

PROYECTO
Sistema de Gestión de Biblioteca Áulica

Documentación Inicial del Sistema

Para el:
Instituto Superior de Formación Docente y Técnica N°166



Carrera:
Tecnicatura Superior en Análisis y Desarrollo de Aplicaciones
Año: 2024

Índice General

1 Introducción.....	3
2 Objetivos.....	4
3 Especificación de requisitos.....	5
3.1 Definición de Roles.....	5
3.2 Identificar y definir los Requisitos Funcionales del Sistema.....	5
3.3 Identificar y definir los Requisitos no Funcionales del Sistema.....	6
4 Base de Datos.....	8
4.1 Modelo Conceptual.....	8
4.2 Modelo Lógico.....	8
5 Diseño	11
5.1 Definir prototipos.....	11
5.2 Diseñar Interfaz pantalla principal.....	11
5.3 Diseñar Interfaz Usuario.....	11
6 Herramientas de Desarrollo.....	16
6.1 Entorno de Desarrollo.....	16
6.1.1 Visual Studio Code.....	16
6.1.2 NodeJS.....	16
7 Lenguajes de Programación.....	16
7.1 Javascript.....	16
7.2 Typescript.....	16

Introducción

El sistema de gestión para la biblioteca áulica se ha desarrollado para atender las necesidades específicas de bibliotecarios, maestros y alumnos. Este software permite una administración eficiente y organizada de los recursos bibliotecarios, facilitando el acceso y el uso de la biblioteca para toda la comunidad educativa.

Para los bibliotecarios, el sistema ofrece herramientas avanzadas para gestionar el inventario de libros, registrar préstamos y devoluciones, y mantener un control detallado de los usuarios y sus actividades. Los maestros tienen acceso a funcionalidades que les permiten supervisar los préstamos de sus alumnos, crear y compartir listas de lectura, y comunicarse directamente con el bibliotecario para solicitar nuevos materiales. Los alumnos, por su parte, pueden buscar y explorar libros de manera intuitiva, gestionar sus préstamos personales y recibir recomendaciones de lectura basadas en sus intereses y niveles de lectura.

El diseño del sistema incluye una interfaz amigable y accesible, asegurando que todos los usuarios puedan interactuar fácilmente con la biblioteca. Además, se han incorporado medidas de seguridad para proteger la información personal de los usuarios y garantizar la integridad del sistema.

En las siguientes secciones, se detallarán las opciones visuales y funcionalidades específicas para cada rol, así como las consideraciones técnicas que respaldan el diseño y la implementación de este sistema de gestión de la biblioteca áulica.

Objetivos

Facilitar el Acceso y la Gestión de Recursos Bibliográficos por parte de los Alumnos

- Ofrecer una interfaz intuitiva para que los alumnos busquen, reserven y gestionen el préstamo de libros.
- Permitir a los alumnos la visualización y renovación de sus préstamos.

Empoderar a los Docentes en la Supervisión y Gestión Bibliográfica

- Implementar una herramienta que permita a los docentes gestionar los alumnos de su clase.
- Implementar una herramienta que permita al docente gestionar los libros de su aula, haciendo seguimiento de cada libro.
- Facilitar a los docentes la creación y compartición de listas de lectura.
- Proveer herramientas para que los docentes supervisen y gestionen los préstamos de sus alumnos.
- Permitir a los docentes hacer recomendaciones de libros basadas en los niveles de lectura e intereses de los alumnos.

Proveer Control Total al Bibliotecario

- Otorgar al bibliotecario la capacidad de administrar todos los aspectos del sistema, incluyendo la gestión de libros, usuarios y permisos de acceso.
- Proveer herramientas para el monitoreo y control de préstamos pendientes y el historial de préstamos.
- Generar reportes y análisis sobre el uso de la biblioteca para una mejor toma de decisiones.

Asegurar la Seguridad y Privacidad

- Implementar medidas de seguridad para proteger la información personal de los usuarios.
- Garantizar la integridad del sistema mediante mecanismos de control de acceso y protección de datos.

Promover el Aprendizaje y la Lectura

- Facilitar el acceso a recursos educativos a través de la biblioteca.

- Proveer un entorno que favorezca el desarrollo de habilidades de lectura y aprendizaje continuo.
- Integrar la biblioteca como un recurso esencial en el proceso educativo del colegio.

Especificación de Requisitos

A continuación se describen las consideraciones iniciales para el análisis de requerimientos , su posterior diseño y desarrollo del sistema, teniendo en cuenta los requerimientos solicitados por el cliente.

3.1 Definición de Roles

En la gestión de una biblioteca áulica, es esencial definir claramente los roles y responsabilidades de cada usuario para asegurar un funcionamiento eficiente y organizado. Cada rol tiene un conjunto específico de permisos y funciones que se alinean con sus necesidades y objetivos dentro del sistema. A continuación, se describen los roles clave dentro del sistema:

- **Bibliotecario:** tendría el acceso más completo al sistema, con permisos de administración y gestión de toda la biblioteca.
- **Maestros:** tendrían acceso a funcionalidades relacionadas con la supervisión de los alumnos y la gestión de recursos para sus clases.
- **Alumnos:** tendrían acceso a funcionalidades básicas que faciliten la búsqueda y el préstamo de libros.

3.2 Requisitos Funcionales

Los requisitos funcionales definen las funciones y características que debe tener un sistema para cumplir con las necesidades del usuario. Estos requisitos describen las acciones específicas que el sistema debe ser capaz de realizar. A continuación se describen los requisitos funcionales del sistema teniendo en cuenta los distintos roles.

Alumnos:

- **Buscar Libros:** Los alumnos pueden buscar libros por título, autor o categoría.
- **Ver Disponibilidad:** Visualizar la disponibilidad de libros y su ubicación en la biblioteca.
- **Gestionar Préstamos:** Solicitar préstamos de libros y ver el historial de préstamos.
- **Recibir Recomendaciones:** Recibir recomendaciones de libros basadas en sus preferencias de

lectura.

Docentes:

- **Crear Listas de Lectura:** Crear listas de lectura para sus clases y compartirlas con los alumnos.
- **Gestionar Préstamos de Clase:** Supervisar los préstamos de libros realizados por sus alumnos.
- **Recomendar Libros:** Recomendar libros específicos a sus alumnos basándose en sus necesidades educativas.
- **Comunicarse con el Bibliotecario:** Enviar solicitudes de adquisición de libros y recibir notificaciones sobre el estado de los préstamos.

Bibliotecario:

- **Administrar Libros:** Agregar, editar y eliminar libros del catálogo de la biblioteca.
- **Administrar Usuarios:** Registrar y gestionar usuarios (alumnos, docentes, otros bibliotecarios).
- **Gestionar Préstamos:** Registrar préstamos, devoluciones y renovaciones de libros.
- **Generar Informes:** Generar informes sobre el uso de la biblioteca y las tendencias de préstamo.

3.3 Requisitos No Funcionales

Los requisitos no funcionales son criterios que describen cómo debe ser el sistema, en lugar de qué debe hacer. Estos requisitos se centran en atributos de calidad como la eficiencia, la usabilidad, la seguridad y la fiabilidad del sistema. A continuación se describen los requisitos no funcionales del sistema.

Interfaz de Usuario

- **Amigable y Accesible:** La interfaz debe ser fácil de usar para todos los usuarios, con un diseño intuitivo que refleje el entorno educativo.
- **Diseño Estético:** Utilizar colores pastel para la interfaz de administrador y colores diferentes para los usuarios "visitantes" para facilitar la diferenciación.
- **Tipografía:** Emplear una tipografía clara y legible que mejore la experiencia de lectura y uso.

Accesibilidad

- **Multiplataforma:** El sistema debe ser accesible desde dispositivos móviles y computadoras, ofreciendo una experiencia consistente en diferentes plataformas.

Seguridad y Privacidad

- **Protección de Datos:** Implementar medidas para proteger la información personal de los usuarios, como el cifrado de datos y políticas de privacidad estrictas.
- **Control de Acceso:** Garantizar que solo usuarios autorizados puedan acceder a funciones y datos específicos, mediante autenticación y autorización adecuadas.

Desempeño y Fiabilidad

- **Rendimiento:** El sistema debe ser rápido y eficiente, con tiempos de respuesta mínimos para las consultas y operaciones de los usuarios.
- **Disponibilidad:** Asegurar que el sistema esté disponible la mayor parte del tiempo, con un tiempo de inactividad mínimo.
- **Escalabilidad:** El sistema debe poder manejar un aumento en el número de usuarios y la cantidad de datos sin degradar el rendimiento.

Usabilidad

- **Tutoriales y Ayuda:** Incluir tutoriales y guías de uso para todos los usuarios, asegurando que puedan aprender a utilizar el sistema de manera rápida y efectiva.
- **ChatBot:** posibilidad de Implementar un ChatBot para asistencia en tiempo real, proporcionando respuestas rápidas a las consultas de los usuarios.

Integración

- **Sistemas Escolares:** Posibilidad de integrar el software con otros sistemas escolares existentes para sincronizar datos de alumnos y clases, facilitando la administración y evitando la duplicación de datos.

Mantenimiento y Soporte

- **Actualizaciones:** Proveer actualizaciones periódicas del sistema para mejorar funcionalidades y corregir errores.
- **Soporte Técnico:** Ofrecer soporte técnico para resolver problemas y responder preguntas de los usuarios.

Documentación

- **Manual de Usuario:** Proporcionar un manual de usuario detallado que cubra todas las funcionalidades del sistema.
- **Documentación Técnica:** Incluir documentación técnica para facilitar el mantenimiento y las futuras modificaciones del sistema.

Base de datos

4.1 Modelo conceptual: es una representación de alto nivel de la estructura y relaciones de los datos en una base de datos. Este modelo se centra en las entidades y sus interacciones.






De acuerdo a los requerimientos, podemos identificar las siguientes entidades:


- Roles
- Users
- Books
- Authors
- Book_Authors
- Loans

4.2. Modelado Lógico


4.2.1 Definición de Tablas

El diseño de la base de datos sigue el modelo relacional. Cada tabla está diseñada para almacenar información específica relacionada con las entidades.

Roles  		
Atributo 	Tipo 	Características 
role_id	int	campo unico ,no nulo y autoincremental
role_name	varchar(255)	campo no nulo

Users ▾ 		
Atributo ▾	Tipo ▾	Características ▾
user_id	int	campo unico ,no nulo y autoincremental
username	varchar(50)	campo no nulo
password_hash	varchar(255)	campo no nulo
email	varchar(100)	campo no nulo
created_at	datetime	campo no nulo
updated_at	datetime	campo no nulo
role_id	int	campo no nulo
id_moroso	boolean	campo no nulo

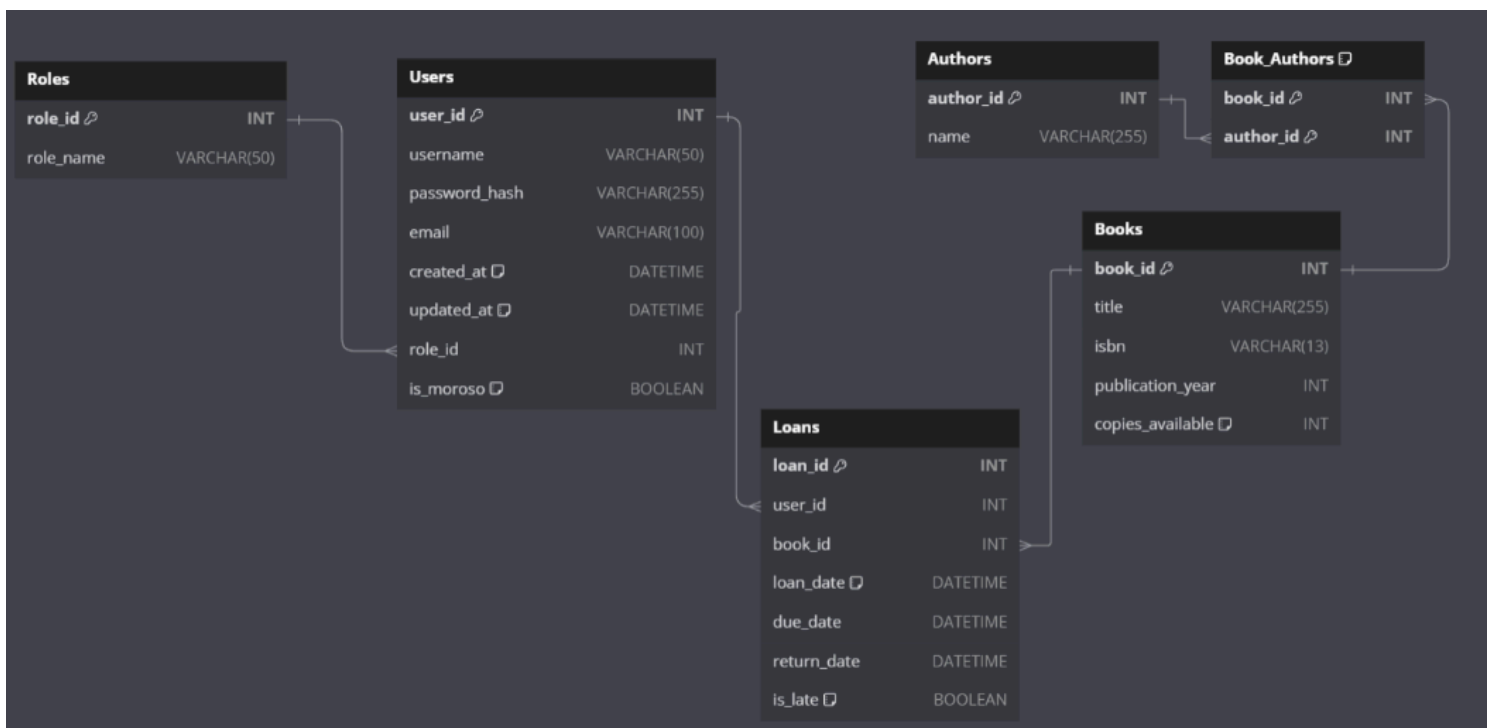
Books ▾ 		
Atributo ▾	Tipo ▾	Características ▾
book_id	int	campo unico ,no nulo y autoincremental
title	varchar_255	campo no nulo
isbn	varchar(13)	campo no nulo
publication_yea	int	campo no nulo
copies_availabl	int	campo no nulo

Authors ▾ 		
Atributo ▾	Tipo ▾	Características ▾
author_id	int	campo unico ,no nulo y autoincremental
name	varchar(255)	campo no nulo

Book_Authors		
Atributo	Tipo	Características
book_id	int	campo unico ,no nulo y autoincremental
author_id	int	campo no nulo

Loans		
Atributo	Tipo	Características
loan_id	int	campo unico ,no nulo y autoincremental
user_id	int	campo no nulo
book_id	int	campo no nulo
due_date	datetime	campo no nulo
return_date	datetime	campo no nulo
is_late	bool	campo no nulo

4.2.2 Modelo Entidad-Relación



Diseño

5.1 Definir los Prototipos de Interfaz: muestran qué información se muestra al usuario, el orden y las navegaciones que tiene el sistema.

5.2 Diseñar Interfaz pantalla principal: se tiene una interfaz de referencia para todas las vistas (Bibliotecario, maestros y alumnos)



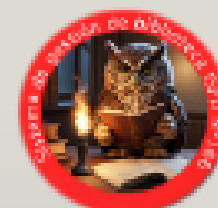
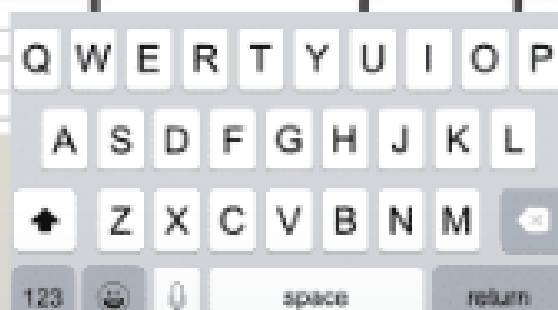
5.3 Diseñar Interfaz Usuario







Registro de Préstamo/Devolución

[illegible]

Préstamos pendientes

[illegible]

[illegible]

Herramientas de Desarrollo

6.1 Entorno de Desarrollo

6.1.1 Visual Studio Code

El sistema se desarrollará utilizando Visual Studio Code como entorno de desarrollo integrado (IDE) principal. Visual Studio Code proporcionará las herramientas necesarias para la escritura, depuración y gestión del código fuente del sistema.

6.1.2 NodeJS

Node.js es un entorno de ejecución de JavaScript construido sobre el motor V8 de Google Chrome. Permite a los desarrolladores utilizar JavaScript tanto en el lado del cliente como del servidor, unificando así el desarrollo web en un solo lenguaje de programación.

Node.js viene con npm (Node Package Manager), que es el administrador de paquetes de Node.js. npm es la mayor biblioteca de software del mundo y facilita la instalación, gestión y compartición de paquetes de código reutilizable.

Lenguajes de Programación

7.1 JavaScript

JavaScript es un lenguaje de programación utilizado en el desarrollo web para crear páginas web interactivas y dinámicas. Es un lenguaje de scripting del lado del cliente, lo que significa que se ejecuta en el navegador web del usuario en lugar de en el servidor. JavaScript se utiliza para agregar interactividad, validar formularios, crear efectos visuales y mucho más en una página web.

En nuestro proyecto, JavaScript se utilizará principalmente en el frontend para mejorar la experiencia del usuario y hacer que la interfaz de usuario sea más interactiva y dinámica.

7.2 TypeScript

TypeScript es un lenguaje de programación que se basa en JavaScript y agrega características como tipado estático, clases, interfaces y módulos. TypeScript se compila a JavaScript estándar, lo que significa que puede ser ejecutado en cualquier entorno que soporte JavaScript.

En nuestro proyecto, TypeScript será utilizado en varias áreas tanto del frontend como del backend para mejorar la calidad del código y la productividad de los desarrolladores.

Frontend

En el frontend, TypeScript se utilizará principalmente en el desarrollo de la lógica de la aplicación en el lado del cliente y la construcción de componentes de la interfaz de usuario. Esto incluirá la validación de formularios, la gestión de eventos y la interacción con APIs RESTful para obtener y enviar datos.

Backend

En el backend, TypeScript será utilizado en el desarrollo de la lógica de negocio, la interacción con la base de datos y la creación de APIs RESTful. Se utilizará en conjunto con frameworks como Express.js para el desarrollo de aplicaciones web en Node.js, aprovechando sus características como el tipado estático y la programación orientada a objetos.