Guia 4

Juan Sebastian Mendoza

Mikey Collazos

Juan Jose Guerrero

Sebastian Lopez

Asly Camelo

UNIVERSIDAD MANUELA BELTRAN

Tablas de contenido

Preguntas

1. Especificación de Requisitos y Diseño

¿Cómo has definido los requisitos funcionales para tu API (qué operaciones necesita hacer el sistema)?

- Crear un libro (POST).
- Leer libros (GET).
- Actualizar un libro (PUT).
- Eliminar un libro (DELETE).
- Agregar libros al carrito (POST).
- Leer el carrito (GET).
- Eliminar libros del carrito (DELETE).
- Procesar la compra (POST).
- Ver historial de pedidos (GET).
- Estas son algunas de las operación que estamos manejando en el proyecto por lo que podemos concluir que no son muchas las que necesitamos para nuestro proyecto

¿Qué requerimientos no funcionales consideraste (por ejemplo, seguridad, rendimiento, escalabilidad)?

- Seguridad: Implementación de autenticación y autorización con JWT.
- Rendimiento: Uso de paginación y optimización de consultas SQL.
- Escalabilidad: Uso de PostgreSQL para soportar grandes volúmenes de datos, con posible expansión a bases distribuidas.

Al ser no funcionales no se les debe restar ninguna importancia ya que estos son los que más pegan para el usuario final, entonces usamos los del ejemplo mismo por que los consideramos los más importantes

¿Qué casos de uso principales estás cubriendo con las rutas de tu API?

- Usuario final: Ver libros, agregar y eliminar libros del carrito, procesar compras.
- Administrador: Gestionar libros (añadir, editar, eliminar)

¿Por qué elegiste PostgreSQL como base de datos? ¿Qué ventajas ofrece frente a otras opciones (por ejemplo, SQLite)?

- MySQL es una base de datos relacional avanzada con soporte de transacciones ACID y optimización de consultas.
- Ofrece mayor capacidad de escalabilidad y concurrencia frente a SQLite, que es más adecuado para aplicaciones pequeñas.

¿Cómo estructurar las tablas de la base de datos? ¿Cuáles son las relaciones entre las entidades de los libros (por ejemplo, títulos, autores, precio)?

- Tablas principales: libros, autores, categorías, usuarios, carrito, pedidos, Relaciones: Un libro tiene un autor y una categoría; un usuario puede tener múltiples libros en su carrito y en pedidos. Un pedido puede tener múltiples libros.

¿Cómo vas a validar la entrada de datos para asegurarte de que las operaciones CRUD funcionen correctamente (por ejemplo, qué campos son obligatorios)?

- Campos obligatorios: o Creación de libros: titulo, autor_id, precio, stock, categoria_id. o Registro de usuario: email, contraseña, nombre. o Carrito: usuario_id, libro_id, cantidad. o Compra: usuario_id, libros (detalle del pedido).

- Validación: Verificación de que los campos como precio y cantidad sean numéricos y positivos; asegurar que el email no esté duplicado al registrar un usuario.

¿Cómo aseguras que las consultas SQL (como SELECT, INSERT, UPDATE, DELETE) se ejecuten correctamente con PostgreSQL?

- Uso de transacciones para asegurar que las operaciones sean atómicas y consistentes.
- Realización de consultas unitarias para verificar que funcionan correctamente.
- Uso de ORMs como Sequelize o TypeORM para simplificar la interacción con la base de datos de manera segura.

2. Implementación de la API CRUD con MySQL

¿Qué medidas tomaste para manejar posibles errores en las rutas de la API (por ejemplo, en caso de que un libro no se encuentre)?

Rta: Para manejar los errores utilizamos bloques try-catch en las rutas de la API. Por ejemplo en el caso de que un libro no está disponible, la API tira el código de error 404 not found y un mensaje indicando que no existe o no está disponible. También se manejan errores de conexión a la base de datos y errores de validación de datos enviados por el cliente.

¿Qué tipo de respuesta esperas de la API para cada tipo de solicitud (por ejemplo, POST, GET, PUT, DELETE)?

- GET: Devuelve la lista de libros o un libro específico en formato JSON.

- POST: Devuelve un mensaje de éxito y los datos del libro creado.

- PUT: Devuelve un mensaje de éxito y los datos del libro actualizado.

- DELETE: Devuelve un mensaje confirmando que el libro fue eliminado.

¿Cómo gestionar la autenticación y seguridad en tu API (si aplica)?

En esta versión básica de la API aún no se ha implementado la autenticación, ya que nos hemos enfocado en CRUD. Pero en cuanto a la seguridad, se busca usar consultas parametrizadas para evitar inyecciones SQL y se validan los datos antes de insertarlos o actualizarlos en la base de datos.

3. Integración Frontend-Backend

¿Cómo haces que el frontend (HTML, CSS, JS) se comunique con el backend (Express y PostgreSQL)?

El frontend se comunica con el backend utilizando la función fetch de JavaScript para realizar peticiones HTTP (GET, POST, PUT, DELETE) a las rutas de la API que están conectadas con la base de datos.

¿Qué métodos usas para realizar peticiones HTTP en el frontend (por ejemplo, fetch o Axios)?

Utilicé fetch porque es una función nativa de JavaScript, no necesita la instalación de librerías externas y es muy práctica para proyectos sencillos como este.

¿Cómo gestionas los errores que pueden surgir en el frontend al interactuar con la API?

Para manejar los errores en el frontend utilizó el método .catch() en las promesas de fetch. Además, muestro mensajes o alertas al usuario en caso de que ocurra algún error, por ejemplo: "Error al cargar los libros" o "No se pudo realizar la operación".

¿Cómo aseguras que el frontend sea responsivo y fácil de usar en diferentes dispositivos?

- Se asegura que la interfaz sea responsiva y fácil de usar en diferentes dispositivos por medio de Media Queries. Esta se usa para aplicar diferentes estilos dependiendo del dispositivo como el ancho de la pantalla, permitiendo que la interfaz se adapte a los

diferentes tamaños de pantallas. Además se usó herramientas como Figma para tener una mejor vista previa sobre el proyecto, esto nos permitió hacer cambios en el diseño cuando era debido para evitar errores y tener un diseño fácil de entender

Figura 1

Ejemplo de implementación en el codigo

```
@media (max-width: 768px) {

    .product-img img {
        height: 350px;
    }
}

@media (max-width: 576px) {
    .search-section .logo h1 {
        font-size: 2rem;
    }

    .hero .hero-text {
        top: 25%;
        padding-top: 15px;
    }

    .hero .hero-text h1 {
        font-size: 1.5rem;
        margin-bottom: 15px;
    }

    .product-img img {
        height: 300px;
    }

    .main-nav a {
        margin: 0 10px;
        font-size: 0.9rem;
    }

    .container {
        width: 95%;
        margin: 30px auto;
        padding: 15px;
    }
```

¿Cómo actualizar la interfaz de usuario cuando los datos cambian (por ejemplo, después de agregar, eliminar o actualizar un libro)?

- A través de pantallas dinámicas, las cuales al agregar un libro o ejercer alguna acción, modifican ciertos aspectos de la original, mostrando una retroalimentación visual del cumplimiento de la acción.

4. Pruebas

¿Qué tipo de pruebas realizaste para asegurarte de que la API funcione correctamente? (Por ejemplo, pruebas unitarias o de integración)

- En pruebas unitarias nos enfocamos en que los libros guardados puedan mostrar, sus detalles (autor, categoría libro, editorial, libro). También, en que funcione correctamente la creación de usuarios. Como prueba de integración probamos que el front-end esté bien integrado con el back-end y la base de datos. Para esto nos aseguramos que al crear una orden de pedido, ésta se cargue correctamente a la base de datos. Y por última prueba que hicimos fue de funcionalidad al validar que el LOGIN funcione correctamente.

¿Cómo probaste las funciones CRUD en Postman o Insomnia? ¿Qué resultados esperas y cómo validaste que fueron correctos?

- Para probar funciones CRUD, se usó Postman, al probar la función CREATE, utilizamos la ruta POST/api/libro. Y enviamos la solicitud en formato JSON con el anexo de los atributos del libro, recibimos una respuesta HTTP 201 created, eso significa que el método POST se ejecutó correctamente. Posterior a eso probamos el método GET para acceder a los datos que anteriormente hemos ingresado en Postman, también fue exitoso y nos envió el código de estado 200 OK. Usamos los mismos datos almacenados para

probar las funciones UPTADE y DELETE. En postman es igual a PUT y DELETE recibimos en ambas solicitudes el código 200 OK.

¿Cómo verificaste que el frontend interactúa correctamente con el backend?

- Verificamos la correcta integración con las pruebas unitarias mencionadas en el punto 1, también verificamos el correcto flujo de datos que se evidencia cuando realizamos peticiones get, post, put, delete en Postman. También al usar la interfaz gráfica de la aplicación vemos que realiza correctamente las operaciones como el login , accediendo a la base de datos para su validación efectiva.

¿Qué harías si se encuentran errores de integración entre el frontend y el backend?

- Como equipo de integración, lo primero que haríamos es verificar la configuración del front-end y back-end para encontrar errores de integración o compatibilidad, si el error no es evidente nos comunicamos con el equipo de front-end y back-end para poder encontrar la solución de una manera ágil y efectiva.

¿Cómo manejas los errores comunes que pueden ocurrir durante las pruebas (por ejemplo, registros no encontrados, datos inválidos)?

- Las manejamos validando backend y frontend, viendo cómo está funcionando el flujo de datos, y si es correcta la sintaxis que usamos como entrada para validar la función que queremos.

5. Documentación

¿Cómo documentar las rutas de la API? ¿Qué información debe tener cada ruta (por ejemplo, método HTTP, URL, cuerpo de la solicitud)?

- Para documentar las rutas de la API, se aclaran los siguientes aspectos:
    - Metodo HTTP (Get,Post, Put,Delete)
    - Endpoint (productos/:id)
    - Descripción (Especificación de la funcionalidad de la ruta)
    - Parámetros (En dado caso de necesitar)

¿Tienes un diagrama de la base de datos que muestre las tablas y sus relaciones? ¿Cómo lo hiciste?

- Si, la documentación de la base de datos incluye un diagrama físico de la misma, el cual fue elaborado con el IDE “DataGrip”, el cual elabora el diagrama en base del código original de la base de datos.

¿Qué información debe contener el README.md de tu repositorio para que otros desarrolladores puedan entender tu proyecto?

- El README.md del Repositorio debe contener una descripcion de todos los aspectos básicos e importantes del programa, al igual que una guía simple para que el usuario pueda seguir y de esa forma lograr montar nuestro programa, esto vendría a ser:

Aspectos importantes:

Mencionar información básica de la estructura de la base de datos usada.

Lenguaje utilizado para el desarrollo del backend.

Motor del servicio en el cual se va a montar la página.

Guia de como montarlo:

Mencionar que archivos descargar

Mencionar la versión tanto del lenguaje y del Motor del servicio usado

Mencionar que archivos se pueden o deben actualizarse o cambiar

Mencionar que se ha de montar la base de datos para que funcione

Mencionar como se corre o inicia la página web

¿Qué pasos y configuraciones deben seguir los usuarios para levantar el proyecto y empezar a usar la API en su entorno local?

- Para ejecutar el proyecto en su entorno local, el usuario deberá tener instalado node.js, MySQL y diferentes aplicaciones especificadas por los desarrolladores.

¿Cómo mantuviste la documentación actualizada conforme avanzabas en el desarrollo del proyecto?

- Para mantener la documentación actualizada conforme el avance o cambios del proyecto, se elaboraron distintas versiones del documento original.

  Manejamos un archivo final llamado “Informe_ingenieria_web” en el cual se mantienen todos los campos actualizados correctamente.

  Cuando se avanza en una guía, se elabora un documento aparte (como este por ejemplo) en el cual se experimenta y se colocan los avances para futuras posibles correcciones.

Luego de las correcciones , se implementan los datos de la guía en el archivo final, se sube al repositorio, reemplazando el archivo anterior.

6. Control de Versiones con Git

¿Cómo gestionar las ramas en Git para el desarrollo de características (por ejemplo, una rama para la API, otra para el frontend)?

- Las ramas las gestionamos en la forma en la que los diferentes grupos de desarrollo, en el caso del ejemplo Backend implementando las API y frontend actualizando las interfaces, tienen diferentes carpetas donde suben los archivos de los respectivos grupos para asi mantener un orden y reconocer el progreso de cada grupo.

¿Qué tipo de mensajes de commit usaste para describir los cambios en el código? ¿Por qué son importantes estos mensajes?

- Los mensajes utilizados están basados en los formatos estandarizados para escribir mensajes de git, como por ejemplo:
    - Feat: nos expresaría que es necesario implementar una nueva funcionalidad
    - Fix: Utilizado para aclarar algún error a corregir
    - Docs: Cambios en la documentación del proyecto(por ejemplo si se cambio tecnologias, base de datos, ETC)
    - Test: para añadir o modificar las pruebas unitarias
    - Style: cambiar el apartado visual del sector sin modificar el lógico
- Estos mensajes son importantes ya que establecen una guía de desarrollo y comunicación indirecta entre desarrolladores, facilitando la fusión de ramas, y simplifica el futuro despliegue

¿Cómo resolviste conflictos al trabajar en equipo con Git? ¿Qué estrategia utilizaste para evitar problemas de integración en el

Código?

- El dividir procesos a medida de tiempo, mantener cada grupo de desarrolladores (Front,Back e integración) en una campo del git, facilita la integración de código final

¿Cómo aseguraste que el historial de commits sea claro y esté bien organizado para el seguimiento del proyecto?

- Estableciendo la creación de funcionalidades en un espacio de tiempo preestablecido, posteriormente las pruebas unitarias, finalizando con la integración, pruebas de integración, y despliegue, generando un historial de comit ordenado cronológica y funcionalmente.

7. Despliegue y Presentación Final

¿Cómo planeas desplegar el proyecto para que otros usuarios puedan acceder a él? ¿Vas a usar plataformas como GitHub Pages o Heroku?

- El despliegue de la aplicación se planea hacer en un servidor propio, el cual es de el compañero, juan guerrero, Se busca montar todo el sistema en dicho host para poder acceder a él sin necesidad de mantener instalando node u otros archivos en varios dispositivos

¿Qué pasos seguiste para preparar tu aplicación para el despliegue (por ejemplo, asegurarse de que los datos sensibles estén protegidos)?

- Se sustituye las claves API y credenciales del repositorio, con las variables de entorno, configurando el archivo .env y excluyendo del .gitignore, finalmente comprobamos que los endpoint estén funcionando correctamente

¿Qué pruebas finales harás antes de entregar el proyecto? ¿Cómo garantizarán que el sistema funcione correctamente después de la integración?

- Usando aplicaciones para el testeo de las APIs, e interactuando con la misma para ejercer las pruebas de test en cada módulo, evidenciando las fallas existentes para ser corregidas antes de la presentación final.

¿Cómo documentar el proceso de despliegue para que otros puedan replicarlo fácilmente?

- A través de un archivo README.md en el git, se incluyen las instrucciones paso a paso, tales como: requisitos previos, comandos para iniciar el proyecto, detalles sobre las variables de entorno.

A la par se incluyen capturas presentando los enlaces para facilitar la comprensión de futuros desarrolladores en el proyecto

Base de Datos

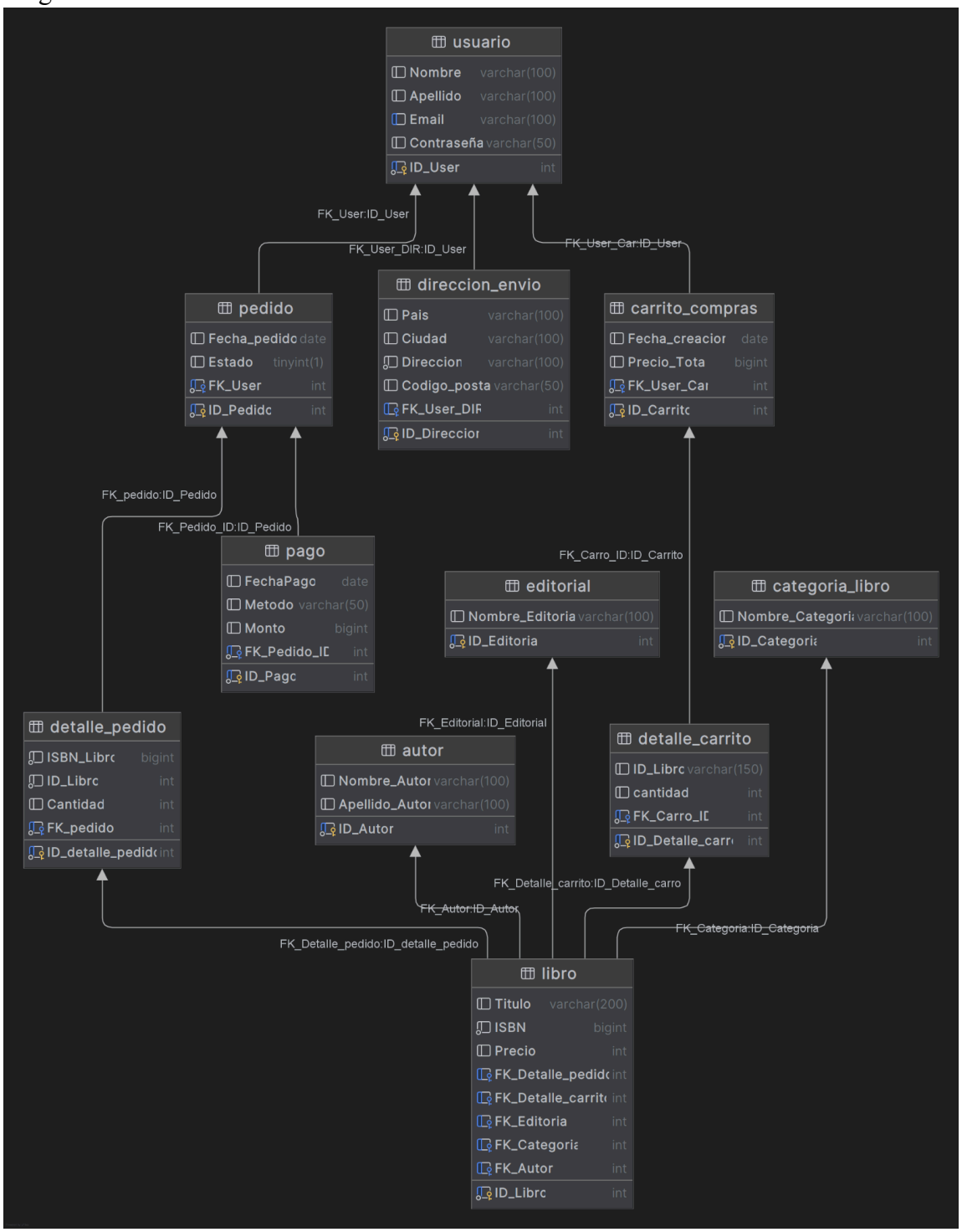
- La base de datos se llama Librería BD, será desarrollada en MySQL. Su objetivo es modelar el flujo de compra de libros en línea. Gestionando usuarios, pedidos, detalles de libros ,ETC, para l cual se implementarán las siguientes entidades:

- Tabla usuario

  Restricciones: El campo Email es único.

- Tabla pedido
- Tabla pedido
- Tabla pago
- Tabla Detalle_Pedido
- Tabla Direccion_Envio
- Tabla Carrito_Compras
- Tabla Detalle_Carrito
- Tabla Editorial
- Tabla Categoria_Libro
- Tabla Autor
- Tabla Libro

Figura 2
Diagrama Base de Datos



Requerimientos y casos de uso

Tabla 1

Requerimiento funcional n°1 [creacion de usuario]

Tipo:	**Prioridad:**
Usuarios	Alta

Descripción:

El sistema debe permitir al usuario crear una nueva cuenta para el uso dentro de la página y debe almacenar el usuario en la base de datos

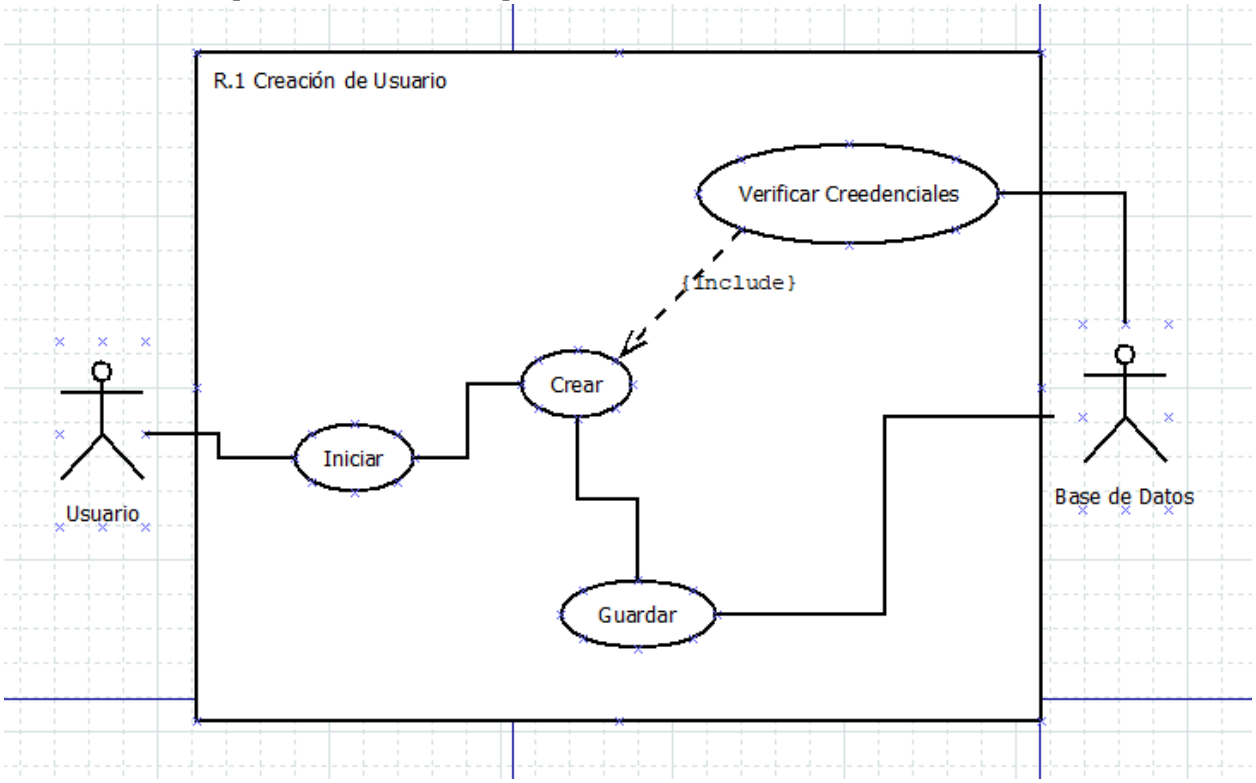
Criterios de aceptación

- Los campos no pueden enviarse vacíos
- El correo debe tener un dominio
- La contraseña debe aceptar todos los caracteres
- El sistema debe redirigir al usuario a la página de login después de crear la cuenta
- El usuario debe aparecer en la base de datos de la página
- Los campos no pueden estar vacíos

Nota: Esta tabla menciona la descripción y criterios de aceptación para manejar el registro de nuevos usuarios en el sistema

Figura 3

Caso de Uso n°1 [creación de usuario]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario con la base de datos al momento de crear y guardar las credenciales creadas en el software

Tabla 2
Requerimiento funcional n°2 [Inicio sesión]

Tipo:	**Prioridad:**
Usuarios	Alta

Descripción:

El sistema debe permitir el ingreso al sistema luego de haber validado las credenciales digitalizadas por el usuario

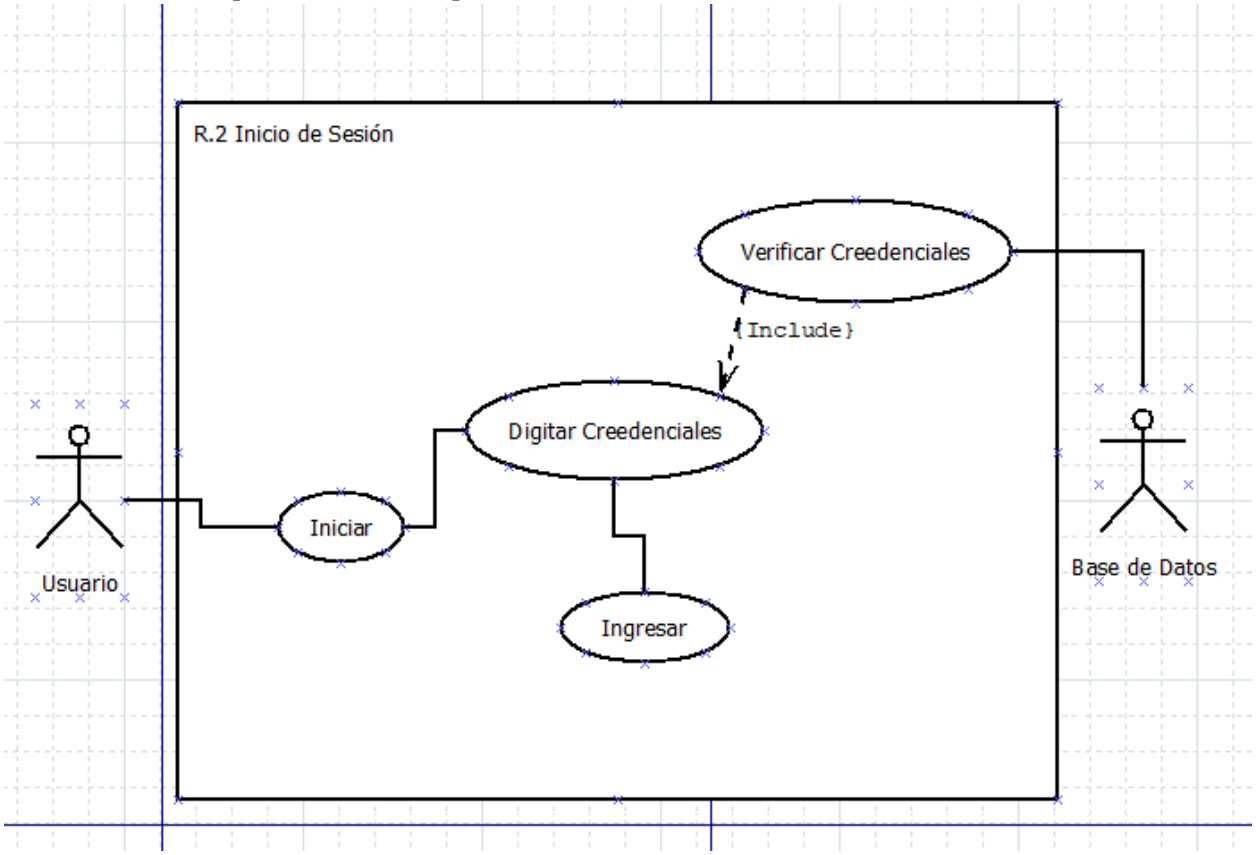
Criterios de aceptación

- Las credenciales deben ser validadas en la base de datos
- El sistema muestra un error si las credenciales no coinciden en el sistema
- Cuando las credenciales coincida el sistema debe redirigir al usuario al menú principal
- Las cuenta debe bloquearse luego de 4 intentos fallidos de login

Nota: La tabla referencia los criterios que se debe tener en el proceso de inicio de sesión dentro del sistema

Figura 4

Caso de Uso n°2 [Inicio de Sesión]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario con la base de datos al momento de iniciar sesión en la página web

Tabla 3
Requerimiento funcional n° 3 [Editar perfil]

Tipo:	**Prioridad:**
Usuario	Alta

Descripción:

El sistema permite al usuario ingresar a las configuraciones de su perfil, permitiendo modificar diferentes variables correspondientes al usuario (nombre, correo, contraseña) y variables dependientes del proceso de compra.

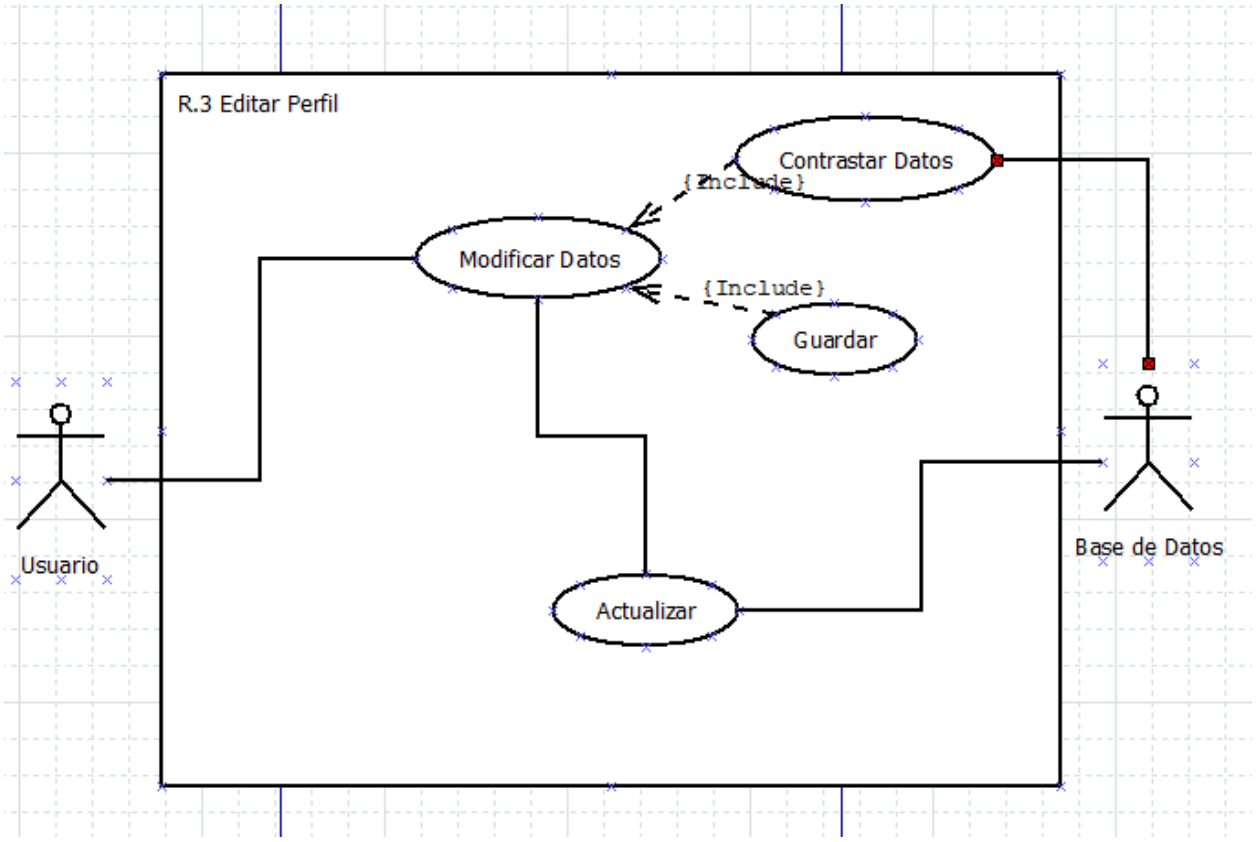
Criterios de aceptación

- El nuevo correo no debe estar vinculado con ninguna otra cuenta del sistema
- Los nuevos datos deben cumplir el mismo formato que los ingresados previamente
- Los datos deben ser actualizados en la base de datos del sistema
- Mostrar un mensaje de confirmación de actualización de datos.

Nota: Esta tabla Especifica los criterios para actualizar datos dentro de el sistema

Figura 6

Caso de Uso n°3 [Editar Perfil]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario con la base de datos al momento de editar el perfil creado por el usuario, se deben contrastar datos para que no se repitan y guardar para actualizar en la Base de Datos

Tabla 4
Requerimiento funcional n°4 [Barra de búsqueda]

Tipo:	**Prioridad:**
Busqueda	Alta

Descripción:

El sistema debe permitir consultas de libros basado en el nombre del mismo, mostrando la descripción

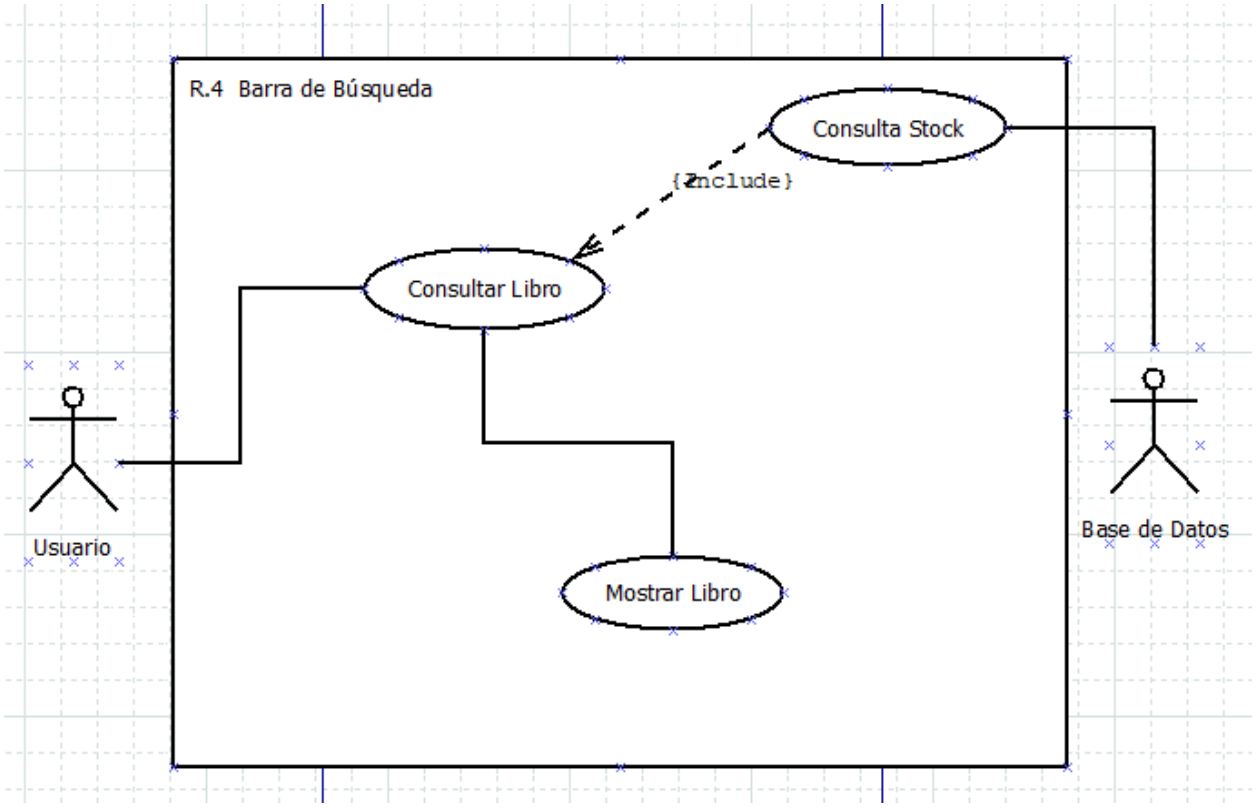
Criterios de aceptación

- El campo de búsqueda no puede enviarse vacío
- Realizar la consulta en la base de datos
- Mostrar detalles del libro, o libros similares
- En dado de no encontrarlo mostrar un mensaje

Nota: La tabla expresa los criterios de búsqueda en el sistema

Figura 7

Caso de Uso n°4 [Barra de Búsqueda]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario con la base de datos al momento de buscar un libro por medio de la barra de búsqueda, donde consulta stock y muestra el libro

Tabla 5
Requerimiento funcional n° 5[Lista de libros (Admin)]

Tipo:	**Prioridad:**
Necesario	Media

Descripción:

El administrador podrá revisar las listas de libros en inventario existente, en ese caso podrá consultarlos, modificarlos o eliminarlos en caso de ser necesarios, todo visualizado en un lista de libros.

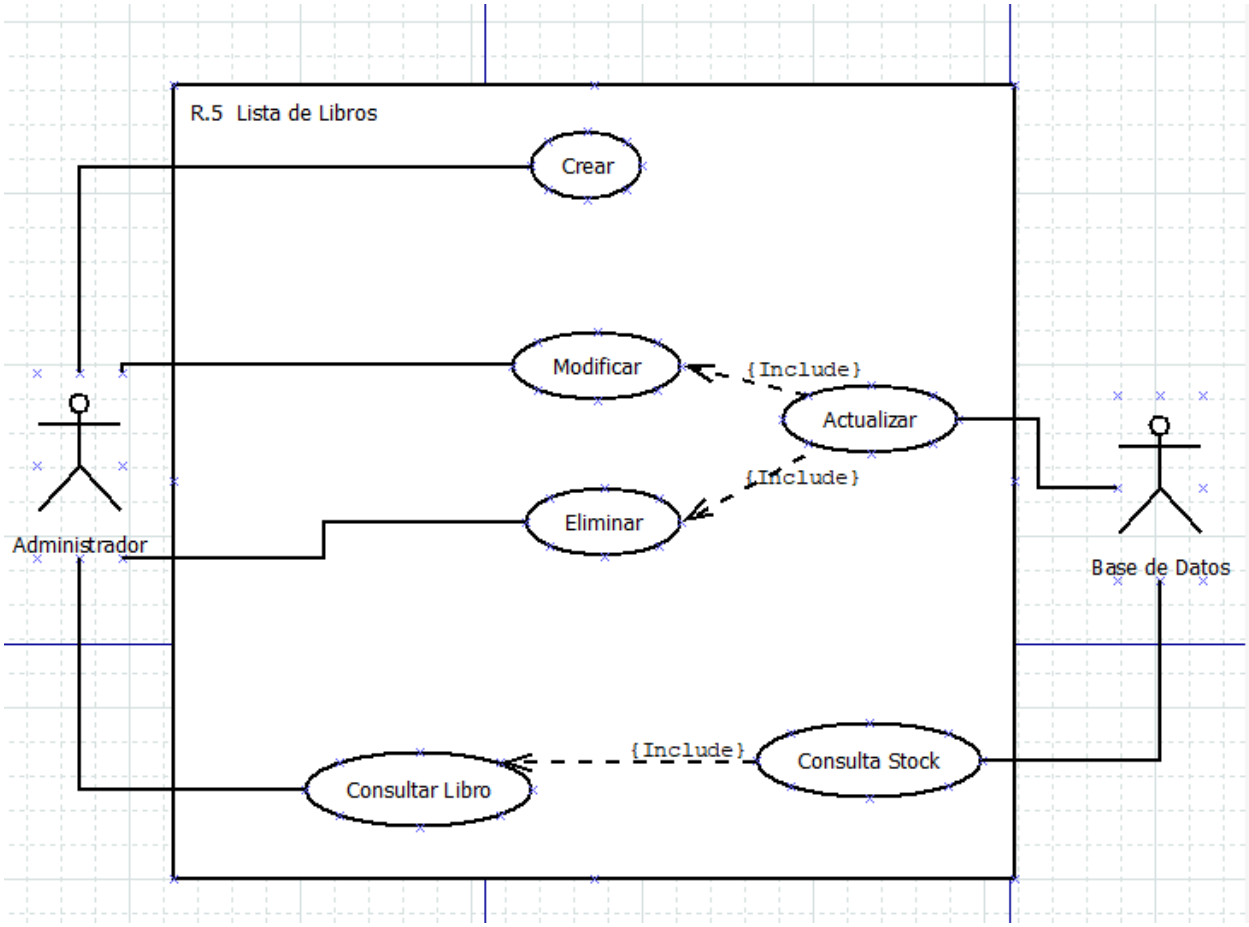
Criterios de aceptación

- Confirmar la existencia de los libros en la base de datos
- Al actualizar información se deberá actualizar en la base de datos.

Nota: La tabla expresa las acciones que puede tener un administrador en el módulo de libros

Figura 8

Caso de Uso n°5 [Lista de Libros]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del administrador con la base de datos al momento de crear, modificar, eliminar o consultar un libro mediante lista de libros y comprobar en la base de datos

Tabla 6
Requerimiento funcional n° 6 [Filtros]

Tipo:	**Prioridad:**
Necesario	Media

Descripción:

El sistema permitirá realizar consultas en la barra de búsqueda por medio de diferentes filtros generados, en ese caso se buscarán por medio de Autor, Año, Género entre otros…

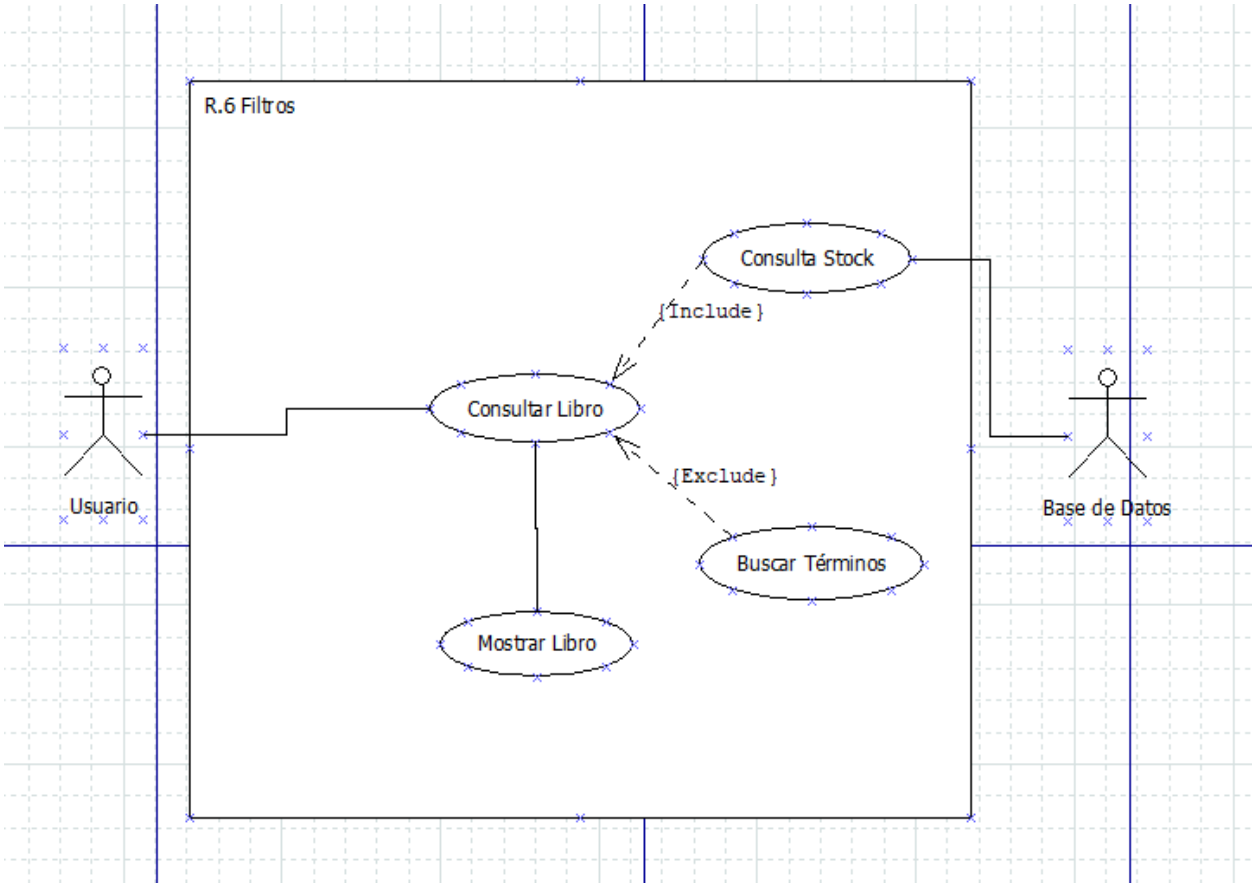
Criterios de aceptación

- El sistema deberá generar grupos por medio de los filtros escogidos.
- El sistema contrastará en la base de datos las consultas por medio de filtros.

Nota: La tabla explica los filtros que se usarán para la barra de búsqueda en los procesos de búsqueda de libros

Figura 9

Caso de Uso n°6 [Filtros]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario y la base de datos al momento de consultar un libro identificando que es posible buscar el libro por medio de Filtros y mostrar el libro

Tabla 7
Requerimiento funcional n° 7 [Crear Filtros]

Tipo:	**Prioridad:**
Necesario	Media

Descripción:

El administrador podrá generar filtros en la búsqueda de libros para el usuario, permitiendo tener mejor acceso a las búsquedas y consultas para el usuario.

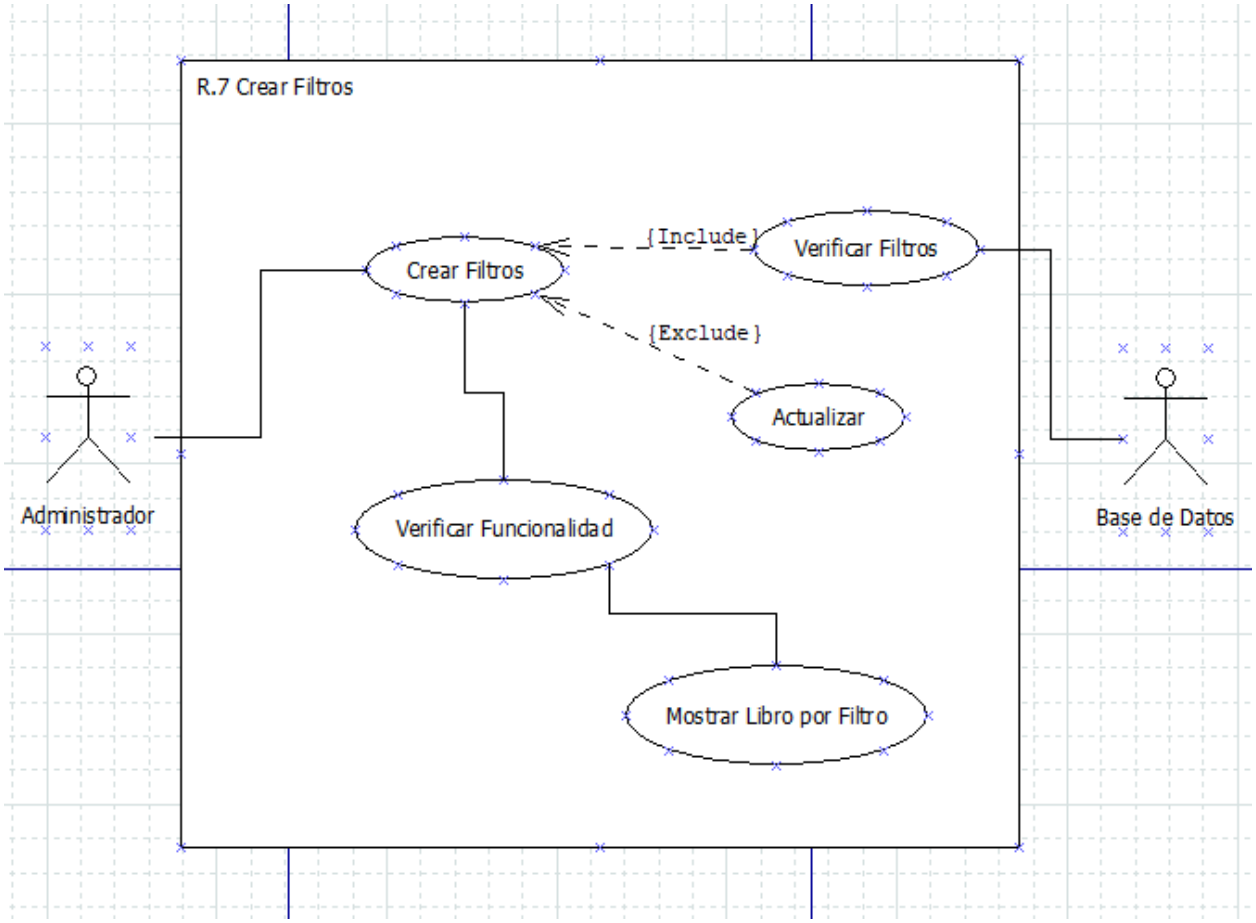
Criterios de aceptación

- El sistema deberá revisar si no está creado el filtro en la base de datos antes de agregarlo.
- El sistema avisará si es posible o no crear el filtro de búsqueda para el usuario, esto con mensajes por pantalla.

Nota: En esta tabla se expresa la posibilidad de que el Administrador pueda crear filtros en beneficio del usuario y la búsqueda de libros

Figura 10

Caso de Uso n°7 [Crear Filtros]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del administrador y la base de datos el crear filtros para la barra de búsqueda, como la base de datos verifica los filtros y se actualiza, se realizan las pruebas para que muestre los libros

Taba 8
Requerimiento funcional n° 8 [Recomendaciones]

Tipo:	**Prioridad:**
Necesario	Media

Descripción:

El sistema identificará los libros más vendidos, los cuales mostrará de primeras a los usuarios en forma de recomendaciones de compra.

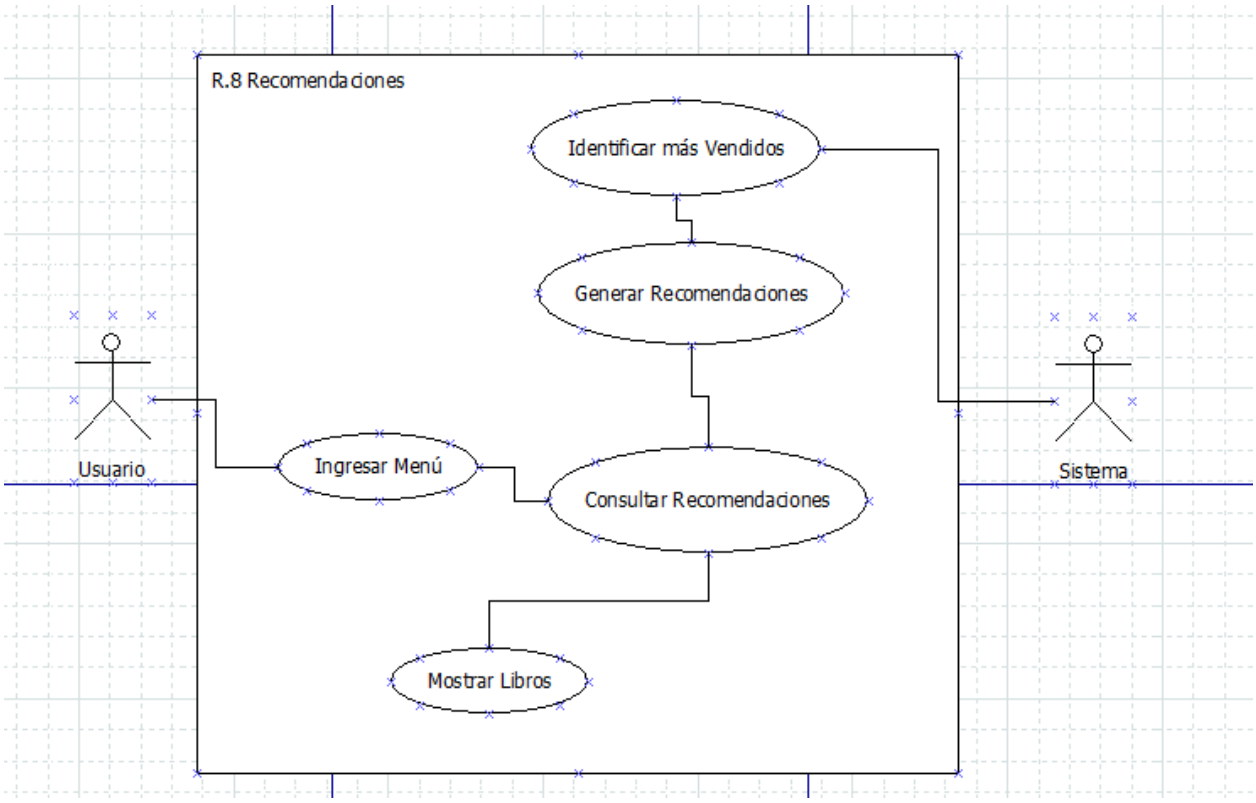
Criterios de aceptación

- El sistema deberá revisar los libros más comprados para poder dejarlos en recomendaciones para todos los usuarios.

Nota: En esta tabla se explica cómo el sistema identificará los libros más vendidos para dar recomendaciones a los usuarios para comprar

Figura 11

Caso de Uso n°8 [Recomendaciones]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario y el sistema, de parte del sistema generar las recomendaciones con libros más vendidos y del usuario consultar las recomendaciones dadas

Tabla 9
Requerimiento funcional n° 9 [Información del libro]

Tipo:	**Prioridad:**
Necesario	Alta

Descripción:

El sistema tendrá los libros guardados, en ese caso, aloja la información de cada uno para mostrar su sinopsis y diferentes detalles del libro para poder mostrar al usuario la descripción del libro.

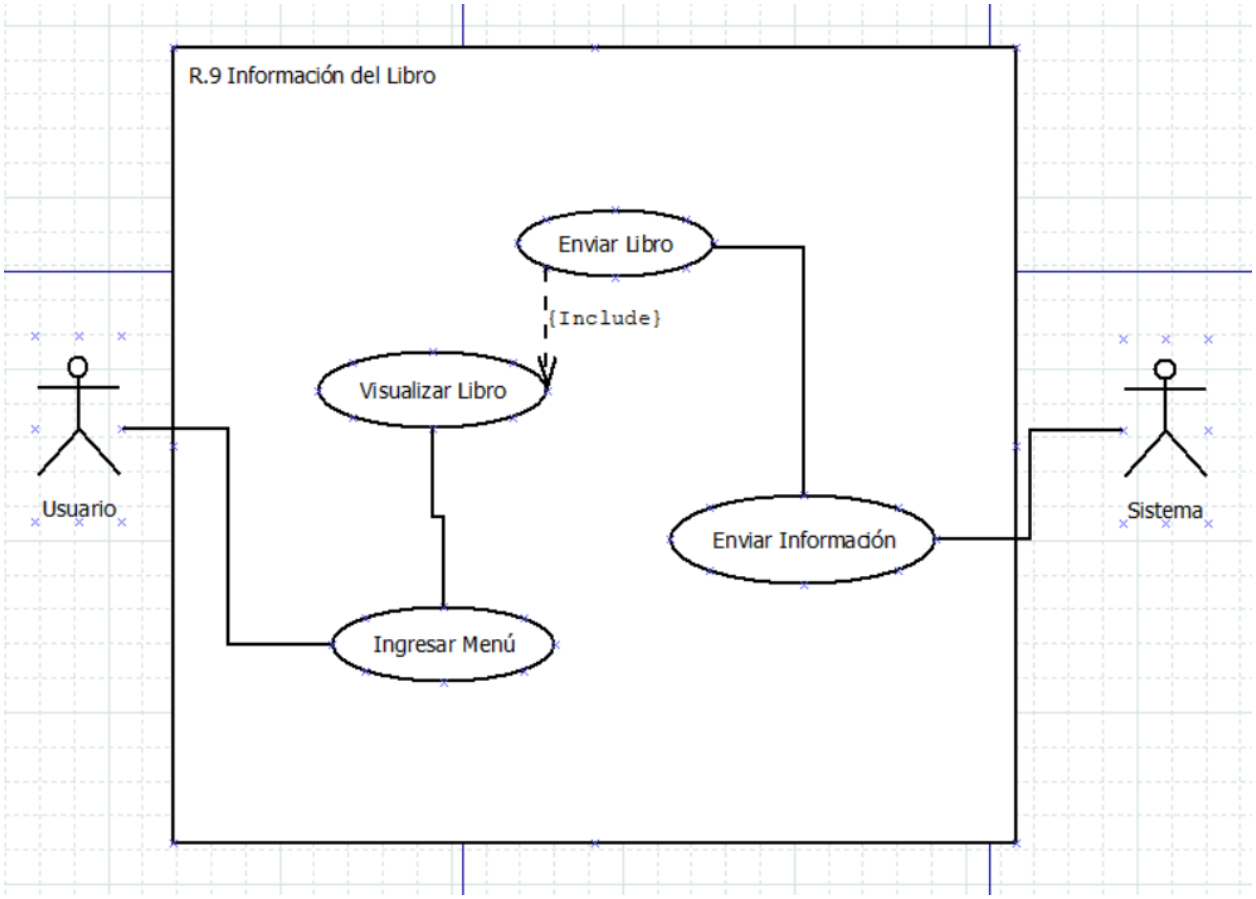
Criterios de aceptación

- Se deben realizar pruebas en un entorno de preproducción antes de desplegar actualizaciones en producción

Nota: La tabla identifica la información del libro guardadas para mostrarlas al usuario

Figura 12

Caso de Uso n°9 [Información del Libro]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario y el sistema, de parte del sistema enviar la información del libro y enviar el libro que se desea visualizar y por parte del Usuario la consulta del libro

Tabla 10
Requerimiento funcional n°10 [Gestión de libros (Usuario)]

Tipo:	**Prioridad:**
Necesario	Alta

Descripción:

El administrador podrá gestionar el catálogo de libros visibles en la biblioteca, permitiendo modificar el catálogo que podrá ser mostrado para el usuario y disponible para su compra

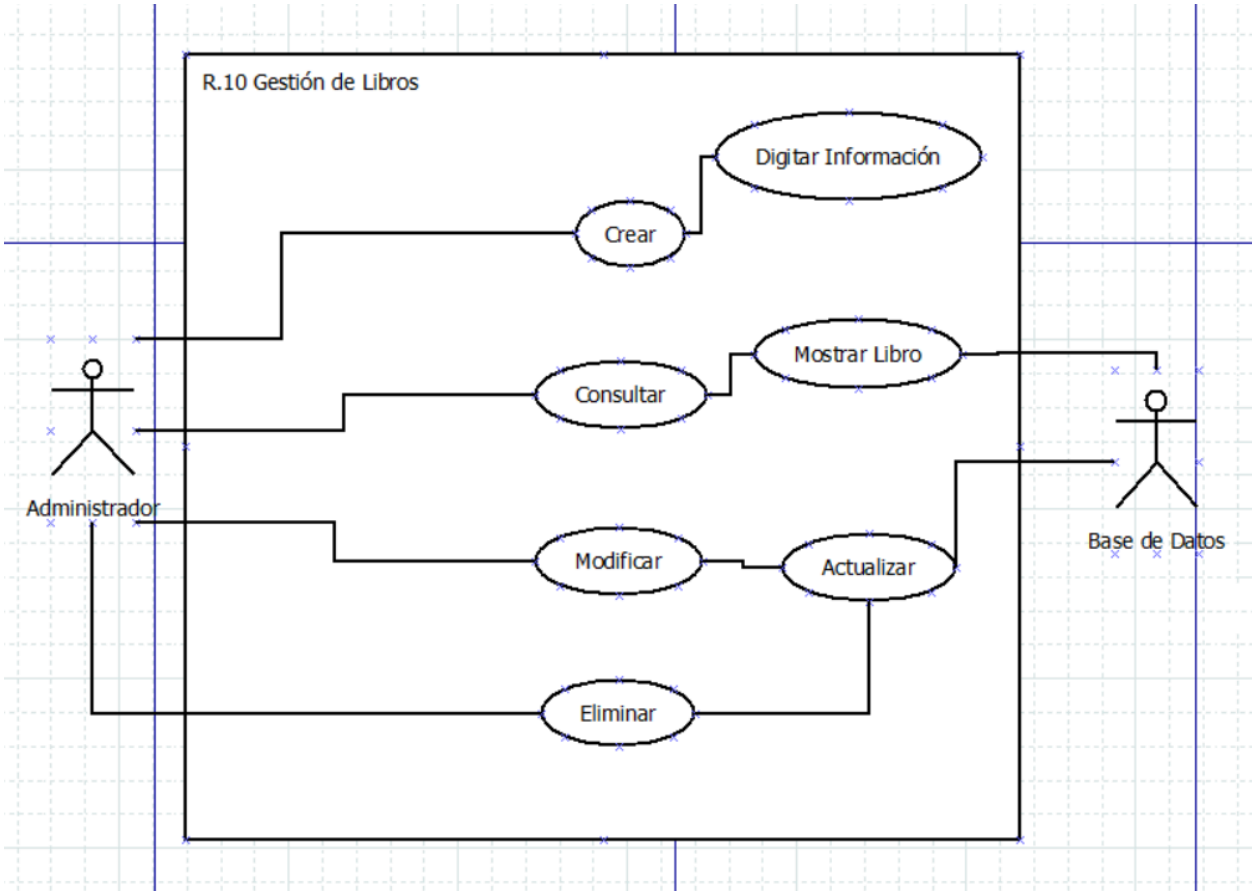
Criterios de aceptación

- El sistema debe permitir agregar, editar y eliminar libros en el catálogo visible para el usuario
- La modificación en el catálogo se debe actualizar en la base de datos
- El catálogo actualizado será mostrado a los usuarios al ingresar a la página

Nota: La tabla nos menciona que el usuario deberá de tener la capacidad de poder visualizar únicamente los libros deseados por el administrador de la página, limitando la capacidad de compra pero no siendo el equivalente a borrar o editar los libros de la base de datos

Figura 13

Caso de Uso n°10 [Gestión de libros]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del administrador y la Base de Datos para la Gestión de Libros de un catálogo, en este caso al momento de realizar las operaciones se contrastará con las Base de Datos o se guardará

Tabla 11
Requerimiento funcional n° 11 [Información del libro (Usuario)]

Tipo:	**Prioridad:**
Necesario	Alta

Descripción:

Muestra los detalles de un libro, tales como :

- Título
- Autor
- Género
- Número de ejemplares

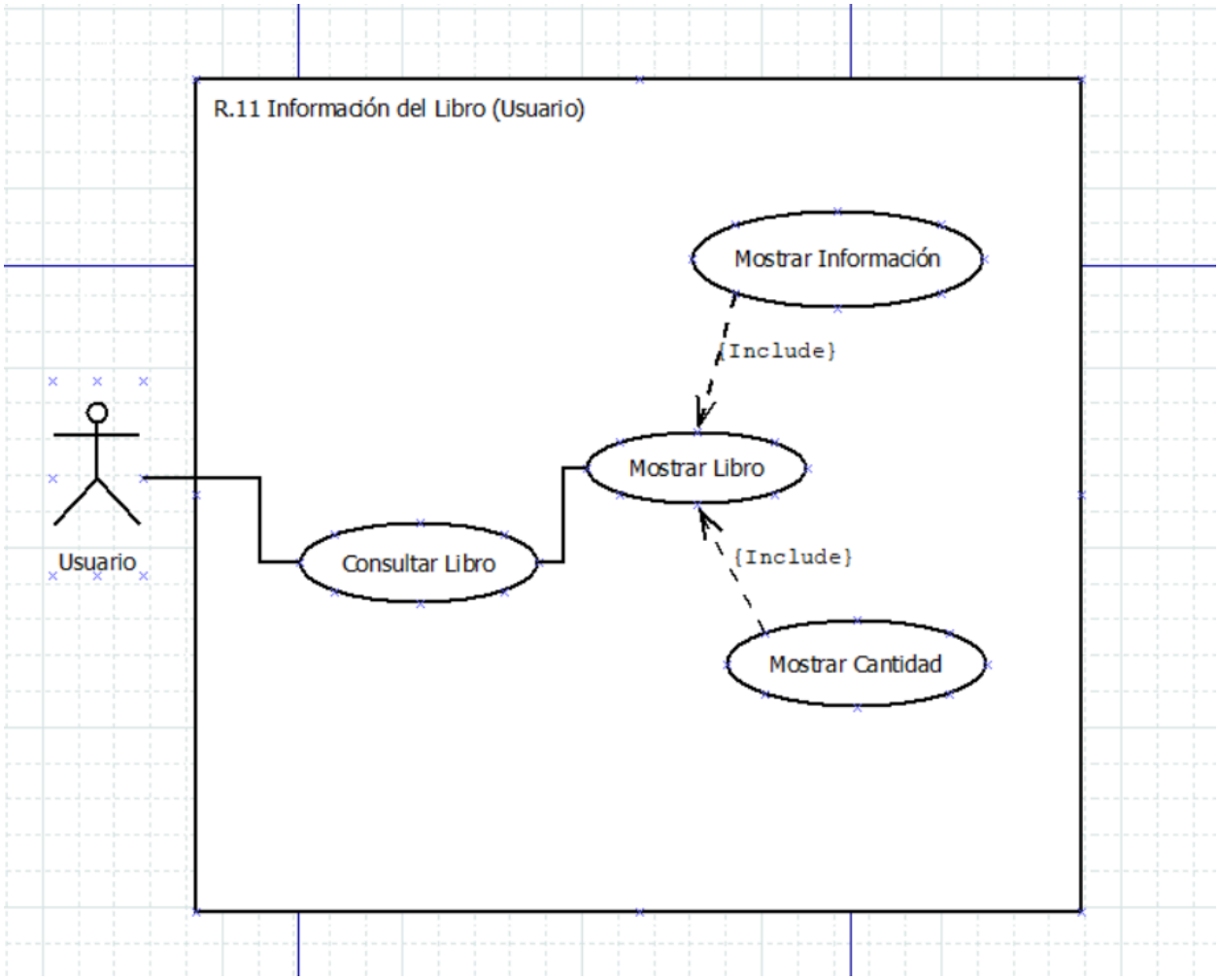
Criterios de aceptación

- La página debe mostrar los detalles de cualquier libro registrado en la biblioteca

Nota: La tabla nos menciona que cuando el usuario entre a buscar un libro a este se le ha de brindar toda la información relacionada a este incluyendo: "Título, Autor, Género, Número de ejemplares, etc…"

Figura 14

Caso de Uso n°11 [Información del Libro (Usuario)]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario con el sistema al momento de consultar un libro y que esté le muestre la información y la cantidad de Stock del libro

Tabla 12
Requerimiento funcional n° 12 [Editar detalles del libro (Administrador)]

Tipo:	**Prioridad:**
Necesario	Media

Descripción:

Permite al administrador modificar la información de un libro en el catálogo, más no eliminarlo

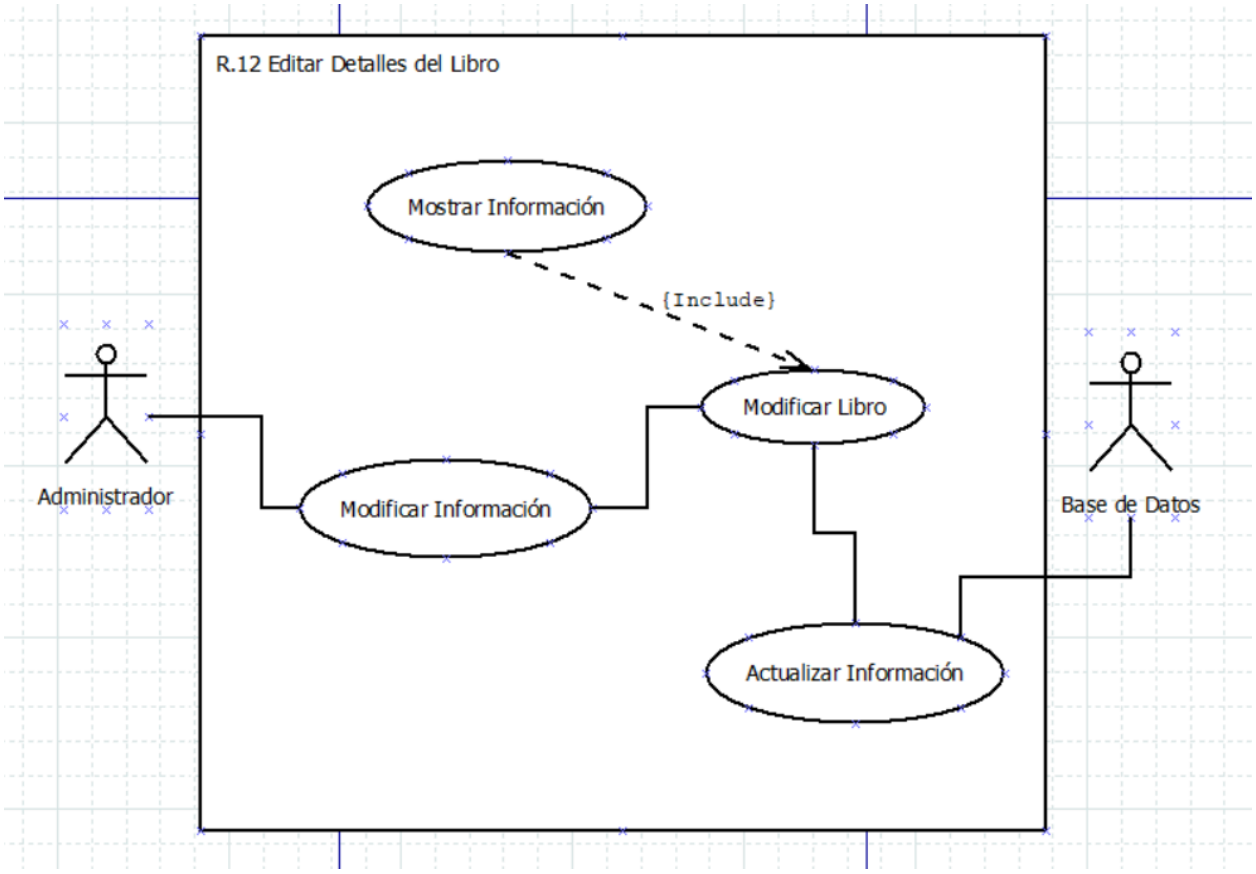
Criterios de aceptación

- Solo los administradores pueden editar los detalles de lo libros
- Los detalles del libro deben ser actualizados en la base de datos

Nota: La tabla nos menciona una de las capacidades del administrador, siendo esta la capacidad de poder editar la información de los libros dentro de la base de datos, esto incluye: "la descripción del libro, su título, autor, categorías, género, imagen de portada o a mostrar de libro"

Figura 15

Caso de Uso n°12 [Editar Detalles del Libro]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del administrador con el sistema y la Base de Datos, en este caso el administrador modifica la información de un libro por lo que se actualiza en la base de datos

Tabla 13
Requerimiento funcional n° 13 [Agregar libros (Administrador)]

Tipo:	**Prioridad:**
Mantenimiento	Media

Descripción:

El administrador ha de poder agregar libros a la base de datos, esto incluye, la descripción del libro, su título, autor, categorías, género, imagen de portada o a mostrar de libro

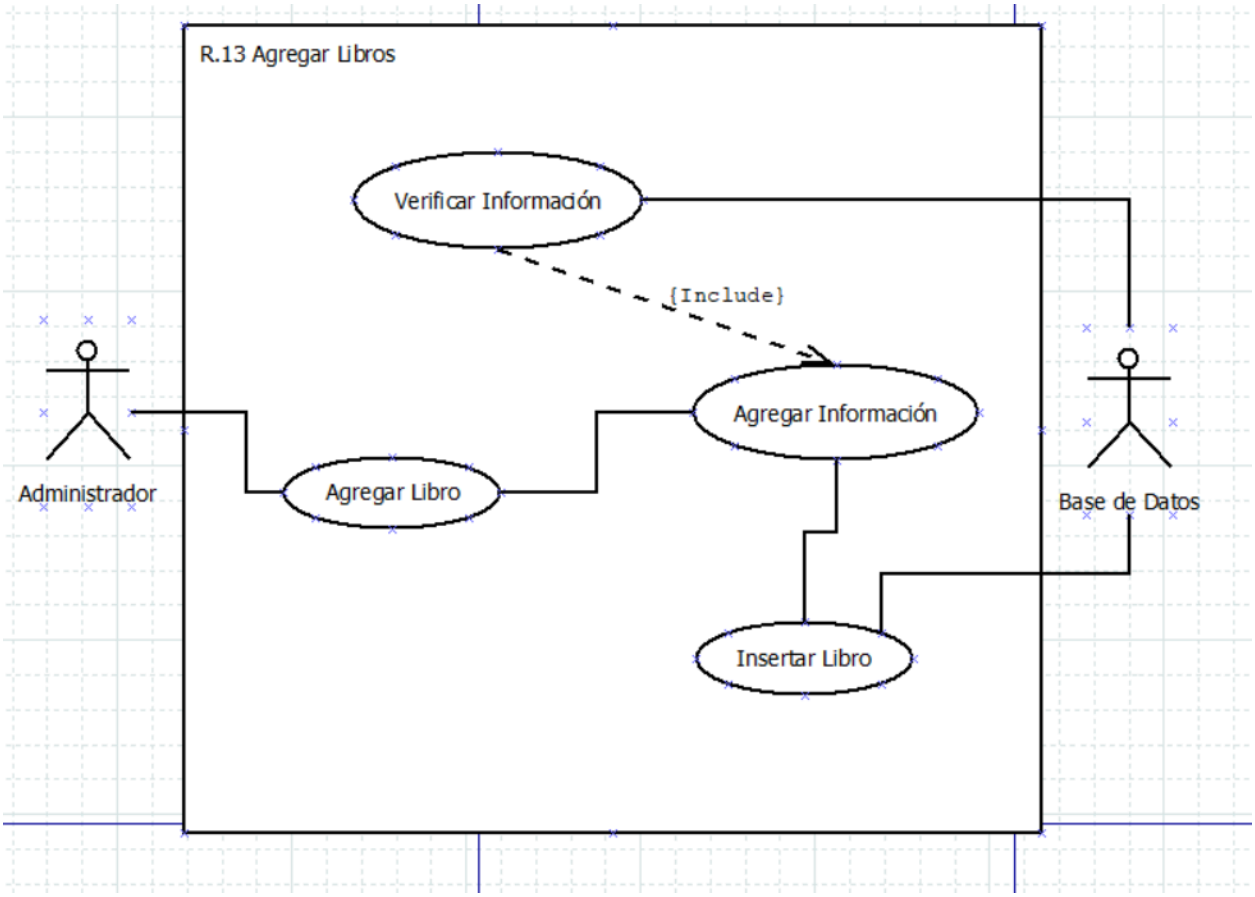
Criterios de aceptación

- El sistema ha de actualizar y subir la base de datos la información de libro a crear por el administrador

Nota: La tabla nos menciona una de las capacidades del administrador de la página, siendo esta, la capacidad de poder agregar libros a la pagina, siendo necesariamente agregar "la descripción del libro, su título, autor, categorías, género, imagen de portada o a mostrar de libro"

Figura 16

Caso de Uso n°13 [Agregar Libros]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del administrador con el sistema y la Base de Datos, en este caso el administrador agrega un libro añadiendo información e insertando el libro en la base de datos, todo es verificado por la base de datos para que sea correcto

Tabla 14
Requerimiento funcional n° 14 [Eliminar libros (Administrador)]

Tipo:	**Prioridad:**
Necesario	Alta

Descripción:

El administrador ha de poder acceder a la base de datos de la librería y eliminar algún libro de ésta, en caso de que ya no desee venderlo o necesitarlo en la librería

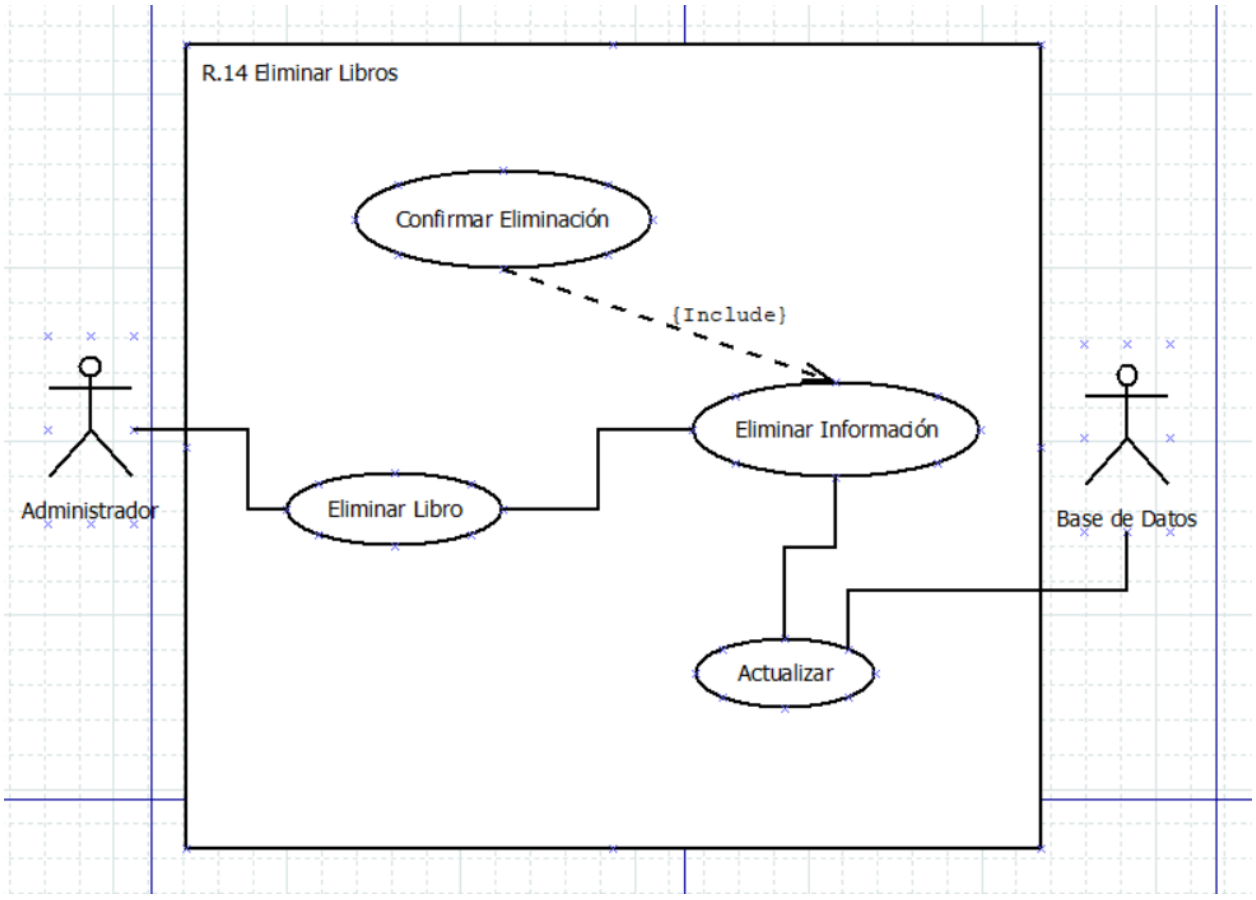
Criterios de aceptación

- El sistema ha de borrar totalmente la información del libro deseado borrar por el administrador

Nota: La tabla nos menciona una de las capacidades del administrador de la página, siendo que esta, la capacidad de poder eliminar libros totalmente de la base de datos

Figura 17

Caso de Uso n°14 [Eliminar Libros]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del administrador con el sistema y la Base de Datos, en este caso el administrador elimina un libro o la información de un libro, para ello se pide que confirme y en ese caso se actualiza en la Base de Datos

Tabla 15
Requerimiento funcional n° 15 [Cantidad a comprar]

Tipo:	**Prioridad:**
Necesario	Media

Descripción:

El usuario al momento de comprar un libro, ha de poder seleccionar la cantidad a comprar antes de agregarla al carrito de compras

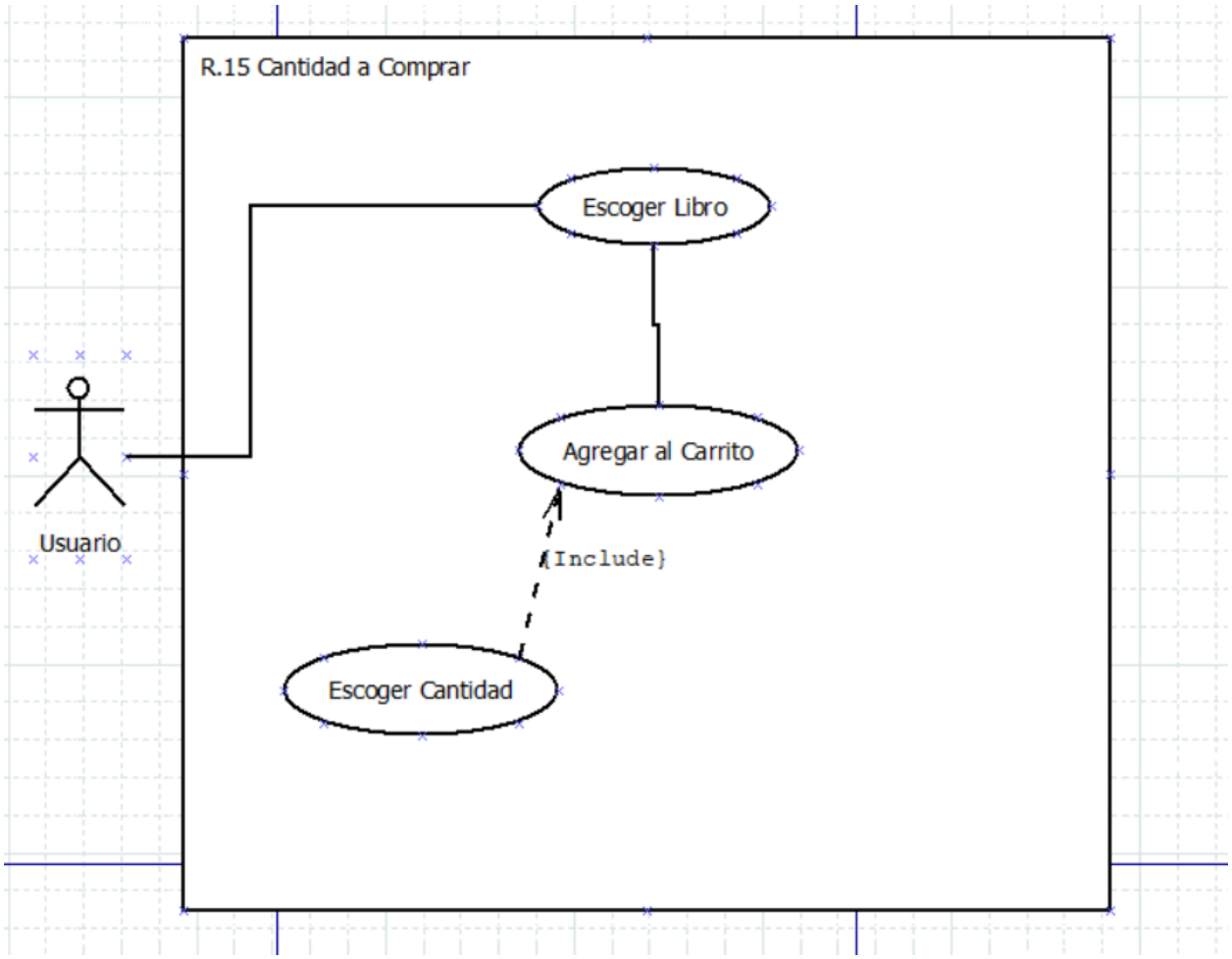
Criterios de aceptación

- El sistema ha de actualizar el carrito con la cantidad indicada del usuario

Nota: La tabla nos menciona que los usuarios han de poder seleccionar la cantidad de libros a comprar antes de agregarlos al carrito

Figura 18

Caso de Uso n°15 [Cantidad a Comprar]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario al momento de escoger un libro y agregarlo al carrito para comprar, en ese caso debe escoger la cantidad antes de comprarlo

Taba 16
Requerimiento funcional n° 16 [Carrito de compras]

Tipo:	**Prioridad:**
Necesario	Alta

Descripción:

El usuario ha de poder visualizar el contenido dentro de su carrito de compras, al igual que la posibilidad de editar el contenido de esta misma

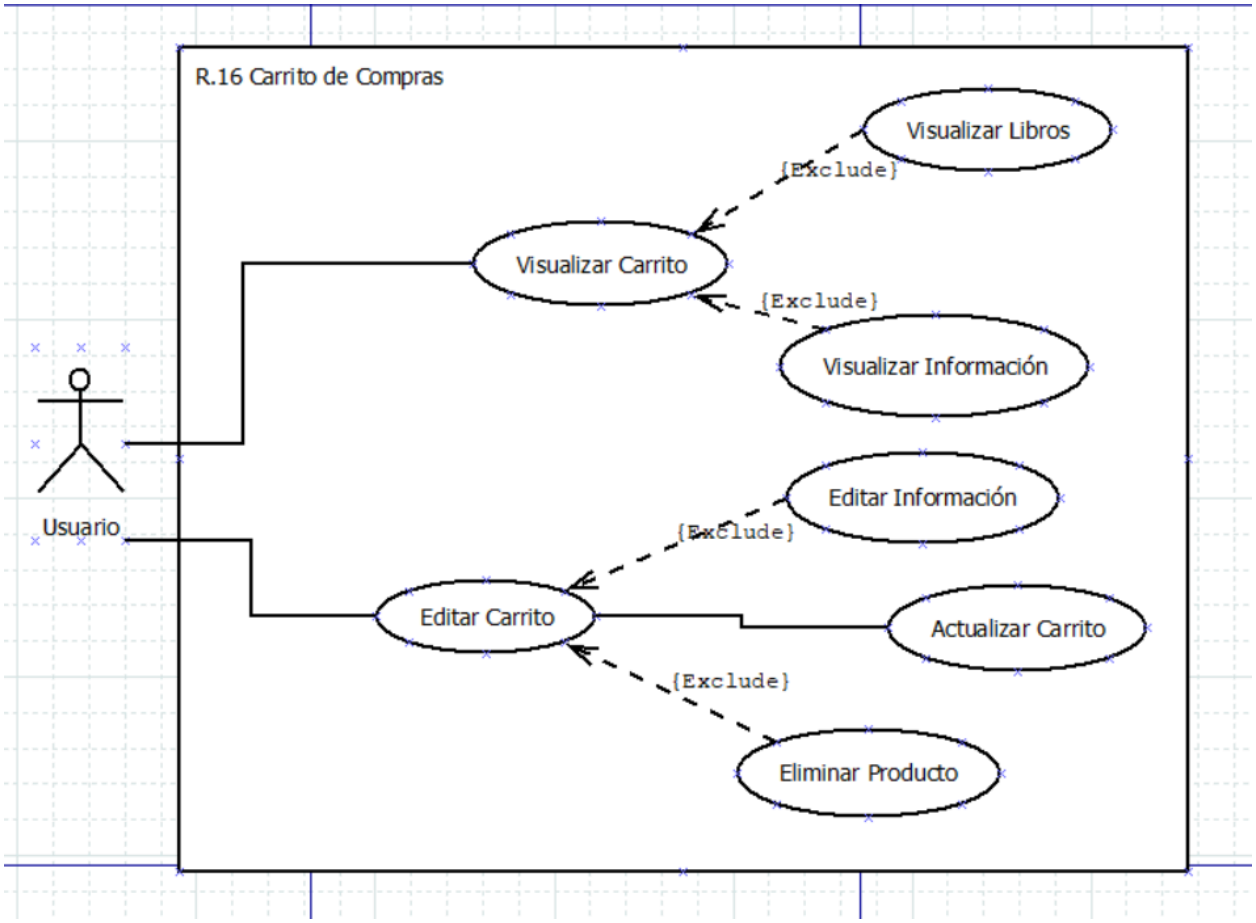
Criterios de aceptación

- El sistema ha de actualizar el carrito de compras del usuario
- El sistema ha de guardar la información del carrito de compras para luego ser procesado en la compra

Nota: La tabla nos menciona la necesidad de un carrito de compras dentro de la página para almacenar de forma temporal los productos deseados por el usuario, este ha de poder ser editado

Figura 19

Caso de Uso n°16 [Carrito de Compras]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario al momento de entrar al carrito de compras, poder visualizar su contenido y productos, además de poder editar el carrito de compras con poder eliminar sus productos o editar información para la compra, una vez editado se actualiza el carrito de compras

Tabla 17
Requerimiento funcional n° 17 [Entrega]

Tipo:	**Prioridad:**
Necesario	Media

Descripción:

El usuario ha de poder ver el estado de su pedido, si este anda en el almacén, en camino, fue regresado, etc.

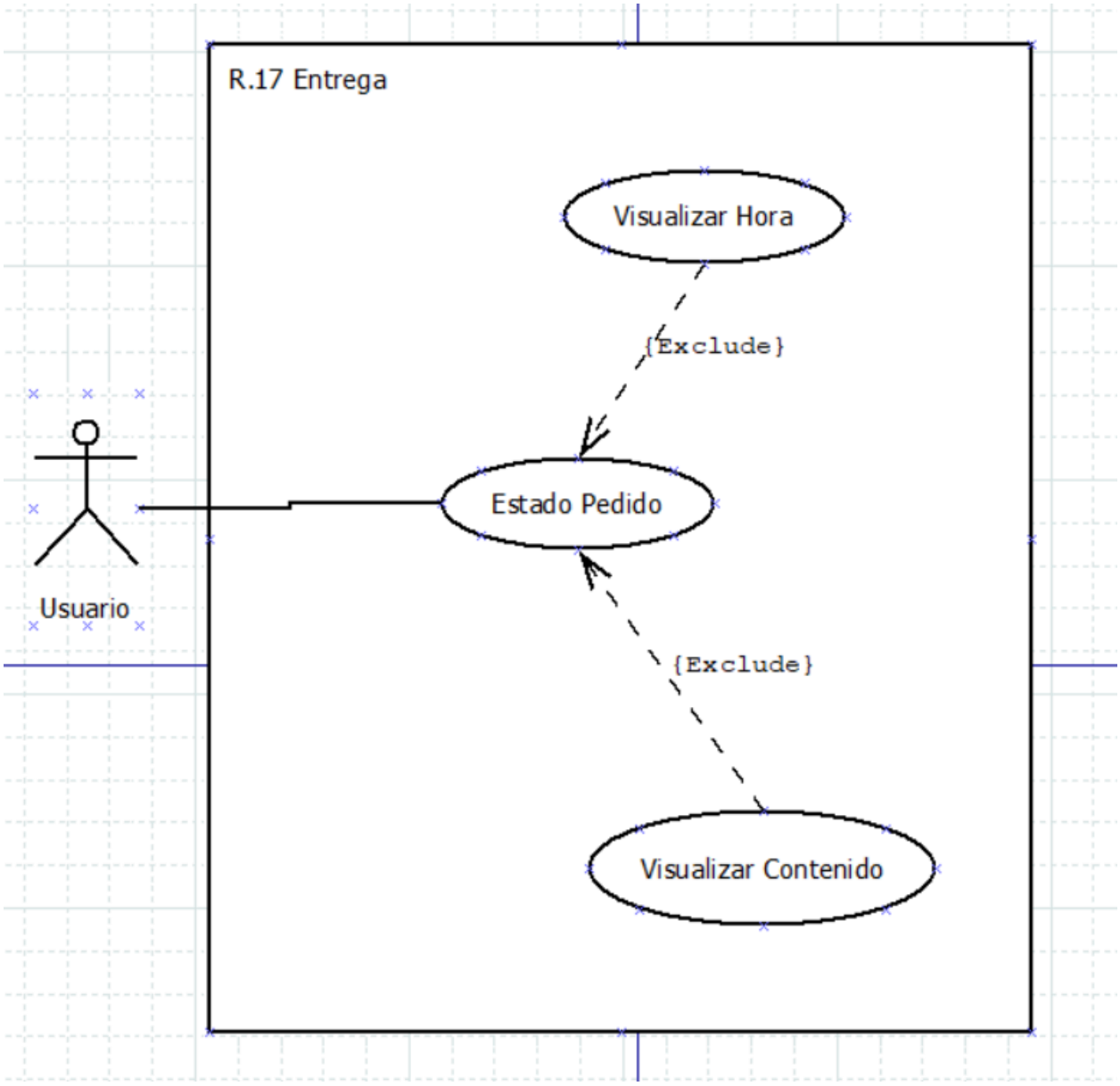
Criterios de aceptación

- El administrador ha de poder cambiar el estado de entrega del pedido
- El sistema ha de actualizar la información la cual únicamente va a ser visible para el usuario

Nota: La tabla nos menciona el cómo se le ha de mostrar y poder actualizar el estado de entrega de un producto al usuario, no siendo necesario un trackeo, si no un simple cambio de estado en el nombre: "**en camino, fue regresado, etc*****"***

Figura 20

Caso de Uso n°17 [Entrega]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario al momento de revisar en qué estado se encuentra su pedido para la entrega del producto o libro que compró

Taba 18

Requerimiento funcional n° 18 [Historial de compra]

Tipo:	**Prioridad:**
Necesario	Alta

Descripción:

El usuario ha de poder ver su historial de compras que ha tenido en la página

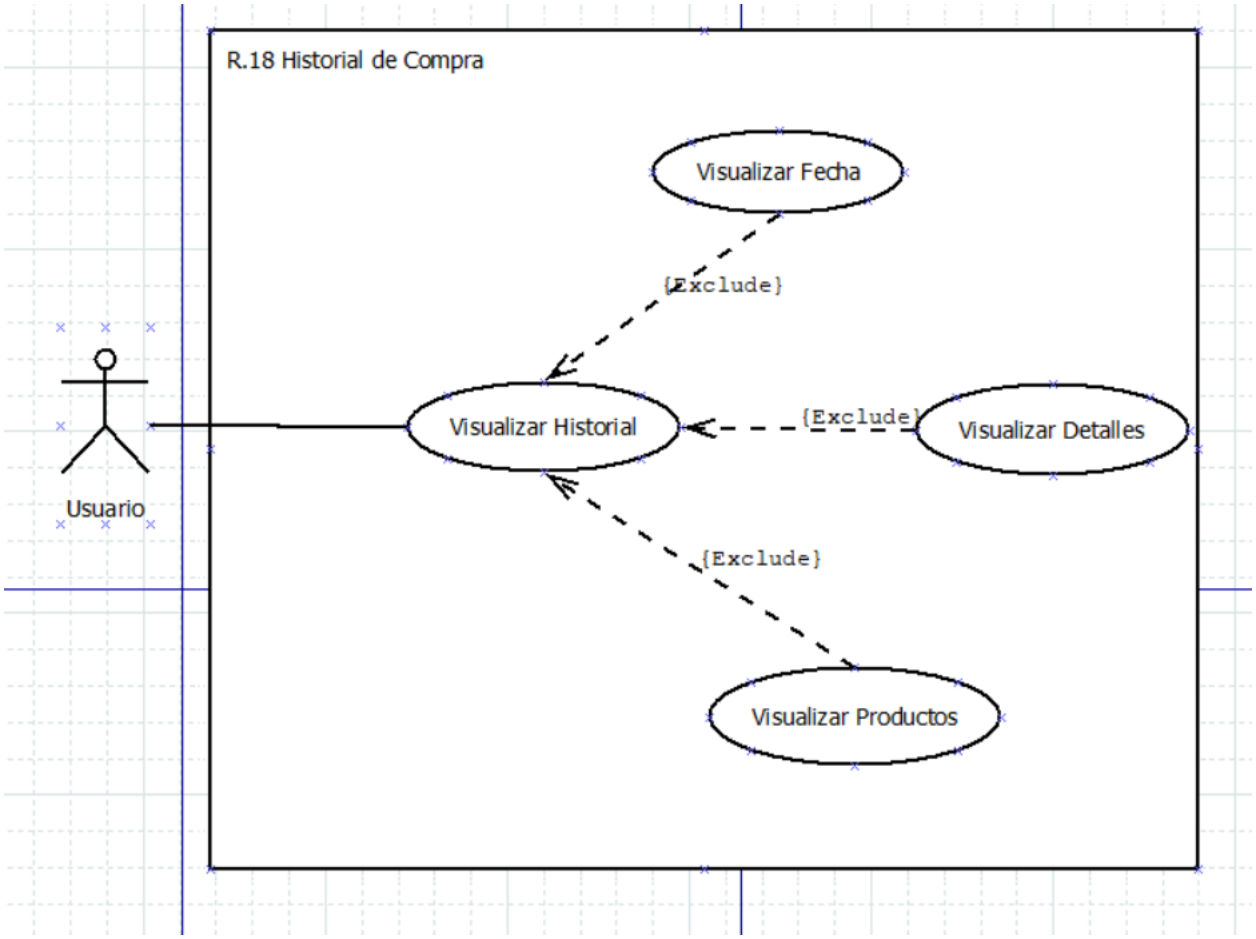
Criterios de aceptación

- El sistema ha de mostrar todas las compras que ha tenido el usuario

Nota: la tabla nos menciona que todos los usuarios han de tener la capacidad de poder ver su historial de compras único

Figura 21

Caso de Uso n°18 [Historial de Compra]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario al momento de revisar el historial de compras que ha realizado al tener una cuenta con el sitio web

Taba 19

Requerimiento funcional n° 19 [Buscar en el registro de compra (Administrador)]

Tipo:	**Prioridad:**
Necesario	Alta

Descripción:

El administrador ha de poder consultar los registros de las compras de los usuarios en orden de más reciente a menos reciente, en un lapso de 1 mes y/o buscar por ID del recibo

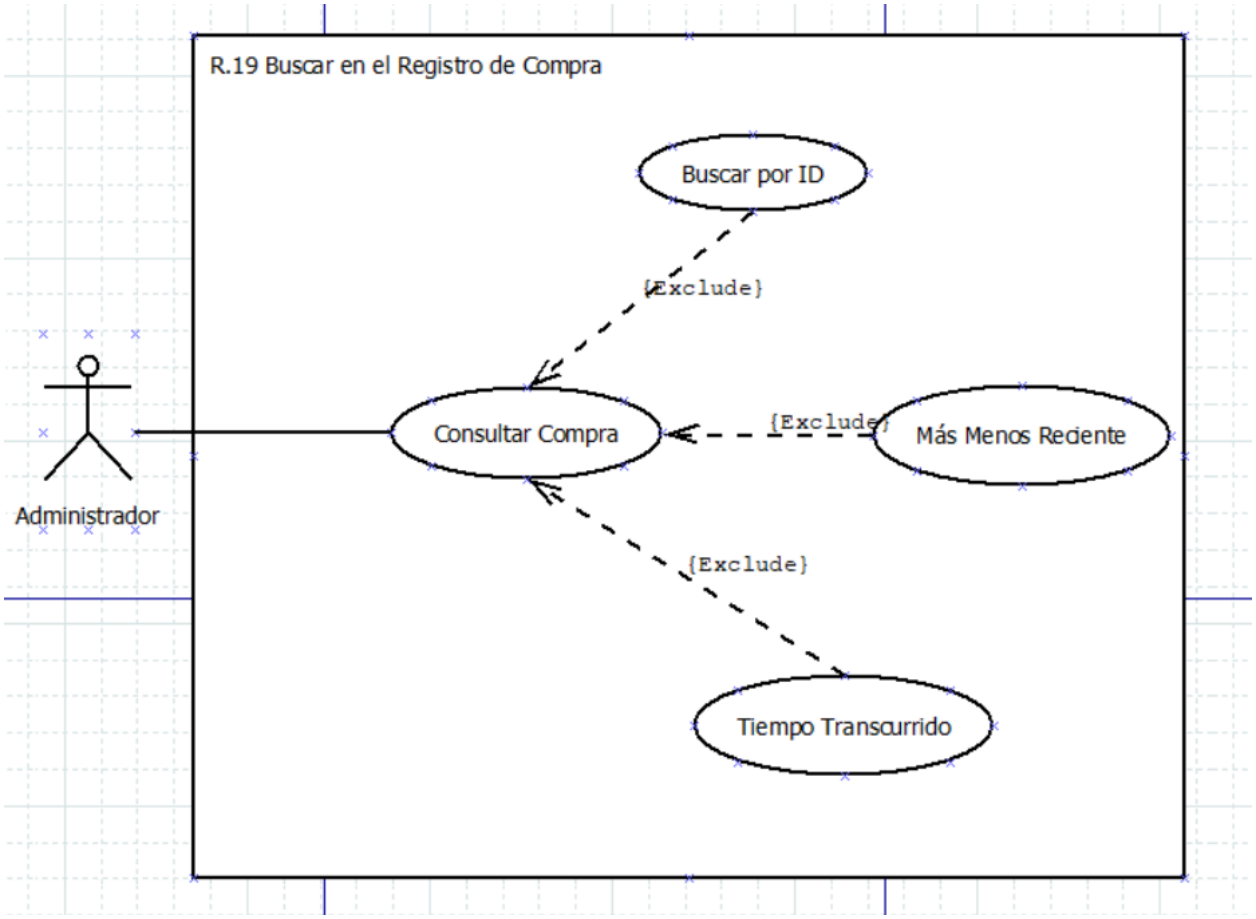
Criterios de aceptación

- El sistema ha de cargar los últimos registros de compras
- El sistema ha de permitir buscar por un registro en específico con su ID

Nota: La tabla nos menciona que el administrador de la página ha de tener la capacidad de poder buscar los registros de compras hechos en su pagina desde el menos reciente al más reciente

Figura 22

Caso de Uso n°19 [Buscar en el Registro de Compra]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario al momento de consultar las compras que se han realizado por medio de clientes en la página

Taba 20
Requerimiento funcional n° 20 [Reseñas]

Tipo:	**Prioridad:**
Necesario	Alta

Descripción:

El usuario ha de poder dejar una reseña en el libro deseado o que esté mirando, solo si este tiene una cuenta dentro de la página y su sesión está activa en el momento de crear la reseña

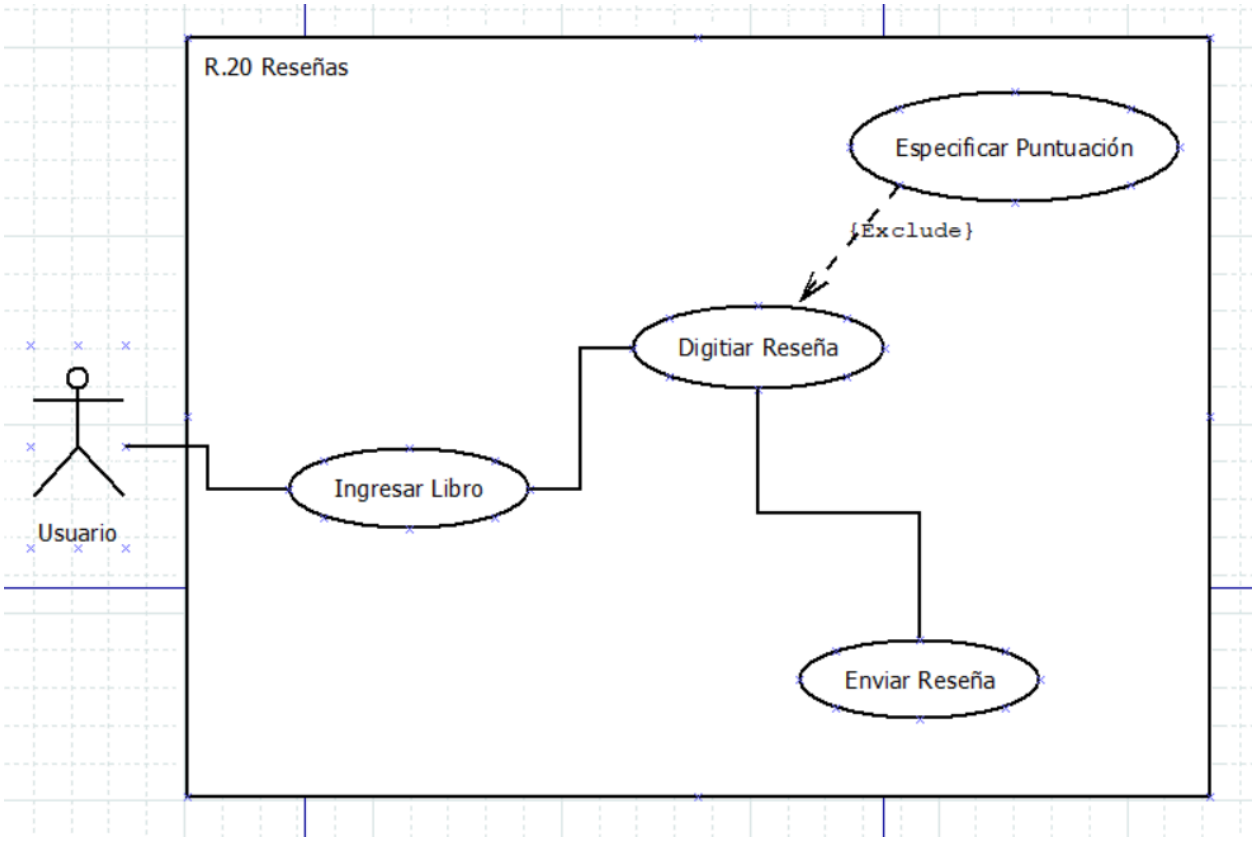
Criterios de aceptación

- La reseña digitada no debe ser vacía
- La reseña del usuario se ha de guardar correctamente en la base de datos
- La reseña del usuario se ha de mostrar a demás usuarios inclusive a él mismo

Nota: La tabla nos menciona las capacidades del usuario dentro de la página al momento de escribir una reseña en algún libro

Figura 23

Caso de Uso n°20 [Reseñas]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario al momento de realizar las reseñas necesarias para los libros que quiera dentro de la página

Taba 21

Requerimiento funcional n° 21 [Procesamiento de pagos]

Tipo:	**Prioridad:**
Necesario	Alta

Descripción:

El sistema debe procesar los pagos y verificar la validez del método de pago seleccionado por el usuario

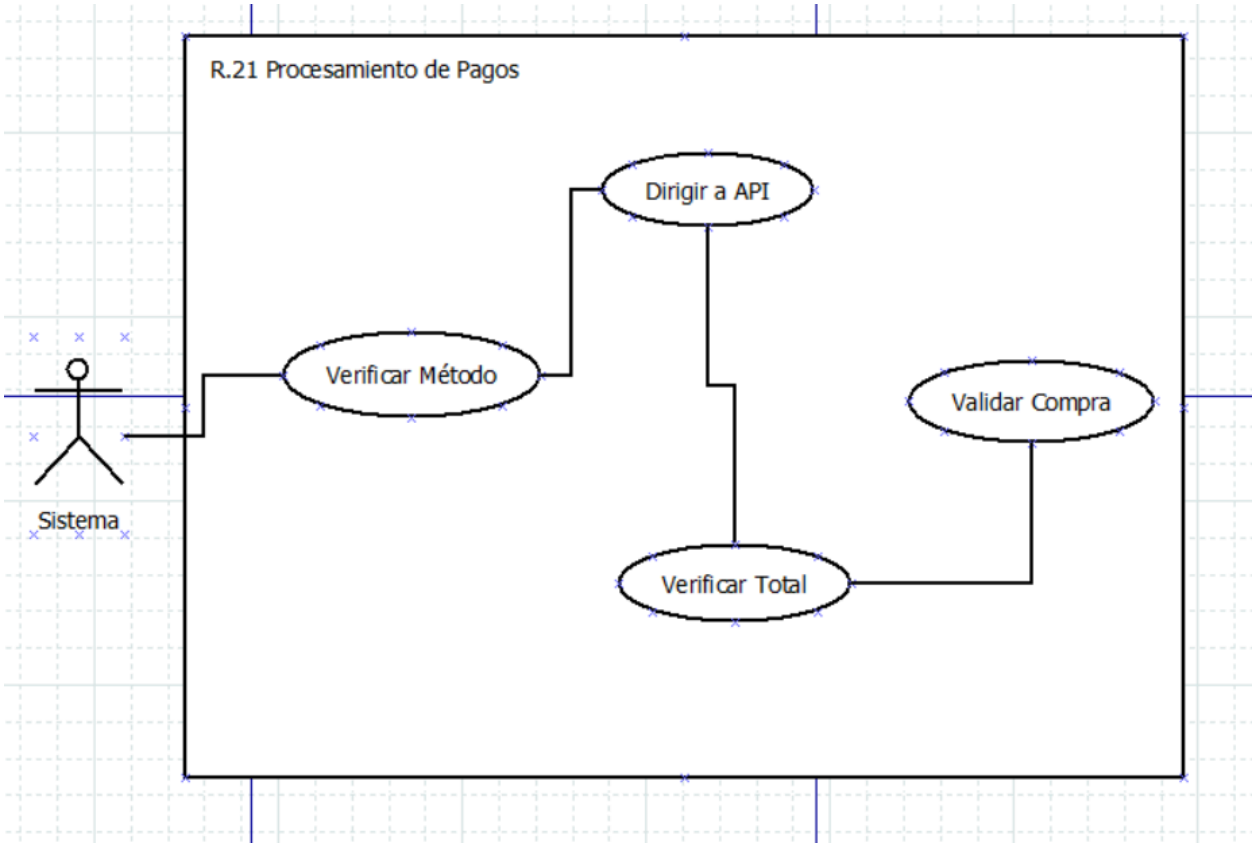
Criterios de aceptación

- El pago debe ser válido
- En caso de que el pago sea negado se le ha de informar al usuario que su pago no fue procesado

Nota: Esta tabla menciona que la pasarela de base de datos usada ha de procesar correctamente los pagos y que ha de ocurrir en caso de que falle

Figura 24

Caso de Uso n°21 [Procesamiento de Pagos]



Nota: Esta figura ilustra mediante caso de uso la funcionalidad e interacción del usuario al momento de realizar la compra, en este caso se muestra como el sistema se comporta redirigiendo a la API encargada de generar el pago y validar la compra