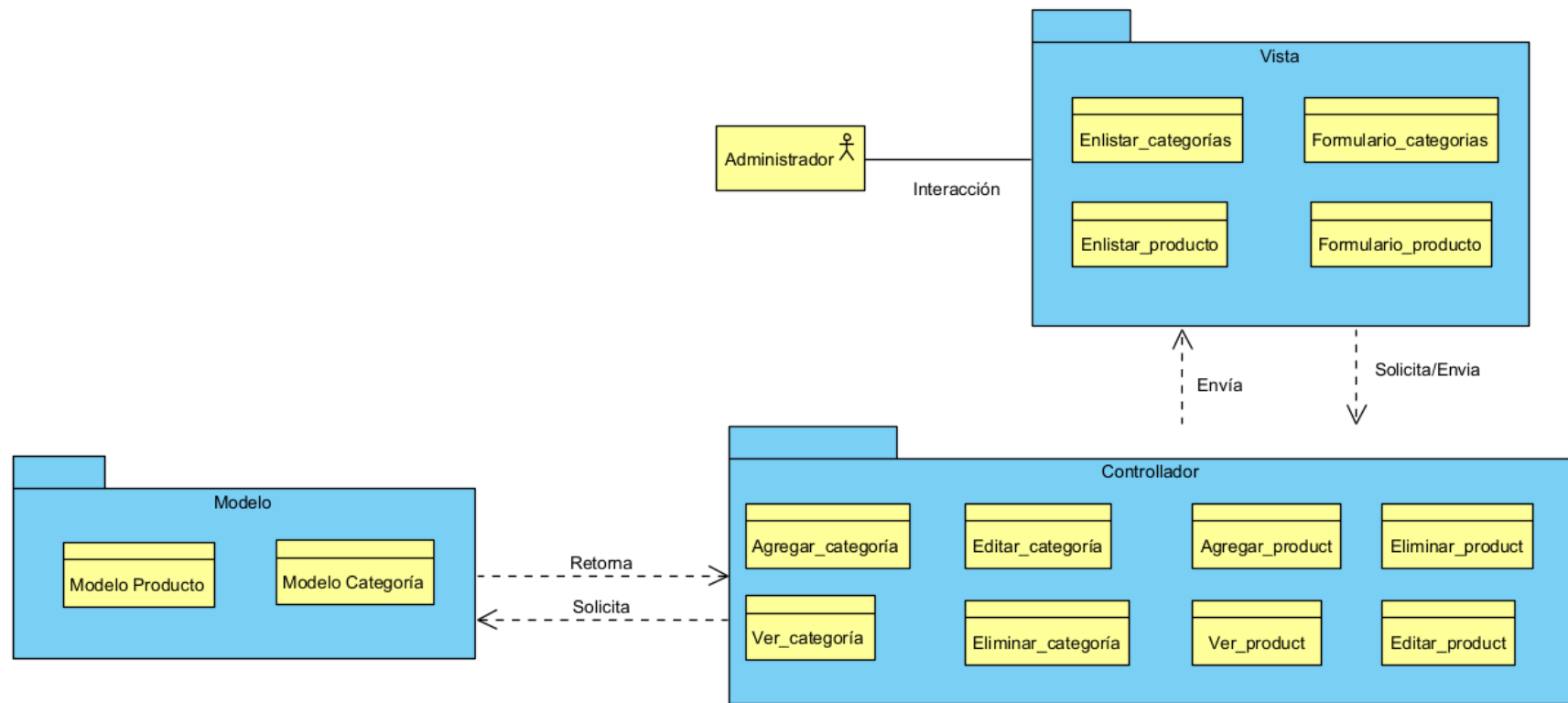


Figura 17. Diagrama de arquitectura patrón de diseño del sistema de inventarios Bloz Cell
Diagrama Elaborado por los Autores

2.4.2 Modelo de Arquitectura Cliente-Servidor

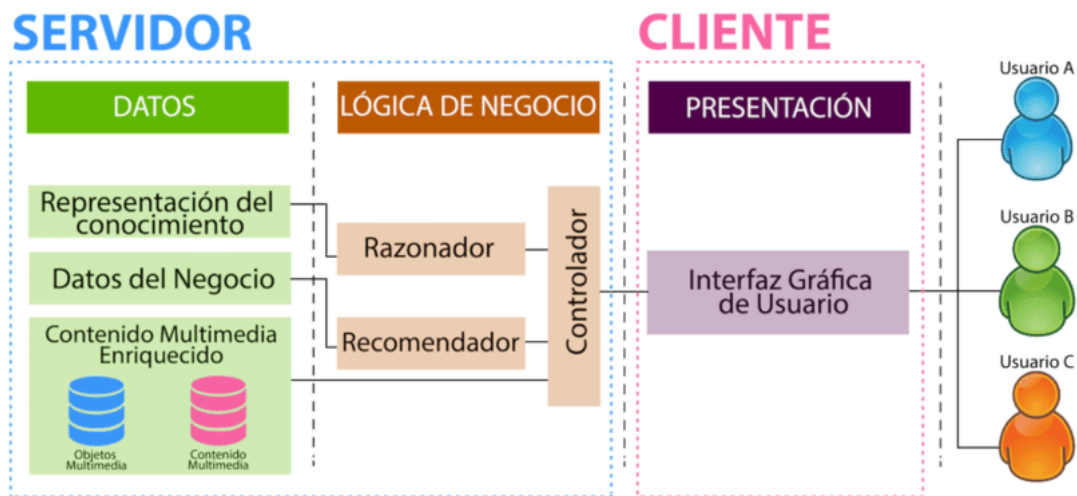


Figura 18. Arquitectura Web

Diagrama elaborado por Hernández, Sebastian & Mancilla, Daniela & Narvaez, Cristian & Iregui, Marcela. (2013). Caso de estudio para validar la representación del conocimiento en aplicaciones web centradas en el usuario por medio de contenido multimedia enriquecido.

CAPÍTULO III

3.1 Pruebas

3.1.1 Prueba de Caja Negra

El resultado de las pruebas de Caja Negra realizadas al sistema se refleja en el documento adjunto “[W G3_CajaNegra_V3.0.docx](#)”.

3.1.2 Prueba de Caja Blanca

El resultado de las pruebas de Caja Blanca realizadas al sistema se refleja en el documento adjunto “[W G3_CajaBlanca_V3.0.docx](#)”.

3.1.3 Documentación de Informe de Errores

Función de la Plantilla

La función de esta plantilla es estandarizar y formalizar todos los aspectos que conformarán un reporte de errores de las pruebas que se aplicaron al sistema de gestión de inventarios para el emprendimiento Bloz Cell.

Reporte de Errores e Inconsistencias			
Nombre del Proyecto:	Bloz Cell		
Fecha de pruebas:	25-01-2023		
Módulos:	Gestión de Productos.		
Analista:	Jonathan Zapata		
Responsable:	Jhonatan Lituma		
Fecha de revisión:	25-01-2023		
Identificación Caso Prueba	Descripción de prueba.	Descripción del error.	Acciones de corrección
CP-001	Prueba de caja blanca de	Dentro del diagrama de flujo el primer nodo no	Aumentar la variable let

	actualizar registro	se especifica la variable, let updatedProduct	updatedProduct dentro del diagrama de flujo
CP-002	Prueba caja negra Caso de Prueba	La condición de entrada nombre permite el ingreso de números de modo que contradice a la documentación.	Insertar la respectiva validación de solo caracteres
CP-003	Prueba caja negra Caso de Prueba	La condición de entrada marca permite el ingreso de números de modo que contradice a la documentación.	Insertar la respectiva validación de solo caracteres
CP-004	Prueba caja negra Caso de Prueba	La entrada 0 en precio es permitida por el sistema de modo que contradice la documentación	Insertar la respectiva validación de precio<0
CP-005	Prueba caja negra Caso de Prueba	La entrada de cantidad permite el ingreso de números no enteros de modo que contradice la documentación	Insertar la respectiva validación de cantidad solo números enteros

Tabla 14. Reporte de errores
Diagrama Elaborado por Jonatan Zapata (Líder Grupo 5)

3.1.3.1 Enlace: video demostrativo de evaluación por pares

- [ReunionPruebas.mp4 \(sharepoint.com\)](#)

3.1.4 Documentación Técnicas Caja Blanca y Caja Negra

3.1.4.1 TÉCNICA DE CAJA BLANCA: Cobertura de caminos

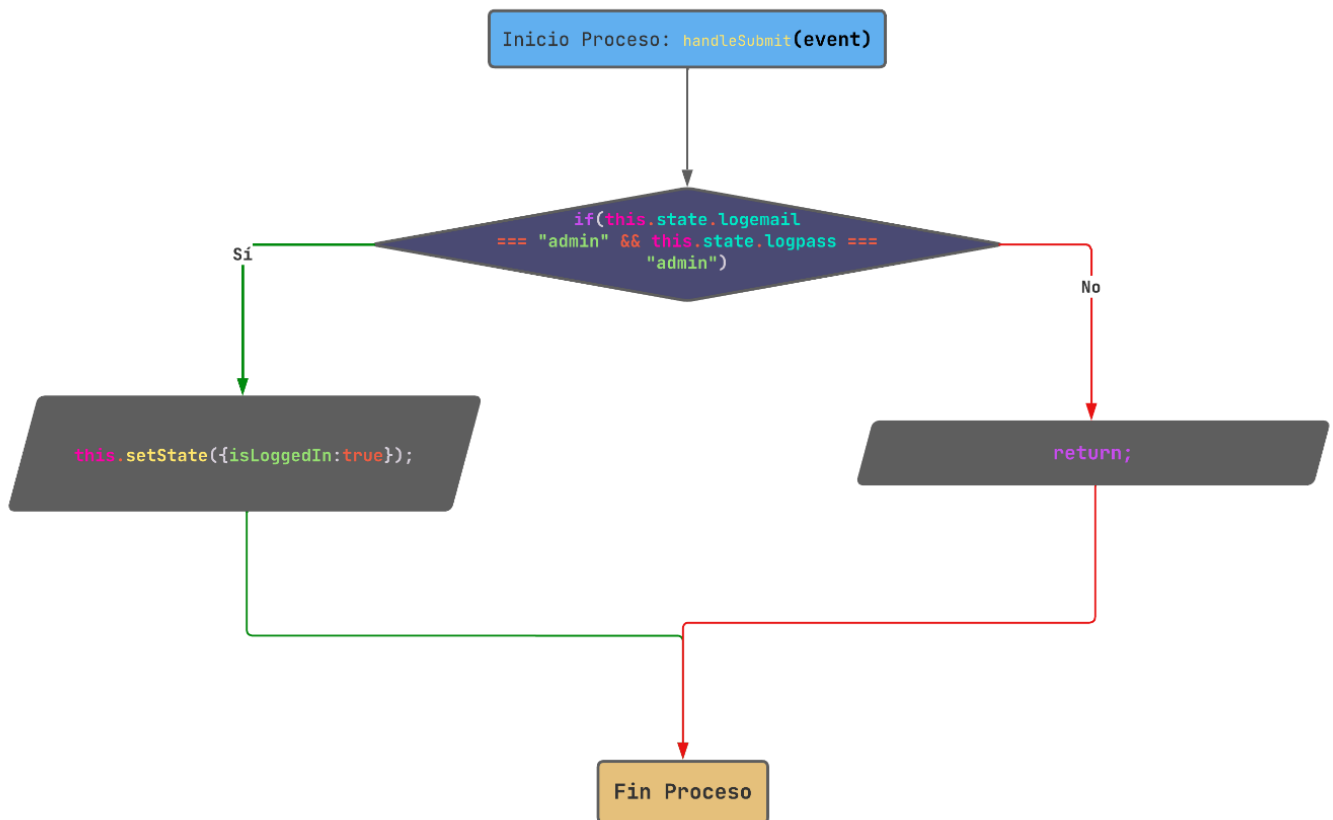
LOGIN

1. PRUEBA CAJA BLANCA VALIDAR LOGIN

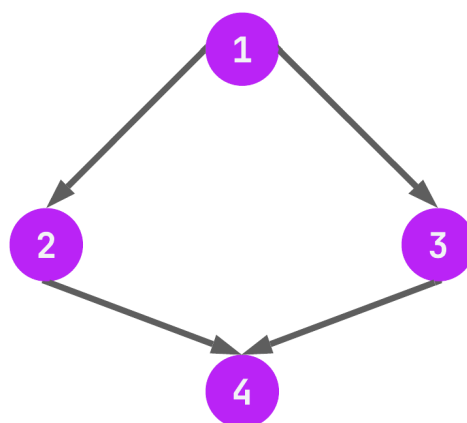
- Código

```
handleSubmit = (event) => {
  event.preventDefault();
  if (this.state.logemail === "admin" && this.state.logpass === "admin") {
    this.setState({ isLoggedIn: true });
  }
};
```

- Diagrama de flujo



- Diagrama de grafos



- Rutas

R1: 1, 2, 4

R2: 1, 3, 4

- Complejidad Ciclomática

E: Número de aristas

N: Número de nodos

P: Número de nodos prediado

$$V(G) = E - N + 2$$

$$V(G) = 4 - 4 + 2$$

$$V(G) = 2$$

$$V(G) = P + 1$$

$$V(G) = 1 \text{ nodo prediado} + 1 = 2$$

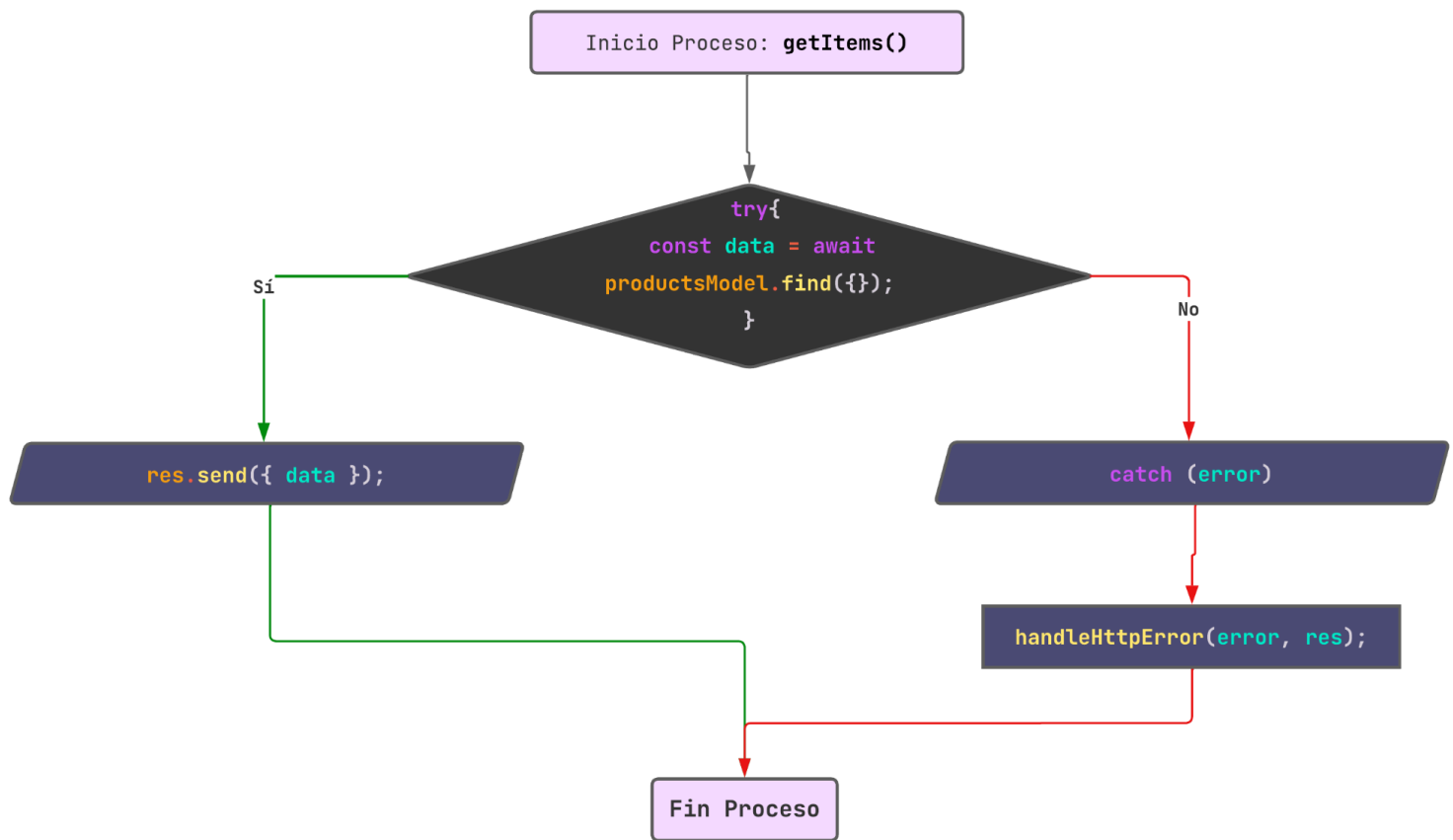
GESTIONAR PRODUCTOS

1. PRUEBA CAJA BLANCA OBTENER LISTA DE REGISTROS

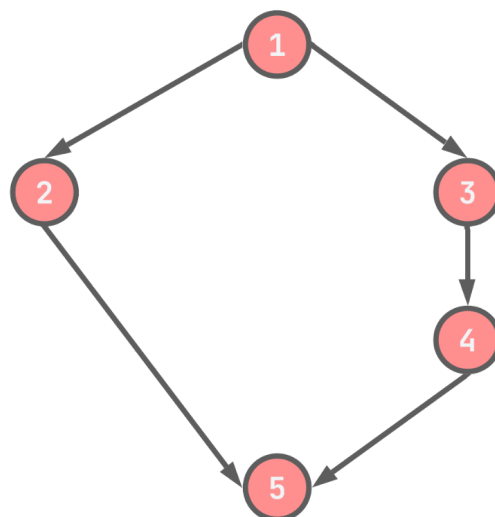
- Código

```
/**
 * Obtener la lista de la base de datos
 * @param {*} req
 * @param {*} res
 */
const getItems = async (req, res) => {
  try {
    const data = await productsModel.find({});
    res.send({ data });
  } catch (error) {
    handleHttpError(error, res);
  }
};
```

- Diagrama de flujo:



- **Diagrama de grafos:**



- **RUTAS:**

R1: 1, 3, 4, 5

R2: 1, 2, 5

- **Complejidad Ciclomática:**

E: 5 (try, await, send, catch, handleHttpError)

N: 5 (try, await, send, catch)

P: 1 (try)

$V(G) = E - N + 2$

$V(G) = 5 - 5 + 2$

$V(G) = 2$

$V(G) = P + 1$

$V(G) = 1 + 1 = 2$

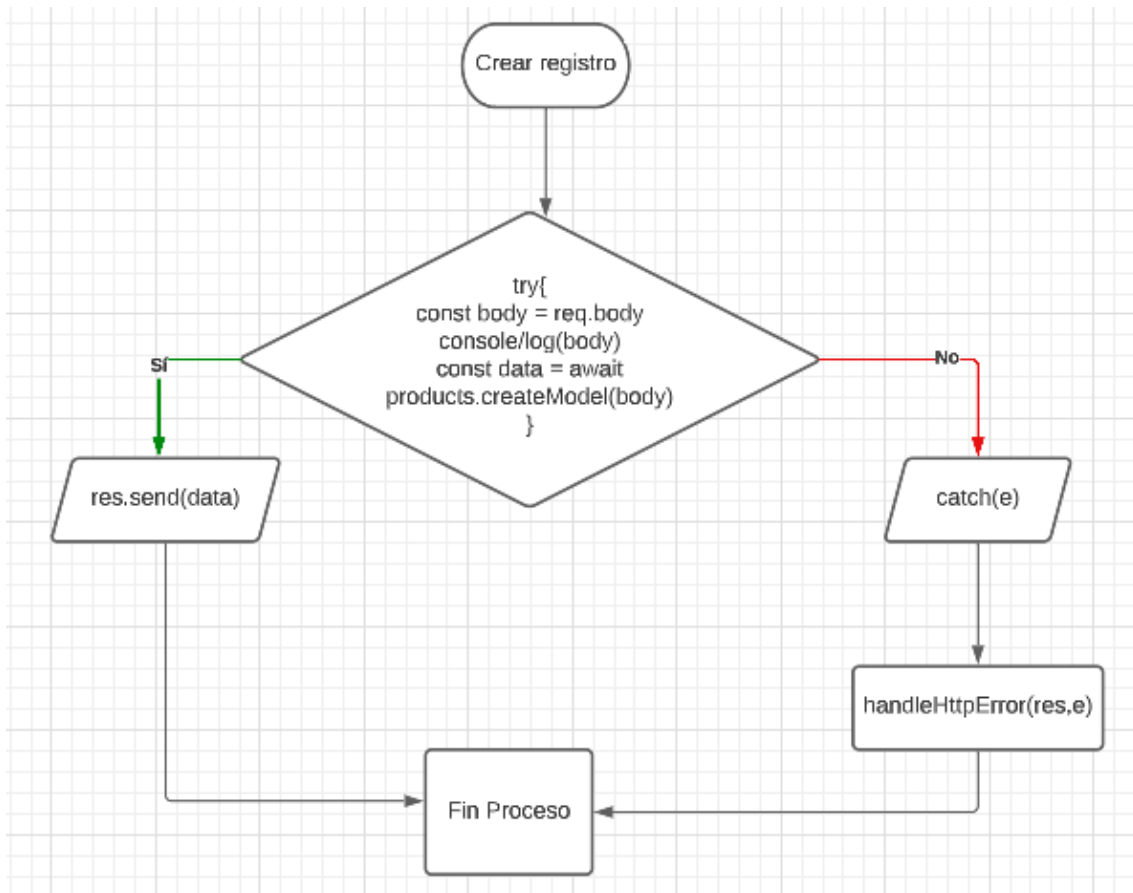
En conclusión, el código tiene una complejidad ciclomática de 2, lo que significa que hay 2 caminos diferentes a través del código. Esto indica que el código es bastante simple y fácil de entender y probar.

2. PRUEBA CAJA BLANCA CREAR REGISTRO

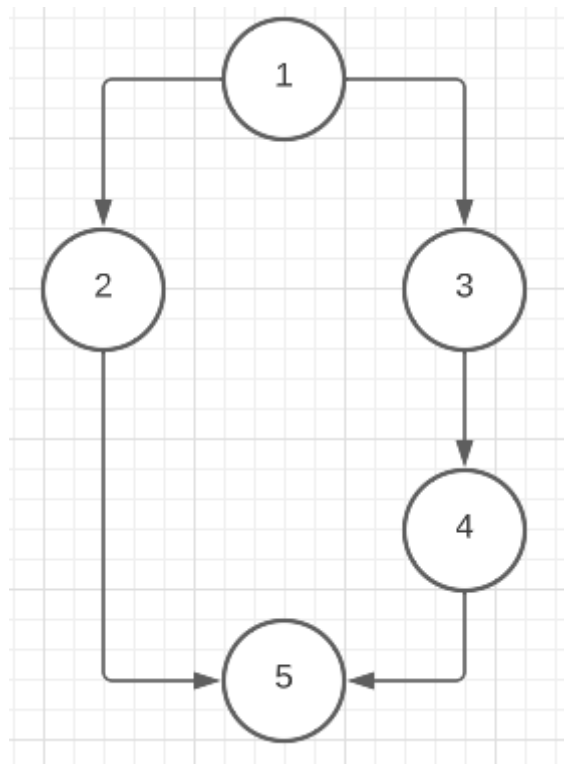
- Código

```
/**
 * Crear Registro
 * @param {*} req
 * @param {*} res
 */
const CreateItem = async (req, res) => {
  try {
    const body = req.body;
    console.log(body);
    const data = await productsModel.create(body);
    res.send({ data });
  } catch (error) {
    handleHttpError(error, res);
  }
}
```

- Diagrama de flujo



- Diagrama de grafos



- Rutas

R1: 1, 2, 5

R2: 1, 3, 4, 5

- Complejidad Ciclomática

E: Número de aristas

N: Número de nodos

P: Número de nodos predicado

$$V(G) = E - N + 2$$

$$V(G) = 5 - 5 + 2$$

$$V(G) = 2$$

$$V(G) = P + 1$$

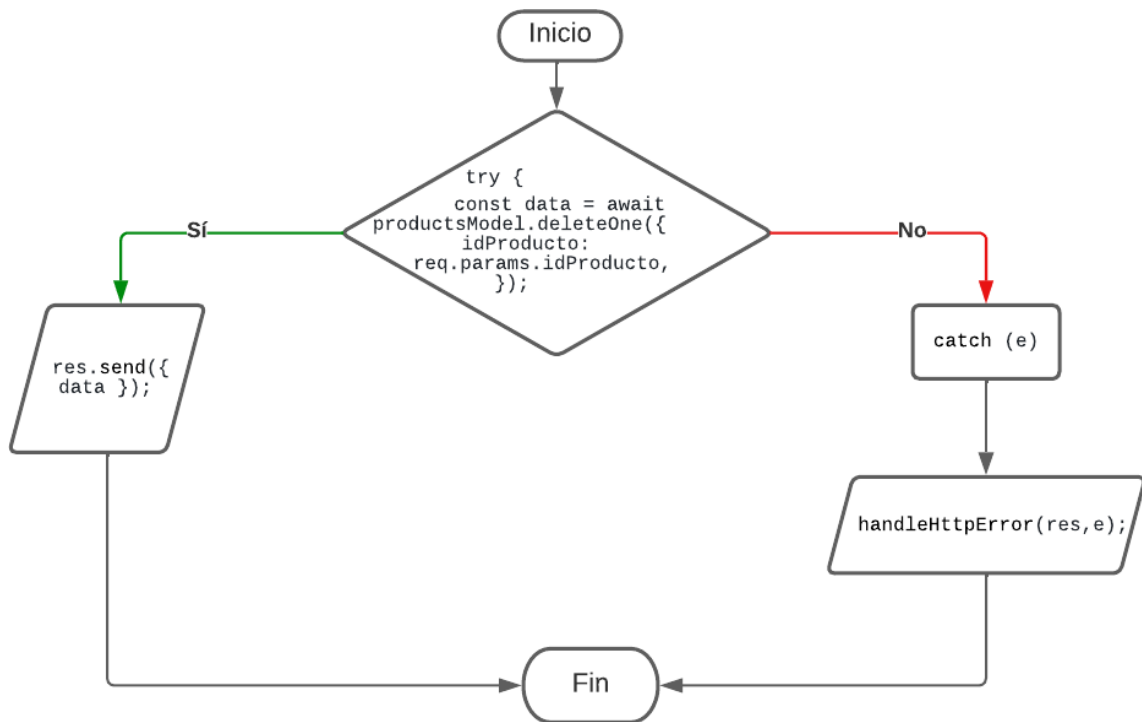
$$V(G) = 1 \text{ nodo predicado} + 1 = 2$$

3. PRUEBA DE CAJA BLANCA DE ELIMINAR REGISTRO

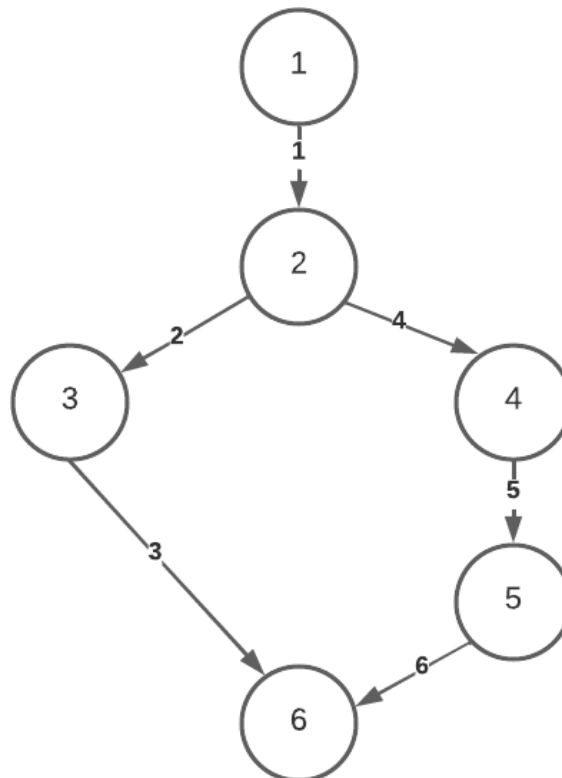
- Código

```
/**
 * Eliminar registro
 * @param { * } req
 * @param {*} res
 */
const deleteItem = async (req, res) => {
  try {
    const data = await productsModel.deleteOne({
      idProducto: req.params.idProducto,
    });
    res.send({ data });
  } catch (e) {
    handleHttpError(res, e);
  }
};
```

- Diagrama de flujo



- **Diagrama de grafos**



- **Rutas**

R1: 1, 2, 3, 6

R2: 1, 2, 4, 5, 6

- **Complejidad Ciclomática**

E: Número de aristas 6

N: Número de nodos 6

P: Número de nodos predicado 1

$$V(G) = E - N + 2A$$

$$V(G) = 6 - 6 + 2 = 2$$

$$V(G) = P + 1$$

$$V(G) = 1 + 1 = 2$$

4. PRUEBA DE CAJA BLANCA DE ACTUALIZAR REGISTRO

```
/**
Actualizar registro
@param { * } req
@param {*} res
*/
const updateItem = async (req, res) => {
  try {
    let updatedProduct = {
      idProducto: req.body.idProducto,
      nombre: req.body.nombre,
      marca: req.body.marca,
      modelo: req.body.modelo,
      precio: req.body.precio,
      características: req.body.características,
      imagen: req.body.image,
      cantidad: req.body.cantidad,
      categoria: req.body.categoria,
    };
    const data = await productsModel.findOneAndUpdate(
      {
        idProducto: req.params.idProducto,
      },
      updatedProduct
    );
    res.send({ data });
  } catch (e) {
    handleHttpError(res, e);
  }
};
```

DIAGRAMA DE FLUJO

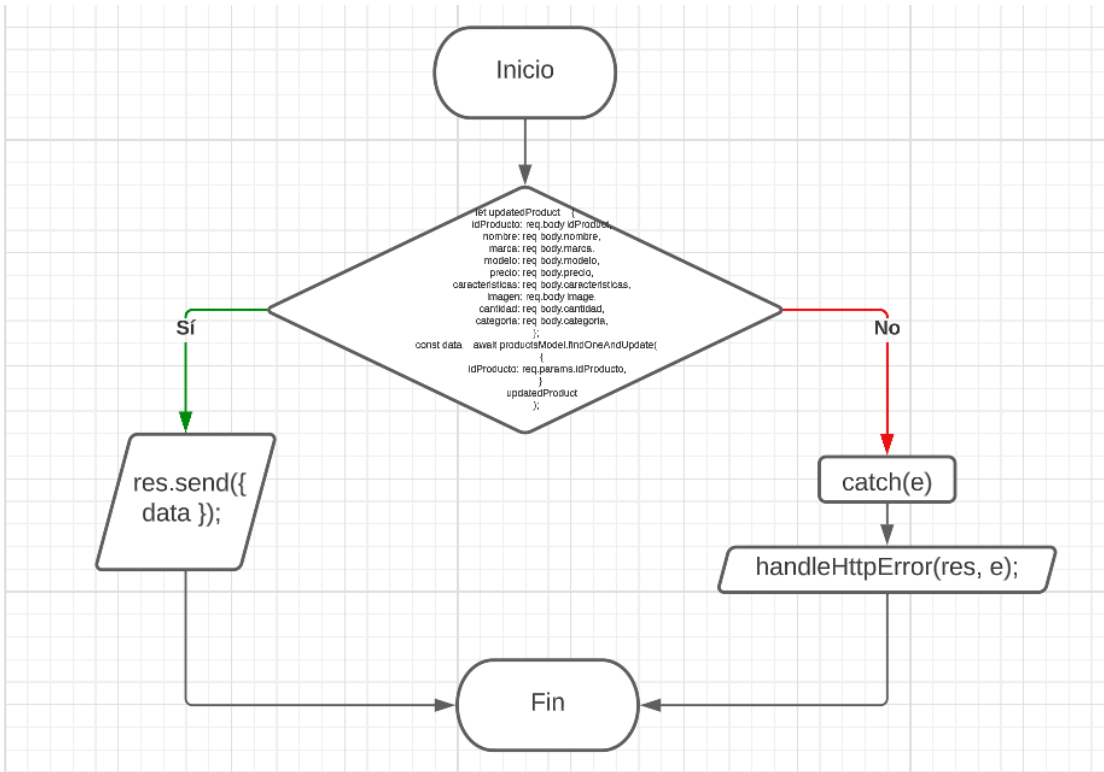
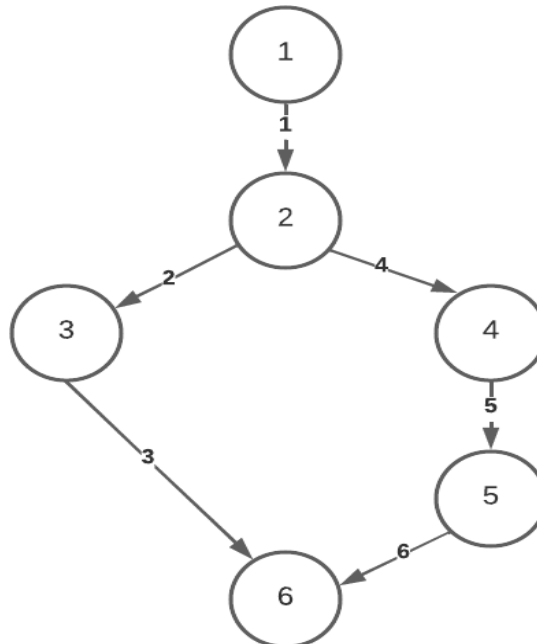


DIAGRAMA DE GRAFOS



RUTAS:

R1: 1, 2, 3, 6

R2: 1, 2, 4, 5, 6

Complejidad Ciclomática

E: Número de aristas

N: Número de nodos

P: Número de nodos predicado

$$V(G) = E - N + 2$$

$$V(G) = 6 - 6 + 2 = 2$$

$$V(G) = P + 1$$

$$V(G) = 1 + 1 = 2$$

Conclusión:

Las pruebas de caja blanca proporcionan una gran cantidad de información sobre el comportamiento interno de un sistema o aplicación. Al analizar el código ejecutado durante las pruebas, los desarrolladores pueden identificar problemas de rendimiento, bugs y otros problemas que podrían afectar la estabilidad y el rendimiento del sistema.

GESTIONAR CATEGORÍAS

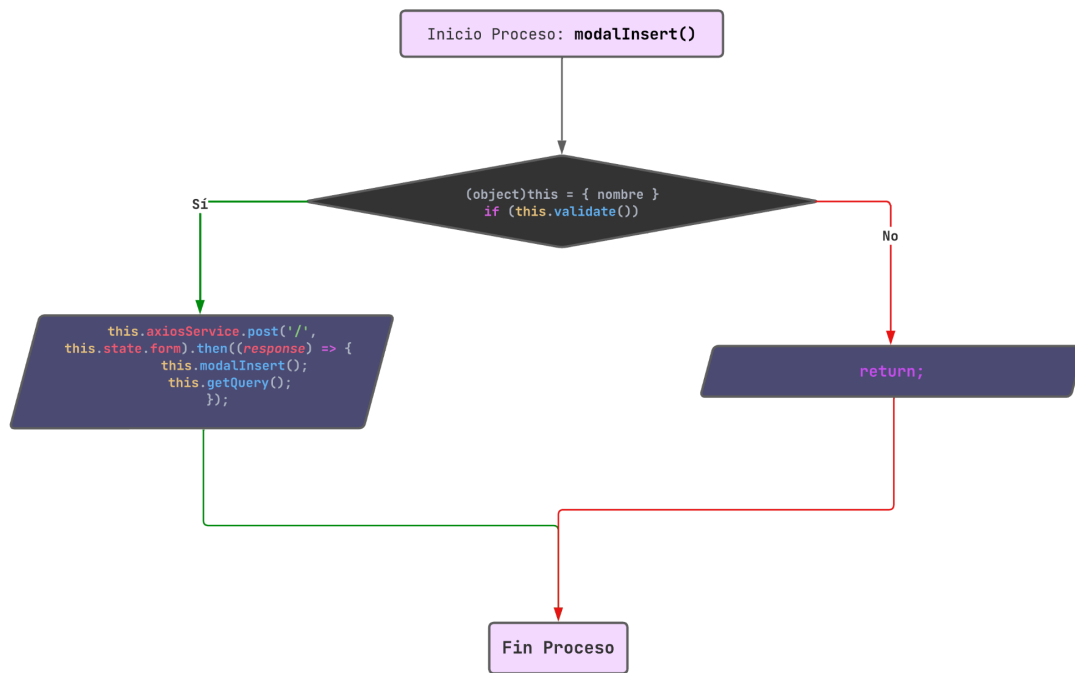
1. PRUEBA CAJA BLANCA CREAR CATEGORÍA

- Código

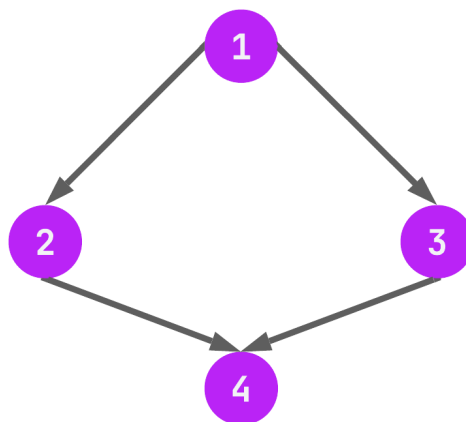
```
validate = () => {
  const { form } = this.state;
  const errors = {};
  //Nombre
  if (!this.validateLetters(form.nombre)) {
    errors.nombre = 'No se permiten números.';
  }
  if (!form.nombre) {
    errors.nombre = 'Campo requerido.';
  }
  this.setState({ errors });
  return Object.keys(errors).length === 0;
};

postQuery = async () => {
  if (this.validate()) {
    this.authService.post('/', this.state.form).then((response)
=> {
      this.modalInsert();
      this.getQuery();
    });
  }
};
```

- Diagrama de flujo



- Diagrama de grafos



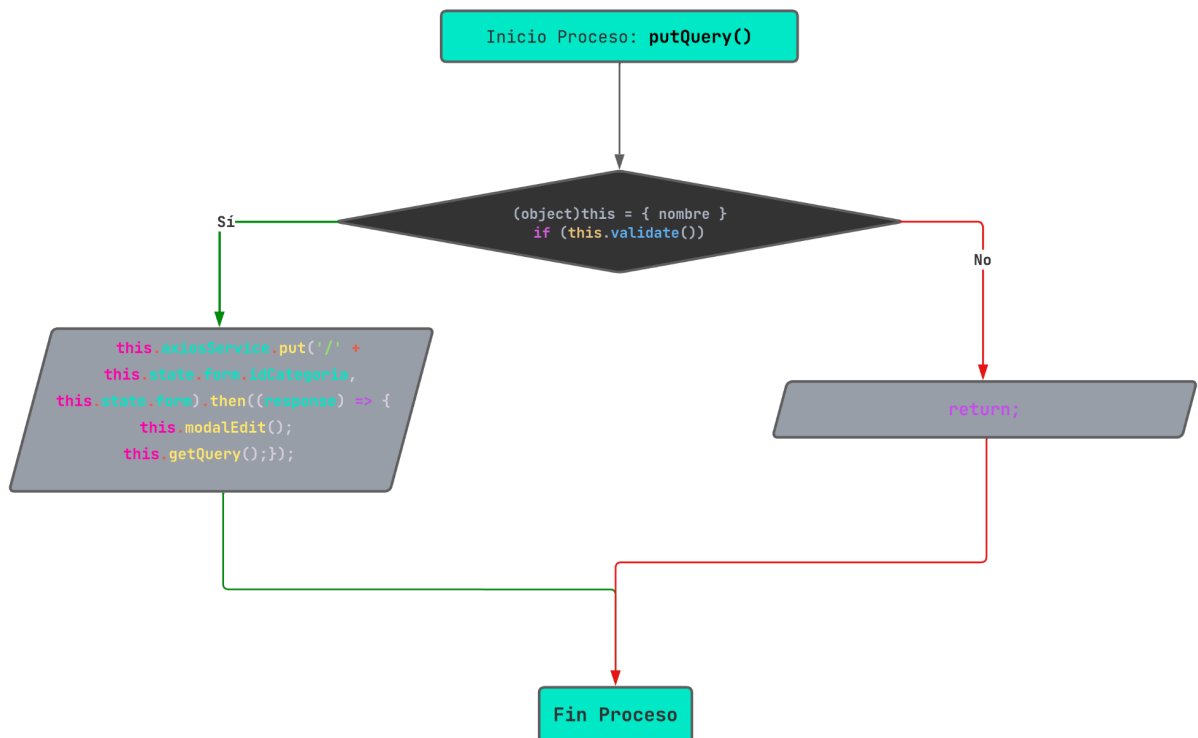
- Rutas
 - R1: 1, 2, 4
 - R2: 1, 3, 4
- Complejidad Ciclomática
 - E: Número de aristas
 - N: Número de nodos
 - P: Número de nodos predicado
 - $V(G) = E - N + 2$
 - $V(G) = 4 - 4 + 2$
 - $V(G) = 2$
 - $V(G) = P + 1$
 - $V(G) = 1 \text{ nodo predicado} + 1 = 2$

1. PRUEBA CAJA BLANCA EDITAR CATEGORÍA

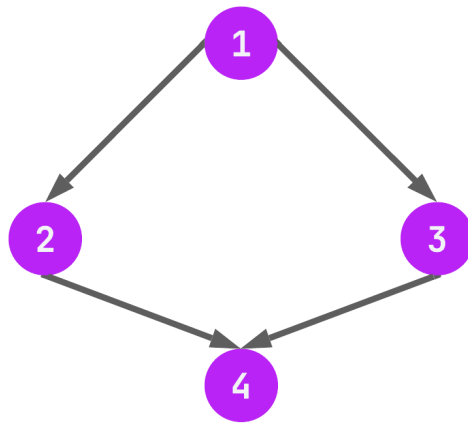
- Código

```
putQuery = () => {  
    if (this.validate()) {  
        this.axiosService  
            .put('/') + this.state.form.idCategoria,  
this.state.form)  
            .then((response) => {  
                this.modalEdit();  
                this.getQuery();  
            });  
    }  
};
```

- Diagrama de flujo:



- Diagrama de grafos



- **Rutas**
 R1: 1, 2, 4
 R2: 1, 3, 4
- **Complejidad Ciclomática**
 E: Número de aristas
 N: Número de nodos
 P: Número de nodos predicado
 $V(G) = E - N + 2$
 $V(G) = 4 - 4 + 2$
 $V(G) = 2$
 $V(G) = P + 1$
 $V(G) = 1 \text{ nodo predicado} + 1 = 2$

2. PRUEBA DE CAJA BLANCA DE ELIMINAR CATEGORÍA

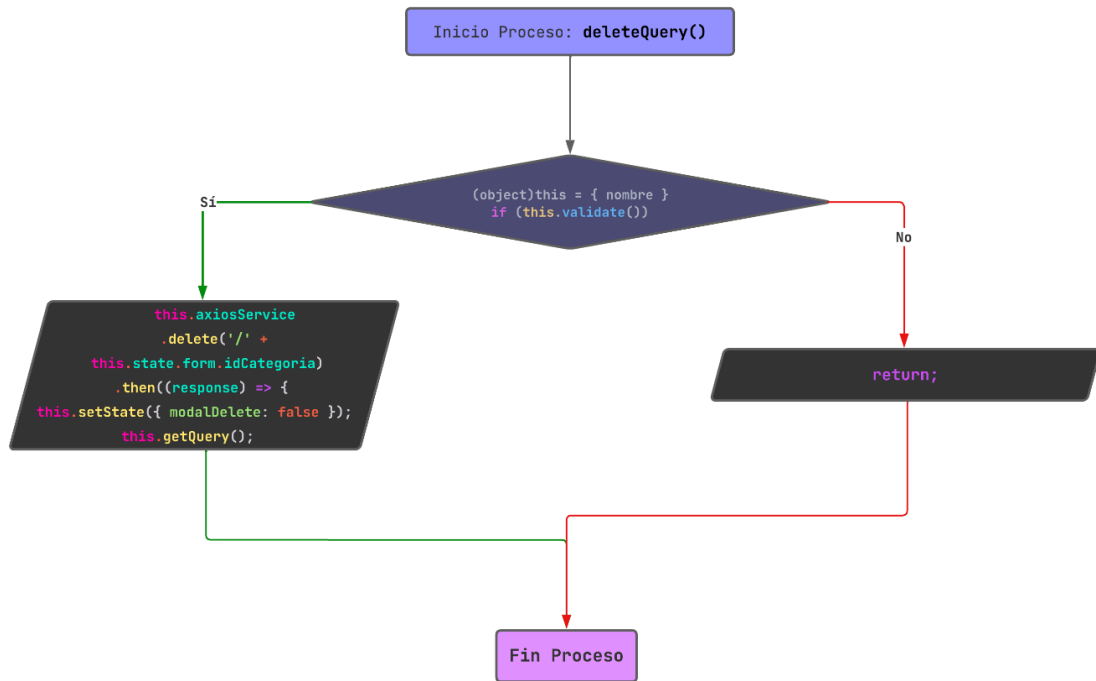
- **Código**

```

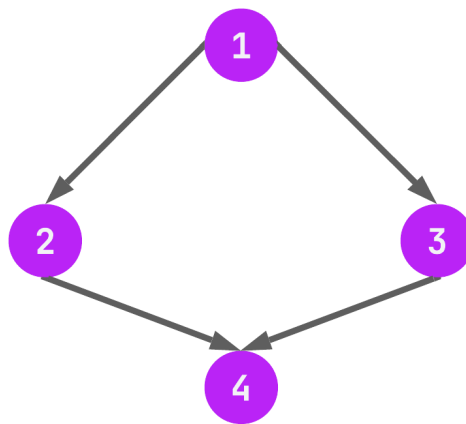
deleteQuery = () => {
  this.authService
    .delete('/') + this.state.form.idCategoria)
    .then((response) => {
      this.setState({ modalDelete: false });
      this.getQuery();
    });
};

```

- **Diagrama de flujo**



- **Diagrama de grafos**



- **Rutas**
R1: 1, 2, 4
R2: 1, 3, 4
- **Complejidad Ciclomática**
E: Número de aristas
N: Número de nodos
P: Número de nodos predicado
 $V(G) = E - N + 2$
 $V(G) = 4 - 4 + 2$
 $V(G) = 2$
 $V(G) = P + 1$
 $V(G) = 1 \text{ nodo predicado} + 1 = 2$

3.1.4.2 TÉCNICA DE CAJA NEGRA: Clases equivalentes

- Registrar Productos

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
Insertar Producto	<code>if (!this.validateLetters(form.nombre))</code>	Válido	“Samsung”
	<code>if (!this.validateLetters(form.nombre))</code>	No Válido	“Samsung 12354”
	<code>if (form.nombre.length > 50)</code>	Válido	“Nokia”
	<code>if (form.nombre.length > 50)</code>	No válido	“Nokiaasdkjsfhdjlaokdjhdhdjdjdfkaiquyetdjdjdsksflsdfksdfklsljdfksdfjsdk”
	<code>if (!form.nombre)</code>	Válido	“iPhone”
	<code>if (!form.nombre)</code>	No válido	“ ”
	<code>if (!this.validateLetters(form.marca))</code>	Válido	“Nokia”
	<code>if(!this.validateLetters(form.marca))</code>	No válido	“Nokia155asd56”
	<code>if (form.marca.length > 50)</code>	Válido	“Nokia”
	<code>if (form.marca.length > 50)</code>	No válido	“Nokiaasdkjsfhdjlaokdjhdhdjdjdfkaiquyetdjdjdsksflsdfksdfklsljdfksdfjsdk”
	<code>if (form.modelo.length > 50)</code>	Válido	“Nokia”
	<code>if (form.modelo.length > 50)</code>	No válido	“Nokiaasdkjsfhdjlaokdjhdhdjdjdfkaiquyetdjdjdsksflsdfksdfklsljdfksdfjsdk”
	<code>if (!form.modelo)</code>	Válido	“XL65”
	<code>if (!form.modelo)</code>	No válido	“ ”
	<code>if (form.cantidad < 0)</code>	Válido	657
	<code>if (form.cantidad < 0)</code>	No Válido	-45
	<code>if (this.validateLetters(form.precio))</code>	Válido	45
	<code>if (this.validateLetters(form.precio))</code>	No Válido	45dfgdf
	<code>if (form.precio < 0)</code>	Válido	-342
	<code>if (form.precio < 0)</code>	No Válido	56
	<code>if (form.precio > 1000)</code>	Válido	300
	<code>if (form.precio > 1000)</code>	No válido	30000

Caso de prueba 1: Campos completos

BLOZ CELL

• [Home](#)

• [Ingreso de Productos](#)

• [Salida de Productos](#)

Insertar Producto

ID	Nombre	Marca	Modelo
1	Samsung	Samsung	S21
2	Xiaomi	Xiaomi	Mi12
3	Huawei	Huawei	P90

ID:

4

Nombre:

Nokia

Marca:

Nokia

Modelo:

P40

Precio:

250

Características:

Ram de 6Gb y almacenamiento de 128GB

Cantidad:

24

Insertar

Cancelar

Cantidad	Acciones
	<div>Editar</div> <div>Eliminar</div>
	<div>Editar</div> <div>Eliminar</div>
	<div>Editar</div> <div>Eliminar</div>

Figura 19. Captura del aplicativo con el caso de prueba 1
Imagen Elaborado por los Autores

Caso de prueba 2: Campos incompletos

BLOZ CELL

• [Home](#)

• [Ingreso de Productos](#)

• [Salida de Productos](#)

Insertar Producto

ID	Nombre	Marca	Modelo
1	Samsung	Samsung	S21
2	Xiaomi	Xiaomi	Mi12
3	Huawei	Huawei	P90
4	Nokia	Nokia	P40

ID:

5

Nombre:

Marca:

Modelo:

Precio:

Características:

Cantidad:

Insertar

Cancelar

Cantidad	Acciones
10	<div>Editar</div> <div>Eliminar</div>
10	<div>Editar</div> <div>Eliminar</div>
10	<div>Editar</div> <div>Eliminar</div>
24	<div>Editar</div> <div>Eliminar</div>

Figura 20. Captura del aplicativo con el caso de prueba 2

Imagen Elaborado por los Autores

Todos los campos son obligatorios, por lo que, al intentar ingresar un producto con campos vacíos, el sistema muestra el error e impide que se agregue el elemento.

The screenshot shows the BLOZ CELL application interface. On the left, there's a sidebar with navigation links: Home, Ingreso de Productos, and Salida de Productos. Below the sidebar is a table of existing products with columns ID, Nombre, Marca, and Modelo. The table contains four rows of product data. To the right of the sidebar is a form to insert a new product. The form has fields for Nombre, Marca, Modelo, Precio, Características, and Cantidad. Each field has a red border and a red error message below it: "Campo requerido." (Required field). On the far right, there's a table with columns Cantidad and Acciones. The table contains four rows of product data, each with a Cantidad value and two buttons: Editar and Eliminar.

ID	Nombre	Marca	Modelo
1	Samsung	Samsung	S21
2	Xiaomi	Xiaomi	Mi12
3	Huawei	Huawei	P90
4	Nokia	Nokia	P40

Cantidad	Acciones
10	<button>Edit</button> <button>Delete</button>
10	<button>Edit</button> <button>Delete</button>
10	<button>Edit</button> <button>Delete</button>
24	<button>Edit</button> <button>Delete</button>

Figura 21. Captura del aplicativo con el caso de prueba 2

Imagen Elaborado por los Autores

• Registrar Categorías

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
Insertar categoría	<code>if (!this.validateLetters(form.nombre))</code>	Válido	“Telefonos patitos”
	<code>if (!this.validateLetters(form.nombre))</code>	No Válido	“Teléfonos patitos456”
	<code>if (!form.nombre)</code>	Válido	“Estuches”
	<code>if (!form.nombre)</code>	No válido	“ ”

Caso de prueba 1: Campo Válido



Figura 21. Captura del aplicativo con el caso de prueba 1
Imagen Elaborado por los Autores

Caso de prueba 1: Campo Invalido

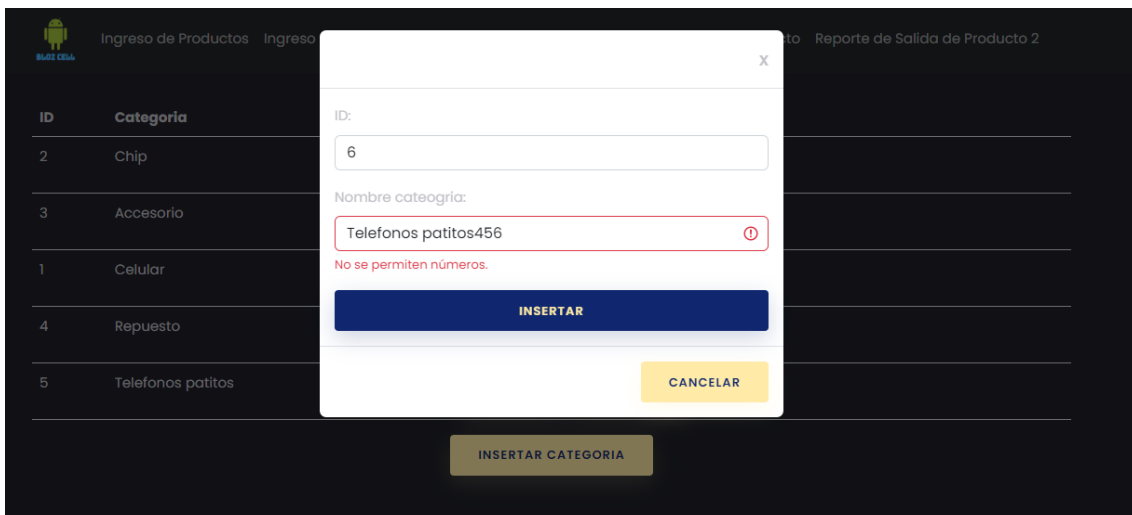


Figura 22. Captura del aplicativo con el caso de prueba 1
Imagen Elaborado por los Autores

Caso de prueba 2: Campo Completo

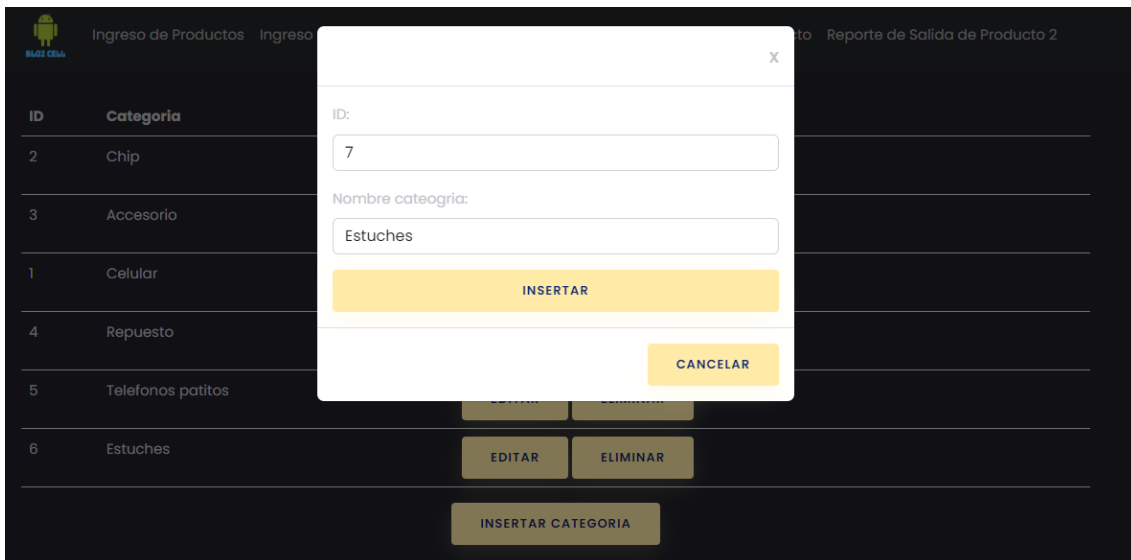


Figura 23. Captura del aplicativo con el caso de prueba 2

Imagen Elaborado por los Autores

Caso de prueba 2: Campo Incompleto

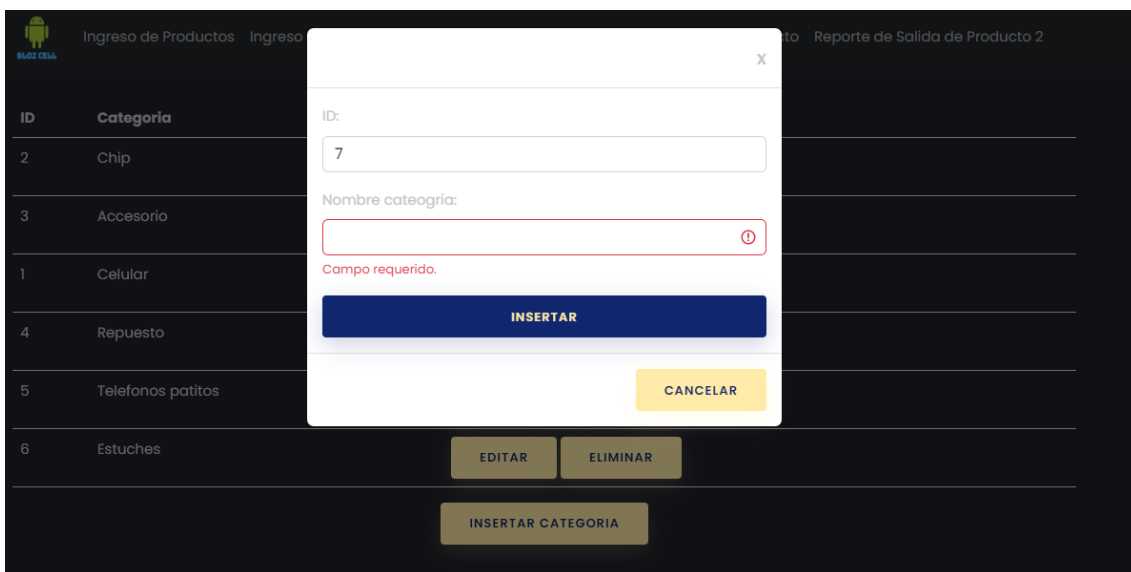


Figura 24. Captura del aplicativo con el caso de prueba 2

Imagen Elaborado por los Autores

• Login

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
Usuario	<code>if (this.state.logemail === "admin")</code>	Válido	“admin”
	<code>if (this.state.logemail === "admin")</code>	No Válido	“ADMIN”
Contraseña	<code>if (this.state.logpass</code>	Válido	“admin”

	<pre>=== "admin")</pre>		
	<pre>if (this.state.logpass === "admin")</pre>	No válido	“ 1234456”

Caso de prueba 1: Usuario y Contraseña incorrectos



Figura 25. Captura del aplicativo con el caso de prueba 1
Imagen Elaborado por los Autores

Caso de prueba 2: Usuario y Contraseña correctos

<div>  Ingreso de Productos Ingreso de Categorías Salida de Productos Reporte de Salida de Producto Reporte de Salida de Producto 2 </div>									
ID	Nombre	Marca	Modelo	Precio	Características	Cantidad	Categoría	Acciones	
1	Samsung	Samsung	S21	53.66	CaracterísticasLargas	10	Celular	EDITAR	ELIMINAR
2	Xiaomi	Xiaomi	Mi12	92.66	CaracterísticasLargas	10	Chip	EDITAR	ELIMINAR
4	Huawei	Huawei	P900	100	CaracterísticasLargas	10	Accesorio	EDITAR	ELIMINAR
7	Huawei	Huawei	P900	100	CaracterísticasLargas	10	Chip	EDITAR	ELIMINAR
8	Huawei	Huawei	P900	100	CaracterísticasLargas	10	Chip	EDITAR	ELIMINAR
<div>INSERTAR PRODUCTO</div>									

Figura 25. Captura del aplicativo con el caso de prueba 2
Imagen Elaborado por los Autores

CAPÍTULO IV

4.1 Conclusiones

Para concluir, se ha logrado desarrollar un sistema de gestión de inventarios para el emprendimiento Bloz Cell, que permite automatizar el seguimiento de los productos y la toma de decisiones relacionadas con el inventario. Este sistema cuenta con una interfaz intuitiva y fácil de usar, y almacena la información en una base de datos eficiente. Además, se ha demostrado que un análisis y diseño detallado es esencial para desarrollar software eficiente, ya que permite realizar pruebas exhaustivas y verificar que los requisitos se cumplen en relación con la documentación. En resumen, el sistema de gestión de inventarios para el emprendimiento Bloz Cell ha logrado simplificar y mejorar el seguimiento de los productos, lo que a su vez ayuda a la toma de decisiones más informadas.

4.2 Recomendaciones.

Se pudo demostrar la importancia del diseño cuidadoso y detallado en el desarrollo de un sistema de gestión de inventarios. Se encontró que la planificación previa de la base de datos es fundamental para garantizar una conexión eficiente con los registros de los productos y para evitar dificultades en el futuro. Mantener documentos claros y actualizados desde el principio también ayuda a mantener un registro claro y accesible de los progresos y decisiones tomadas durante el desarrollo, lo que puede ser de gran ayuda para resolver cualquier problema futuro. Por lo tanto, se recomienda dedicar tiempo y esfuerzo a la planificación detallada y documentación de la base de datos en cualquier proyecto de desarrollo de software.