

INFORME TÉCNICO DEL SISTEMA BIBLIOTECAWEBSERVICE

Sistema de Gestión de Biblioteca Digital

Base de Datos: BibliotecaBD

TABLA DE CONTENIDOS

1. *Introducción*
2. *Modelo de Datos (Entidades)*
3. *Elaboración de la API*
4. *Pruebas Automáticas del Sistema*
5. *Integración con el Frontend*
6. *Manuales de Usuario del Sistema*
7. *Conclusiones*
8. *Recomendaciones*
9. *Bibliografía*

1. INTRODUCCIÓN

1.1 Objetivo del Sistema

El sistema BibliotecaWebService tiene como propósito principal facilitar la gestión completa de una biblioteca digital, permitiendo a los usuarios realizar reservas de libros, préstamos, y a los administradores controlar todo el inventario y las operaciones del sistema de manera eficiente y organizada.

El sistema busca modernizar los procesos tradicionales de una biblioteca, eliminando la necesidad de registros manuales y proporcionando una plataforma accesible desde cualquier dispositivo con conexión a internet.

1.2 Alcance del Sistema

El sistema cubre las siguientes funcionalidades principales:

Para Usuarios (Lectores):

- *Consulta del catálogo de libros disponibles*
- *Realización de reservas de libros*
- *Visualización del estado de sus préstamos activos*
- *Consulta de multas pendientes*

- *Acceso a estadísticas personales*

Para Administradores:

- *Gestión completa del inventario de libros*
- *Administración de autores, categorías y editoriales*
- *Control de préstamos y reservas*
- *Aprobación y rechazo de reservas*
- *Confirmación de entregas de libros*
- *Gestión de multas*
- *Acceso a estadísticas generales del sistema*

Funcionalidades del Sistema:

- *Autenticación segura mediante tokens*
- *Control de roles (Administrador y Usuario)*
- *Gestión de stock en tiempo real*
- *Sistema de reservas con estados (Pendiente, Aprobada, Rechazada, Completada, Cancelada)*
- *Generación automática de multas por retrasos*
- *Interfaz web moderna y responsiva*

1.3 Justificación

En la actualidad, las bibliotecas tradicionales enfrentan desafíos significativos en la gestión de sus recursos. Los sistemas manuales son propensos a errores, requieren mucho tiempo y no permiten un acceso eficiente a la información.

BibliotecaWebService resuelve estos problemas proporcionando:

- 10. Automatización de Procesos: Reduce el tiempo necesario para realizar préstamos y reservas*
- 11. Acceso Remoto: Los usuarios pueden consultar disponibilidad y realizar reservas desde cualquier lugar*
- 12. Control de Inventario: Los administradores tienen visibilidad completa del estado de todos los libros*
- 13. Trazabilidad: Cada operación queda registrada en el sistema*
- 14. Escalabilidad: El sistema puede crecer según las necesidades de la biblioteca*

2. MODELO DE DATOS (ENTIDADES)

2.1 Descripción de Entidades

El sistema BibliotecaWebService utiliza una base de datos relacional que almacena información sobre libros, usuarios, préstamos, reservas y multas. A continuación se describen las principales entidades:

Usuario

- *Almacena la información de las personas que utilizan el sistema*
- *Cada usuario tiene un nombre completo, correo electrónico y contraseña*
- *Los usuarios pueden tener el rol de Administrador o Usuario regular*
- *Un usuario puede tener múltiples préstamos, reservas y multas*

Libro

- *Representa cada ejemplar disponible en la biblioteca*
- *Contiene información como título, ISBN, año de publicación y cantidad disponible*
- *Cada libro pertenece a una categoría y una editorial*
- *Un libro puede tener múltiples autores*
- *Se relaciona con préstamos y reservas*

Categoría

- *Clasifica los libros por temas o áreas de conocimiento*
- *Ejemplos: Ficción, Ciencia, Historia, etc.*
- *Una categoría puede contener muchos libros*

Editorial

- *Representa la casa editorial que publicó el libro*
- *Almacena nombre y dirección*
- *Una editorial puede publicar muchos libros*

Autor

- *Representa a los escritores de los libros*
- *Un autor puede escribir muchos libros*
- *Un libro puede tener muchos autores (relación muchos a muchos)*

Reserva

- *Registra cuando un usuario solicita un libro*
- *Tiene estados: Pendiente, Aprobada, Rechazada, Completada, Cancelada*
- *Incluye fechas de reserva, expiración y devolución prevista*
- *Una reserva aprobada puede convertirse en un préstamo*

Préstamo

- *Registra cuando un libro es entregado físicamente a un usuario*
- *Incluye fechas de préstamo, devolución prevista y devolución real*
- *Puede originarse de una reserva o ser un préstamo directo*
- *Un préstamo puede contener múltiples libros (detalles de préstamo)*

Detalle de Préstamo

- *Representa cada libro individual dentro de un préstamo*
- *Permite que un préstamo incluya múltiples libros*
- *Se relaciona con multas si hay retrasos*

Multa

- *Se genera automáticamente cuando un préstamo se retrasa*
- *Tiene estados: Pendiente, Pagada, Anulada*
- *Incluye monto y fechas de generación y pago*

2.2 Relaciones entre Entidades

Las entidades se relacionan de la siguiente manera:

- *Usuario → Préstamo: Un usuario puede tener muchos préstamos*
- *Usuario → Reserva: Un usuario puede tener muchas reservas*
- *Usuario → Multa: Un usuario puede tener muchas multas*
- *Libro → Categoría: Muchos libros pertenecen a una categoría*
- *Libro → Editorial: Muchos libros pertenecen a una editorial*
- *Libro → Autor: Relación muchos a muchos (un libro puede tener varios autores)*
- *Reserva → Préstamo: Una reserva puede convertirse en un préstamo (relación uno a uno opcional)*
- *Préstamo → Detalle de Préstamo: Un préstamo puede tener muchos detalles*
- *Detalle de Préstamo → Multa: Un detalle puede tener una multa (relación uno a uno opcional)*

2.3 Migraciones en Entity Framework

El sistema utiliza Entity Framework Core para gestionar la base de datos. Las migraciones permiten crear y actualizar la estructura de la base de datos de manera controlada.

Proceso de Migración:

- 15. Se definen los modelos en código*
- 16. Entity Framework genera las migraciones automáticamente*
- 17. Las migraciones se aplican a la base de datos*
- 18. Esto asegura que la estructura de la base de datos siempre coincida con los modelos del código*

Ventajas:

- *Control de versiones de la estructura de la base de datos*
- *Facilita el trabajo en equipo*
- *Permite revertir cambios si es necesario*
- *Mantiene la consistencia entre desarrollo y producción*

2.4 Diagrama de la Base de Datos

[ESPACIO PARA DIAGRAMA DE BASE DE DATOS]

Prompt para generar el diagrama:

Crear un diagrama entidad-relación (ER) que muestre todas las tablas de la base de datos BibliotecaBD: Usuarios, Libros, Categorías, Editoriales, Autores, Reservas, Préstamos, DetallesPrestamo, Multas, y la tabla intermedia LibroAutor. Mostrar todas las relaciones entre las entidades con sus cardinalidades (uno a uno, uno a muchos, muchos a muchos). Incluir los campos principales de cada tabla. Usar colores diferentes para distinguir tipos de entidades (usuarios en azul, libros en verde, transacciones en naranja).

3. ELABORACIÓN DE LA API

3.1 Lista de Endpoints

El sistema BibliotecaWebService expone una API REST que permite interactuar con todos los recursos del sistema. A continuación se listan los principales endpoints organizados por funcionalidad:

Autenticación:

- *POST /api/Auth/register - Registrar nuevo usuario*
- *POST /api/Auth/login - Iniciar sesión*

Libros:

- *GET /api/Libros - Obtener todos los libros*
- *GET /api/Libros/{id} - Obtener un libro específico*
- *POST /api/Libros - Crear nuevo libro (Solo Administrador)*
- *PUT /api/Libros/{id} - Actualizar libro (Solo Administrador)*
- *DELETE /api/Libros/{id} - Eliminar libro (Solo Administrador)*

Autores:

- *GET /api/Autores - Obtener todos los autores*
- *GET /api/Autores/{id} - Obtener un autor específico*
- *POST /api/Autores - Crear nuevo autor (Solo Administrador)*
- *PUT /api/Autores/{id} - Actualizar autor (Solo Administrador)*
- *DELETE /api/Autores/{id} - Eliminar autor (Solo Administrador)*

Categorías:

- *GET /api/Categorias - Obtener todas las categorías*
- *GET /api/Categorias/{id} - Obtener una categoría específica*
- *POST /api/Categorias - Crear nueva categoría (Solo Administrador)*
- *PUT /api/Categorias/{id} - Actualizar categoría (Solo Administrador)*
- *DELETE /api/Categorias/{id} - Eliminar categoría (Solo Administrador)*

Editoriales:

- *GET /api/Editoriales - Obtener todas las editoriales*
- *GET /api/Editoriales/{id} - Obtener una editorial específica*
- *POST /api/Editoriales - Crear nueva editorial (Solo Administrador)*
- *PUT /api/Editoriales/{id} - Actualizar editorial (Solo Administrador)*
- *DELETE /api/Editoriales/{id} - Eliminar editorial (Solo Administrador)*

Reservas:

- *GET /api/Reservas - Obtener reservas (propias para usuarios, todas para administradores)*
- *GET /api/Reservas/{id} - Obtener una reserva específica*
- *POST /api/Reservas - Crear nueva reserva*
- *DELETE /api/Reservas/{id} - Cancelar reserva*
- *POST /api/Reservas/{id}/aprobar - Aprobar reserva (Solo Administrador)*
- *POST /api/Reservas/{id}/rechazar - Rechazar reserva (Solo Administrador)*

Préstamos:

- *GET /api/Prestamos - Obtener préstamos (propios para usuarios, todos para administradores)*
- *GET /api/Prestamos/{id} - Obtener un préstamo específico*
- *POST /api/Prestamos - Crear préstamo directo (Solo Administrador)*
- *POST /api/Prestamos/confirmar-entrega/{reservaId} - Confirmar entrega de reserva (Solo Administrador)*
- *PUT /api/Prestamos/{id}/devolver - Devolver libro*

Multas:

- *GET /api/Multas - Obtener multas (propias para usuarios, todas para administradores)*
- *GET /api/Multas/{id} - Obtener una multa específica*
- *PUT /api/Multas/{id}/pagar - Marcar multa como pagada (Solo Administrador)*

Estadísticas:

- *GET /api/Estadisticas/dashboard - Obtener estadísticas del dashboard*
- *GET /api/Estadisticas/mensual - Obtener estadísticas mensuales*
- *GET /api/Estadisticas/distribucion - Obtener distribución de datos*

3.2 Detalle de los Endpoints

Ejemplo: Crear Reserva

Endpoint: POST /api/Reservas

Descripción: Permite a un usuario crear una nueva reserva de un libro.

Autenticación: Requerida (token JWT)

Cuerpo de la Solicitud:

```
{
  "libroId": 1,
  "fechaDevolucionPrevista": "2024-12-31T00:00:00Z"
}
```

Respuesta Exitosa (200 OK):

```
{
  "id": 1,
  "libroId": 1,
  "libroTitulo": "El Quijote",
  "fechaReserva": "2024-01-15T10:30:00Z",
  "fechaExpiracion": "2024-01-18T10:30:00Z",
  "fechaDevolucionPrevista": "2024-12-31T00:00:00Z",
  "estado": "Pendiente"
}
```

Flujo del Proceso:

19. El usuario selecciona un libro disponible
20. El usuario especifica la fecha en que planea devolver el libro
21. El sistema crea la reserva con estado "Pendiente"
22. El administrador revisa y aprueba o rechaza la reserva
23. Si se aprueba, cuando el usuario recoge el libro, el administrador confirma la entrega
24. Se crea automáticamente un préstamo y se descuenta el stock del libro

Ejemplo: Aprobar Reserva

Endpoint: POST /api/Reservas/{id}/aprobar

Descripción: Permite a un administrador aprobar una reserva pendiente.

Autenticación: Requerida (token JWT con rol Administrador)

Respuesta Exitosa (200 OK):

```
{
  "message": "Reserva aprobada exitosamente. Ahora puede
```

```
confirmar la entrega en la sección de Préstamos."
```

```
}
```

Ejemplo: Confirmar Entrega de Reserva

Endpoint: POST /api/Prestamos/confirmar-entrega/{reservaId}

Descripción: Cuando un usuario recoge físicamente un libro de una reserva aprobada, el administrador confirma la entrega. Esto crea el préstamo y descuenta el stock.

Autenticación: Requerida (token JWT con rol Administrador)

Respuesta Exitosa (200 OK):

```
{
  "id": 1,
  "usuarioId": "user123",
  "fechaPrestamo": "2024-01-20T14:00:00Z",
  "fechaDevolucionPrevista": "2024-12-31T00:00:00Z",
  "reservaId": 1,
  "detalles": [
    {
      "libroId": 1,
      "libroTitulo": "El Quijote",
      "cantidad": 1
    }
  ]
}
```

3.3 Seguridad y Roles

El sistema implementa un sistema de seguridad robusto basado en tokens JWT (JSON Web Tokens) y control de acceso basado en roles.

Autenticación:

- *Todos los endpoints (excepto registro y login) requieren un token JWT válido*
- *El token se obtiene al iniciar sesión*
- *El token tiene una duración de 60 minutos*
- *El token debe incluirse en el header de cada solicitud: `Authorization: Bearer {token}`*

Roles del Sistema:

Usuario (Lector):

- *Puede consultar libros, autores, categorías y editoriales*
- *Puede crear reservas*
- *Puede ver sus propios préstamos y reservas*

- *Puede cancelar sus propias reservas*
- *Puede ver sus multas*
- *Puede devolver libros prestados*

Administrador:

- *Tiene todos los permisos de Usuario*
- *Puede crear, editar y eliminar libros, autores, categorías y editoriales*
- *Puede ver todas las reservas y préstamos del sistema*
- *Puede aprobar o rechazar reservas*
- *Puede crear préstamos directos*
- *Puede confirmar entregas de reservas*
- *Puede marcar multas como pagadas*
- *Tiene acceso a estadísticas completas del sistema*

Configuración de Seguridad:

- *Las contraseñas se almacenan de forma encriptada*
- *Los tokens JWT se firman con una clave secreta*
- *Se valida el origen de las solicitudes (CORS)*
- *Se verifica la validez del token en cada solicitud*

4. PRUEBAS AUTOMÁTICAS DEL SISTEMA

El sistema incluye un conjunto completo de pruebas automáticas que verifican el correcto funcionamiento de todos los componentes.

4.1 Tipos de Pruebas

Pruebas Unitarias:

- *Verifican el funcionamiento individual de cada componente*
- *Se prueban los controladores de la API*
- *Se validan las operaciones de creación, lectura, actualización y eliminación*
- *Se verifica el manejo de errores*

Pruebas de Integración:

- *Verifican que los componentes trabajen correctamente juntos*
- *Se prueban los flujos completos de operaciones*
- *Se valida la interacción con la base de datos*

4.2 Cobertura de Pruebas

Las pruebas cubren los siguientes aspectos:

Autenticación:

- *Registro de usuarios exitoso*
- *Registro con usuario existente (debe fallar)*
- *Login con credenciales válidas*
- *Login con credenciales inválidas*

Gestión de Libros:

- *Creación de libros*
- *Validación de stock disponible*
- *Actualización de libros*
- *Eliminación de libros*

Reservas:

- *Creación de reservas*
- *Aprobación de reservas*
- *Rechazo de reservas*
- *Cancelación de reservas*
- *Validación de que el stock no se descuenta al crear reserva*

Préstamos:

- *Creación de préstamos directos*
- *Confirmación de entrega de reservas*
- *Validación de que el stock se descuenta al confirmar entrega*
- *Devolución de libros*

Validaciones:

- *Fechas en el pasado (deben rechazarse)*
- *Stock insuficiente (debe rechazarse)*
- *Operaciones sin permisos (deben rechazarse)*

4.3 Herramientas de Pruebas

El sistema utiliza xUnit como framework de pruebas, que es el estándar para aplicaciones .NET. Las pruebas se ejecutan automáticamente antes de cada despliegue para asegurar que no se introduzcan errores.

5. INTEGRACIÓN CON EL FRONTEND

5.1 Arquitectura del Sistema

El sistema sigue una arquitectura de tres capas:

Frontend (Interfaz de Usuario):

- *Desarrollado con React y Vite*
- *Interfaz moderna y responsiva*
- *Se comunica con el backend mediante peticiones HTTP*

Backend (API):

- *Desarrollado con ASP.NET Core*
- *Proporciona servicios REST*
- *Gestiona la lógica de negocio*
- *Interactúa con la base de datos*

Base de Datos:

- *SQL Server*
- *Almacena toda la información del sistema*

[ESPACIO PARA DIAGRAMA DE ARQUITECTURA]

Prompt para generar el diagrama:

Crear un diagrama de arquitectura de tres capas mostrando: 1) Capa de Presentación (Frontend React con componentes de usuario), 2) Capa de Aplicación (Backend ASP.NET Core con controladores API), 3) Capa de Datos (SQL Server con base de datos BibliotecaBD). Mostrar las flechas de comunicación entre capas. Incluir elementos como navegador web, servidor de aplicaciones, y servidor de base de datos. Usar colores diferentes para cada capa.

5.2 Flujo de Trabajo del Sistema

Desde la Vista del Usuario:

25. Acceso al Sistema:

- *El usuario ingresa a la aplicación web*
- *Si no está autenticado, se le muestra la pantalla de login*
- *Ingresa sus credenciales (correo y contraseña)*
- *El sistema valida las credenciales y proporciona un token*

26. Navegación:

- *Una vez autenticado, el usuario ve el dashboard*
- *Puede navegar por las diferentes secciones: Libros, Reservas, Préstamos, Multas*
- *Cada sección muestra información relevante para el usuario*

27. Realizar una Reserva:

- *El usuario busca un libro en el catálogo*
- *Selecciona el libro deseado*
- *Especifica la fecha en que planea devolverlo*
- *Confirma la reserva*
- *El sistema crea la reserva con estado "Pendiente"*
- *El usuario puede ver el estado de su reserva*

28. Seguimiento:

- *El usuario puede ver todas sus reservas*
- *Puede ver si fueron aprobadas o rechazadas*
- *Si fue aprobada, puede acudir a la biblioteca a recoger el libro*

29. Recoger el Libro:

- *Cuando el usuario llega a la biblioteca*
- *El administrador confirma la entrega en el sistema*
- *Se crea automáticamente el préstamo*
- *El usuario puede ver su préstamo activo*

30. Devolver el Libro:

- *Cuando el usuario devuelve el libro*
- *El administrador registra la devolución*
- *Si hay retraso, se genera automáticamente una multa*
- *El stock del libro se incrementa automáticamente*

Desde la Vista del Administrador:

31. Gestión de Inventario:

- *El administrador puede agregar nuevos libros*
- *Puede editar información de libros existentes*
- *Puede gestionar autores, categorías y editoriales*
- *Tiene visibilidad completa del stock disponible*

32. Gestión de Reservas:

- *Ve todas las reservas pendientes*
- *Puede aprobar o rechazar reservas*
- *Al aprobar, la reserva queda lista para que el usuario la recoja*
- *Al rechazar, se notifica al usuario*

33. Gestión de Préstamos:

- *Ve todos los préstamos activos*
- *Puede crear préstamos directos (sin reserva previa)*
- *Confirma entregas de reservas aprobadas*
- *Registra devoluciones de libros*

34. Gestión de Multas:

- *Ve todas las multas del sistema*
- *Puede marcar multas como pagadas*
- *Tiene visibilidad de multas pendientes*

35. Estadísticas:

- *Accede a dashboard con estadísticas generales*
- *Ve gráficos de actividad mensual*
- *Analiza distribución de préstamos y reservas*

[ESPACIO PARA DIAGRAMA DE FLUJO DE TRABAJO]

Prompt para generar el diagrama:

Crear un diagrama de flujo que muestre el proceso completo desde que un usuario hace una reserva hasta que devuelve el libro. Incluir: 1) Usuario crea reserva (Pendiente), 2) Administrador aprueba reserva (Aprobada), 3) Usuario recoge libro (Administrador confirma entrega), 4) Se crea préstamo y se descuenta stock, 5) Usuario devuelve libro, 6) Se registra devolución, 7) Si hay retraso se genera multa. Mostrar los estados de la reserva en cada paso. Usar formas diferentes para procesos (rectángulos), decisiones (rombos) y estados (elipses).

5.3 Manual de Despliegue en la Nube

El sistema está diseñado para desplegarse en la plataforma Render, que permite alojar tanto el backend como el frontend de manera gratuita.

Requisitos Previos:

- *Cuenta en GitHub con el código del proyecto*

- Cuenta en Render (gratuita)
- Base de datos SQL Server accesible desde internet

Despliegue del Backend:

36. Preparación:

- El código del backend está en la carpeta *WebApplication3*
- Incluye un archivo *Dockerfile* para containerización
- Incluye un archivo *render.yaml* con la configuración

37. Configuración en Render:

- Crear un nuevo Web Service
- Conectar el repositorio de GitHub
- Render detecta automáticamente el *Dockerfile*
- Configurar las variables de entorno:
- *ConnectionStrings__DefaultConnection*: Cadena de conexión a la base de datos
- *Jwt__Key*: Clave secreta para firmar tokens
- *Jwt__Issuer*: URL del backend (<https://biblioteca.onrender.com>)
- *Jwt__Audience*: URL del backend
- *ASPNETCORE_ENVIRONMENT*: *Production*
- *CORS_ORIGINS*: URL del frontend (<https://frontedbiblioteca.onrender.com>)

38. Despliegue:

- Render construye automáticamente la imagen Docker
- Inicia el servicio
- El backend queda disponible en la URL proporcionada

Despliegue del Frontend:

39. Preparación:

- El código del frontend está en la carpeta *frontend*
- Incluye un archivo *render.yaml* con la configuración

40. Configuración en Render:

- Crear un nuevo Web Service
- Conectar el repositorio de GitHub
- Configurar:
- Build Command: ``npm install && npm run build``
- Start Command: ``npm start``

- *Variables de entorno:*
- *VITE_API_URL: URL del backend (<https://biblioteca.onrender.com/api>)*
- *NODE_VERSION: 18.17.0*

41. Despliegue:

- *Render instala las dependencias*
- *Construye la aplicación*
- *Inicia el servidor*
- *El frontend queda disponible en la URL proporcionada*

Configuración de CORS:

- *El backend debe permitir solicitudes desde el frontend*
- *Se configura en el archivo Program.cs del backend*
- *Se agregan las URLs permitidas en appsettings.Production.json*

[ESPACIO PARA DIAGRAMA DE DESPLIEGUE]

Prompt para generar el diagrama:

Crear un diagrama de despliegue en la nube mostrando: 1) Repositorio GitHub con código, 2) Plataforma Render con dos servicios (Backend y Frontend), 3) Base de datos SQL Server en la nube, 4) Usuarios accediendo desde navegadores web. Mostrar las conexiones entre componentes. Incluir elementos como contenedores Docker, servidores web, y bases de datos. Mostrar el flujo de datos desde el usuario hasta la base de datos y viceversa. Usar iconos representativos para cada componente.

5.4 Capturas del Consumo del API por el Frontend

El frontend consume la API del backend para todas las operaciones. A continuación se describen las principales interacciones:

Autenticación:

- *El frontend envía credenciales al endpoint /api/Auth/login*
- *Recibe un token JWT que almacena localmente*
- *Incluye este token en todas las solicitudes posteriores*

Consulta de Libros:

- *El frontend solicita la lista de libros a /api/Libros*
- *Muestra los libros en una tabla con información completa*
- *Permite filtrar y buscar libros*

Creación de Reservas:

- *El usuario completa un formulario en el frontend*
- *El frontend envía los datos a /api/Reservas*
- *Muestra confirmación y actualiza la lista de reservas*

Gestión de Préstamos:

- *El frontend consulta /api/Prestamos para mostrar préstamos activos*
- *Los administradores pueden confirmar entregas mediante /api/Prestamos/confirmar-entrega*
- *Se actualiza la interfaz en tiempo real*

Estadísticas:

- *El frontend consulta /api/Estadisticas para obtener datos*
- *Muestra gráficos y tablas con la información*
- *Actualiza automáticamente según los datos del sistema*

6. MANUALES DE USUARIO DEL SISTEMA

6.1 Manual para Usuarios (Lectores)

Iniciar Sesión:

42. *Abrir la aplicación web en el navegador*
43. *Hacer clic en "Iniciar Sesión"*
44. *Ingresar correo electrónico y contraseña*
45. *Hacer clic en "Entrar"*

Registrarse:

46. *Si no tienes cuenta, hacer clic en "Registrarse"*
47. *Completar el formulario con nombre completo, correo y contraseña*
48. *Hacer clic en "Registrar"*
49. *Iniciar sesión con las credenciales creadas*

Buscar y Reservar un Libro:

50. *Ir a la sección "Libros"*
51. *Buscar el libro deseado usando la barra de búsqueda*
52. *Verificar que el libro esté disponible (stock mayor a 0)*
53. *Hacer clic en "Nueva Reserva"*
54. *Seleccionar la fecha en que planeas devolver el libro*
55. *Confirmar la reserva*
56. *La reserva quedará con estado "Pendiente" hasta que el administrador la apruebe*

Ver Mis Reservas:

- 57. Ir a la sección "Reservas"*
- 58. Ver todas tus reservas con su estado actual*
- 59. Si una reserva está "Aprobada", puedes acudir a la biblioteca a recogerla*
- 60. Puedes cancelar reservas pendientes haciendo clic en el botón de eliminar*

Ver Mis Préstamos:

- 61. Ir a la sección "Préstamos"*
- 62. Ver todos tus préstamos activos*
- 63. Ver la fecha de devolución prevista*
- 64. Devolver el libro a tiempo para evitar multas*

Ver Mis Multas:

- 65. Ir a la sección "Multas"*
- 66. Ver todas tus multas pendientes y pagadas*
- 67. Contactar a la biblioteca para pagar multas pendientes*

6.2 Manual para Administradores

Gestión de Libros:

- 68. Ir a la sección "Libros"*
- 69. Para agregar un libro: hacer clic en "Nuevo Libro", completar el formulario con título, ISBN, año, categoría, editorial, autores y cantidad disponible*
- 70. Para editar: hacer clic en el botón de editar junto al libro*
- 71. Para eliminar: hacer clic en el botón de eliminar (solo si no hay préstamos activos)*

Gestión de Autores, Categorías y Editoriales:

- 72. Ir a las secciones correspondientes*
- 73. Usar los botones "Nuevo" para agregar elementos*
- 74. Editar o eliminar según sea necesario*

Aprobar o Rechazar Reservas:

- 75. Ir a la sección "Reservas"*
- 76. Ver todas las reservas pendientes*
- 77. Para aprobar: hacer clic en el botón de aprobar (✓)*
- 78. Para rechazar: hacer clic en el botón de rechazar (X)*
- 79. Las reservas aprobadas aparecerán en la sección de Préstamos para confirmar entrega*

Confirmar Entrega de Reserva:

- 80. Ir a la sección "Préstamos"*
- 81. En la sección "Reservas Pendientes de Entrega", ver las reservas aprobadas*

- 82. Cuando el usuario recoge el libro físicamente, hacer clic en "Confirmar Entrega"
- 83. El sistema creará automáticamente el préstamo y descontará el stock

Crear Préstamo Directo:

- 84. Ir a la sección "Préstamos"
- 85. Hacer clic en "Nuevo Préstamo"
- 86. Seleccionar el usuario
- 87. Agregar los libros a prestar
- 88. Establecer la fecha de devolución prevista
- 89. Confirmar el préstamo

Registrar Devolución:

- 90. Ir a la sección "Préstamos"
- 91. Encontrar el préstamo activo
- 92. Hacer clic en "Devolver"
- 93. El sistema registrará la devolución y, si hay retraso, generará una multa automáticamente

Gestionar Multas:

- 94. Ir a la sección "Multas"
- 95. Ver todas las multas del sistema
- 96. Cuando un usuario paga una multa, hacer clic en "Marcar como Pagada"
- 97. El estado de la multa cambiará a "Pagada"

Ver Estadísticas:

- 98. Ir al "Dashboard"
- 99. Ver las estadísticas generales del sistema
- 100. Analizar gráficos de actividad mensual
- 101. Revisar distribución de préstamos y reservas

7. CONCLUSIONES

El sistema BibliotecaWebService representa una solución completa y moderna para la gestión de bibliotecas digitales. A través de su implementación, se han logrado los siguientes objetivos:

Automatización de Procesos:

El sistema ha logrado automatizar completamente los procesos de reserva, préstamo y devolución de libros, eliminando la necesidad de registros manuales y reduciendo significativamente el tiempo requerido para estas operaciones.

Accesibilidad:

La implementación de una interfaz web moderna y responsiva permite que tanto usuarios como administradores accedan al sistema desde cualquier dispositivo con conexión a internet, facilitando la gestión remota de la biblioteca.

Control y Seguridad:

El sistema de autenticación basado en tokens JWT y el control de acceso por roles garantizan que solo usuarios autorizados puedan realizar operaciones específicas, manteniendo la integridad y seguridad de los datos.

Escalabilidad:

La arquitectura del sistema permite que crezca según las necesidades de la biblioteca, pudiendo manejar desde pequeñas colecciones hasta grandes inventarios sin requerir cambios significativos en la estructura.

Trazabilidad:

Cada operación realizada en el sistema queda registrada, permitiendo un seguimiento completo de todos los préstamos, reservas y multas, lo que facilita la auditoría y el análisis del uso de la biblioteca.

Experiencia del Usuario:

La interfaz intuitiva y los procesos simplificados mejoran significativamente la experiencia tanto de los usuarios que buscan libros como de los administradores que gestionan el sistema.

El sistema está completamente funcional y listo para su uso en producción, proporcionando una base sólida para la gestión moderna de bibliotecas.

8. RECOMENDACIONES

Basado en el desarrollo y análisis del sistema BibliotecaWebService, se presentan las siguientes recomendaciones para futuras mejoras y mantenimiento:

Mejoras de Funcionalidad:

- 102. Implementar un sistema de notificaciones por correo electrónico para informar a los usuarios sobre el estado de sus reservas y recordatorios de devolución*
- 103. Agregar un sistema de búsqueda avanzada que permita filtrar libros por múltiples criterios simultáneos*
- 104. Implementar un sistema de recomendaciones basado en el historial de préstamos del usuario*
- 105. Agregar la funcionalidad de renovación de préstamos desde la interfaz del usuario*
- 106. Implementar un sistema de reserva de espacios de estudio o salas de reunión*

Mejoras Técnicas:

- 107. *Implementar caché para consultas frecuentes y mejorar el rendimiento del sistema*
- 108. *Agregar logs más detallados para facilitar el debugging y monitoreo del sistema*
- 109. *Implementar backups automáticos de la base de datos*
- 110. *Considerar la implementación de un sistema de cola para manejar operaciones asíncronas*
- 111. *Agregar más pruebas automatizadas para aumentar la cobertura del código*

Mejoras de Seguridad:

- 112. *Implementar rate limiting para prevenir ataques de fuerza bruta*
- 113. *Agregar validación de entrada más estricta en todos los endpoints*
- 114. *Considerar la implementación de autenticación de dos factores para administradores*
- 115. *Realizar auditorías de seguridad periódicas*

Mejoras de Usabilidad:

- 116. *Agregar tutoriales interactivos para nuevos usuarios*
- 117. *Implementar un sistema de ayuda contextual en la interfaz*
- 118. *Mejorar la visualización de información en dispositivos móviles*
- 119. *Agregar opciones de personalización de la interfaz según preferencias del usuario*

Mejoras de Reportes:

- 120. *Implementar un sistema de generación de reportes en PDF*
- 121. *Agregar más gráficos y visualizaciones en el dashboard*
- 122. *Implementar exportación de datos a Excel*
- 123. *Agregar reportes personalizables según necesidades específicas*

Mantenimiento:

- 124. *Establecer un calendario regular de actualizaciones y mantenimiento*
- 125. *Documentar todos los cambios y actualizaciones del sistema*
- 126. *Capacitar al personal administrativo en el uso del sistema*
- 127. *Establecer un sistema de soporte técnico para usuarios*

9. BIBLIOGRAFÍA

Microsoft. (2024). ASP.NET Core Documentation. Recuperado de <https://docs.microsoft.com/en-us/aspnet/core/>

Microsoft. (2024). Entity Framework Core Documentation. Recuperado de <https://docs.microsoft.com/en-us/ef/core/>

React Team. (2024). React Documentation. Recuperado de <https://react.dev/>

Vite. (2024). Vite Documentation. Recuperado de <https://vitejs.dev/>

Render. (2024). Render Documentation. Recuperado de <https://render.com/docs>

xUnit. (2024). xUnit.net Documentation. Recuperado de <https://xunit.net/>

JWT.io. (2024). JSON Web Token Introduction. Recuperado de <https://jwt.io/introduction>

SQL Server Documentation. (2024). Microsoft SQL Server Documentation. Recuperado de <https://docs.microsoft.com/en-us/sql/>

FIN DEL INFORME