

**INSTITUTO TECNOLÓGICO SUPERIOR**

**DE JESÚS CARRANZA**

**INFORME FINAL DE RESIDENCIA PROFESIONAL**

**ÁMBITO: TECNOLÓGICO**

**SOFTWARE ADMINISTRATIVO DE BIBLIOTECA PARA LA ESCUELA DE BACHILLERES “24 DE FEBRERO”**

**BACHILLERATO “24 DE FEBRERO”**

**PRESENTA: C. ARTURO SALAS HERNÁNDEZ**

**19180054**

**INGENIERÍA EN SISTEMAS COMPUTACIONALES**

**DE LA ESPECIALIDAD INGENIERÍA DE SOFTWARE**

**ISC -2010-224**

**ASESOR INTERNO: ING. AMADO LÓPEZ HILARIO**

**ASESOR EXTERNO: BIOL. YATZAMIL ROMÁN ÁNGELES**

**JESÚS CARRANZA, VER., DICIEMBRE DE 2023.**

**INFORME FINAL DE RESIDENCIA PROFESIONAL**

CAPITULO I PRELIMINARES

## Agradecimientos

Quisiera aprovechar esta oportunidad para expresar mi gratitud a todas las personas que hicieron posible la realización de este proyecto. Sin su apoyo, dedicación y contribuciones, este trabajo no habría sido posible.

En primer lugar, quiero expresar mi profundo agradecimiento a mi familia. Su amor, paciencia y apoyo incondicional fueron la base que me permitió desarrollar este proyecto y llevarlo a buenos términos.

También quiero agradecer de manera especial a mis asesores, el BIOL. Yatzamil Román Ángeles por su tiempo y su disposición de ayudar en todo momento pero sobre todo por haberme permitido realizar este proyecto en el instituto a su cargo. Por otra parte al ISC. Amado López Hilario por su orientación brindada y por sus valiosos conocimientos los cuales fueron invaluables y fundamentales para establecer las bases de este presente informe.

Finalmente agradezco profundamente a cada uno de mis docentes, gracias por su tiempo, su dedicación y por cada una de sus enseñanzas brindadas en el aula las cuales han sido parte esencial en desarrollo de mi formación profesional.

A todos y cada uno de ustedes, les agradezco de corazón su contribución y apoyo. Su participación ha sido invaluable y ha dejado una huella duradera en esta investigación.

¡Muchas gracias!

## Resumen

El presente proyecto tiene como objetivo principal la creación de un sistema administrativo de biblioteca para la escuela de bachilleres “24 de febrero”. El propósito de este sistema es optimizar la gestión y el acceso a los recursos bibliográficos disponibles, brindando a estudiantes y personal administrativo una herramienta eficiente y moderna para el control del inventario de los libros.

Para este proyecto se llevó a cabo un análisis exhaustivo de las necesidades y/o requerimientos de la biblioteca escolar. A partir de la información recabada se diseñó y desarrolló un sistema personalizado que se adapta a las necesidades específicas de dicha institución educativa.

El sistema administrativo de biblioteca creado ofrece diversas funcionalidades, incluyendo el registro y la catalogación de libros, la gestión de préstamos y devoluciones, todo esto utilizando las medidas de seguridad pertinentes para garantizar la integridad de los datos.

Además, durante el desarrollo del proyecto se llevaron a cabo pruebas exhaustivas para asegurar su funcionalidad, usabilidad y rendimiento de dicho software.

**INDICE GENERAL**

[CAPITULO I PRELIMINARES ii](#_Toc151817131)

[Agradecimientos ii](#_Toc151817132)

[Resumen iii](#_Toc151817133)

[CAPITULO II GENERALIDADES 6](#_Toc151817134)

[2.1 INTRODUCCIÓN 6](#_Toc151817135)

[2.2 DESCRIPCIÓN DE LA EMPRESA 7](#_Toc151817136)

[2.3 PROBLEMA A RESOLVER 9](#_Toc151817137)

[2.4 OBJETIVO GENERAL Y ESPECIFICO 10](#_Toc151817138)

[2.4.1 OBJETIVO GENERAL 10](#_Toc151817139)

[2.4.2 OBJETIVOS ESPECÍFICOS 10](#_Toc151817140)

[2.5 JUSTIFICACIÓN 11](#_Toc151817141)

[CAPITULO III MARCO TEÓRICO 12](#_Toc151817142)

[3.1 FUNDAMENTO TEÓRICO 12](#_Toc151817144)

[3.1.1 DISEÑO WEB 12](#_Toc151817145)

[3.1.2 UML (UNIFIED MODELING LANGUAGE) 15](#_Toc151817146)

[3.1.3 TECNOLOGÍAS FRONTEND 16](#_Toc151817147)

[3.1.4 TECNOLOGÍAS BACKEND 24](#_Toc151817148)

[3.1.5 BASES DE DATOS 26](#_Toc151817149)

[3.1.6 MOTOR DE BASE DE DATOS 28](#_Toc151817150)

[3.1.7 SISTEMA GESTOR DE BASES DE DATOS (SGBD) 29](#_Toc151817151)

[3.1.8 PROGRAMAS ADICIONALES 30](#_Toc151817152)

[3.2 MARCO METODOLÓGICO 33](#_Toc151817156)

[3.2.1 MODELOS DE PROCESO PRESCRIPTIVO 34](#_Toc151817157)

[3.2.2 METODOLOGÍA A EMPLEAR 35](#_Toc151817158)

[3.3 MARCO LEGAL 38](#_Toc151817163)

[3.3.1 LEY FEDERAL DEL DERECHO DE AUTOR 38](#_Toc151817164)

[3.4 ANTECEDENTES 43](#_Toc151817170)

[CAPÍTULO IV DESARROLLO 44](#_Toc151817171)

[4.1 PROCEDIMIENTO Y DESCRIPCIÓN DE LAS ACTIVIDADES REALIZADAS 44](#_Toc151817172)

[CAPÍTULO V RESULTADOS 46](#_Toc151817173)

[5.1 RESULTADOS, PLANOS, GRAFICAS, PROTOTIPOS, MANUALES, ETC. 46](#_Toc151817174)

[CAPÍTULO VI CONCLUSIONES Y RECOMENDACIONES 49](#_Toc151817175)

[6.1 CONCLUSIONES 49](#_Toc151817176)

[6.2 RECOMENDACIONES 50](#_Toc151817177)

[CAPÍTULO VII COMPETENCIAS DESARROLLADAS 51](#_Toc151817178)

[7.1 COMPETENCIAS DESARROLLADAS Y/O APLICADAS 51](#_Toc151817179)

[FUENTES DE INFORMACIÓN 52](#_Toc151817180)

[ANEXOS 55](#_Toc151817181)

**ÍNDICE DE ILUSTRACIONES**

[Ilustración 1 Organigrama de la empresa 8](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817063)

[Ilustración 2 Ciclo evolutivo del software 33](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817064)

[Ilustración 3 Representación de la metodología en cascada 36](#_Toc151817065)

[Ilustración 4 Diseño de la base de datos 57](#_Toc151817066)

[Ilustración 5 Diagrama Entidad-Relación 63](#_Toc151817067)

[Ilustración 6 Pseudocódigo inicio sesión 64](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817068)

[Ilustración 7 Pseudocódigo visualización módulo préstamos 64](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817069)

[Ilustración 8 Pseudocódigo registro de préstamo 65](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817070)

[Ilustración 9 Pseudocódigo actualizar estado préstamo 65](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817071)

[Ilustración 10 Pseudocódigo eliminar registro préstamo 66](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817072)

[Ilustración 11 Pseudocódigo visualización módulo libros 66](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817073)

[Ilustración 12 Pseudocódigo registro de libro 67](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817074)

[Ilustración 13 Pseudocódigo actualizar datos libro 68](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817075)

[Ilustración 14 Pseudocódigo eliminar registro libro 68](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817076)

[Ilustración 15 Pseudocódigo visualización módulo alumnos 69](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817077)

[Ilustración 16 Pseudocódigo registro de alumno 69](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817078)

[Ilustración 17 Pseudocódigo actualizar datos alumno 70](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817079)

[Ilustración 18 Pseudocódigo eliminar registro alumno 70](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817080)

[Ilustración 19 Diagrama de flujo 1 71](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817081)

[Ilustración 20 Diagrama de flujo 2 71](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817082)

[Ilustración 21 Diagrama de flujo 3 72](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817083)

[Ilustración 22 Diagrama de flujo 4 72](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817084)

[Ilustración 23 Diagrama de flujo 5 73](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817085)

[Ilustración 24 Diagrama de flujo 6 73](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817086)

[Ilustración 25 Diagrama de flujo 7 74](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817087)

[Ilustración 26 Diagrama de flujo 8 74](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817088)

[Ilustración 27 Diagrama de flujo 9 75](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817089)

[Ilustración 28 Diagrama de flujo 10 75](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817090)

[Ilustración 29 Diagrama de flujo 11 76](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817091)

[Ilustración 30 Diagrama de flujo 12 76](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817092)

[Ilustración 31 Diagrama de flujo 13 77](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817093)

[Ilustración 32 Interfaz inicio de sesión 78](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817094)

[Ilustración 33 Interfaz principal 78](#_Toc151817095)

[Ilustración 34 Interfaz del módulo préstamos 79](#_Toc151817096)

[Ilustración 35Interfaz del módulo libros 79](#_Toc151817097)

[Ilustración 36 Diseño modular 80](#_Toc151817098)

[Ilustración 37 Diagrama de casos de uso 80](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817099)

[Ilustración 38 Diagrama de secuencia 81](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817100)

[Ilustración 39 Error 1 barra lateral 82](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817101)

[Ilustración 40 Corrección 1 barra lateral 82](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817102)

[Ilustración 41 Error 2 interfaz prestamos 83](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817103)

[Ilustración 42 Corrección 2 interfaz prestamos 83](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817104)

[Ilustración 43 Error 3 módulo usuarios 84](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817105)

[Ilustración 44 Corrección 3 módulo usuarios 84](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817106)

[Ilustración 45 Error 4 módulo usuarios 85](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817107)

[Ilustración 46 Corrección 4 módulo usuarios 85](file:///C:\xampp\htdocs\informe-final\INFORME%20FINAL%20RP-23.docx#_Toc151817108)

ÍNDICE DE TABLAS

CAPITULO II GENERALIDADES

## INTRODUCCIÓN

En los últimos años, la tecnología ha emergido como un factor transformador en múltiples industrias, impactando de manera significativa debido a su versatilidad y aplicabilidad. Hoy en día, se ha convertido en un componente imprescindible para empresas de todos los tamaños, desde las micro hasta las macroempresas.

La adopción de sistemas digitales se ha vuelto fundamental para una gestión empresarial eficaz, permitiendo el aprovechamiento óptimo de recursos y la ejecución de tareas de manera eficiente.

El uso del software ofrece mayores ventajas en comparación con los métodos tradicionales de administración de información (libros de registro). La portabilidad, escalabilidad y capacidad de personalización de estos sistemas los posicionan como herramientas superiores. Además, su agilidad para responder a múltiples solicitudes representa una mejora significativa en los tiempos de respuesta, aspecto que es clave en un entorno que demanda respuestas precisas e inmediatas.

Los sistemas digitales nos permiten gestionar grandes cantidades de datos de forma práctica, esto no solo permite la manipulación eficiente de información masiva, sino que también permite a las empresas satisfacer rápidamente las demandas de sus usuarios. Sin esta infraestructura digital, sería prácticamente imposible para las empresas gestionar y procesar la enorme cantidad de información que estas almacenan.

En conclusión, la implementación de sistemas digitales no solo implica una modernización de procesos, sino una transformación integral en la mentalidad empresarial. Las herramientas digitales se han convertido en aliados estratégicos que permiten no solo una gestión más eficiente, sino también la generación de ventajas competitivas sostenibles.

## DESCRIPCIÓN DE LA EMPRESA

**Empresa:** Telebachillerato “24 de febrero”.

**Dirección:** Congregación Veinticuatro de febrero, Jesús Carranza, Ver., México. CP. 96950.

**Teléfono:** 924 244 5081.

**Dirección de correo electrónico:** bachillerato24@gmail.com.

**Misión:** Nuestra misión es proporcionar a los estudiantes una educación integral y de calidad, que los prepare para enfrentar los desafíos del futuro y les permita alcanzar sus metas personales y profesionales. Nos comprometemos a brindar una educación basada en valores, con un enfoque en el desarrollo de habilidades y conocimientos prácticos que los estudiantes puedan aplicar en su vida diaria.

**Visión:** Nuestra visión es ser reconocidos como una escuela líder en la formación de líderes y ciudadanos comprometidos, capaces de enfrentar los desafíos del mundo globalizado de hoy. Nos esforzamos por ser una institución educativa que fomente el desarrollo integral de nuestros estudiantes, y que promueva la innovación y la excelencia académica. Trabajamos con la comunidad educativa y nuestros socios para construir un ambiente de aprendizaje colaborativo, inclusivo y respetuoso, en el que nuestros estudiantes puedan alcanzar su máximo potencial.

**Estructura organizacional:** El telebachillerato “24 de febrero” actualmente se encuentra a cargo del Biol. Yatzamil Román Ángeles quien funge con el cargo de director general.

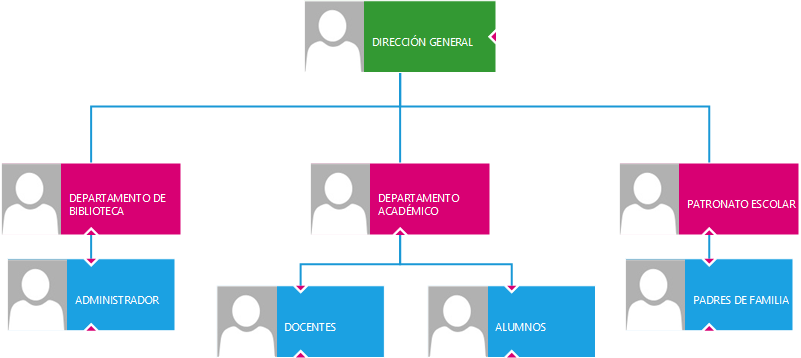


Ilustración 1 Organigrama de la empresa

**Nombre del departamento:** Departamento de biblioteca.

## PROBLEMA A RESOLVER

Recientemente la escuela de bachilleres “24 de febrero” ubicada en el municipio de Jesús Carranza, Ver., inició con la creación de una biblioteca escolar a la cual los alumnos pueden acudir para solicitar libros prestados.

Al implementar un sistema digital, los administradores podrán acceder a un catálogo para consultar la disponibilidad de los libros en tiempo real. Esta función eliminaría la incertidumbre al permitir verificar la existencia y ubicación de los libros deseados de manera instantánea, optimizando así el proceso de solicitud. Además, la digitalización de los registros de préstamos y devoluciones garantizaría la integridad y seguridad de la información. Al almacenar estos datos en una plataforma digital segura, se reduciría significativamente el riesgo de pérdida o daño de registros debido a desastres naturales, extravíos o deterioro físico, lo que facilitaría la recuperación y consulta de información de manera rápida y eficiente en cualquier momento.

Este sistema ofrece una interfaz intuitiva y de fácil acceso para el personal de la biblioteca, lo que permite una administración más eficiente de los préstamos, seguimiento y devoluciones.

## OBJETIVO GENERAL Y ESPECIFICO

### 2.4.1 OBJETIVO GENERAL

Desarrollar e implementar un sistema de administración de biblioteca para la escuela de nivel medio superior “24 de febrero” ubicada en la congregación 24 de febrero del municipio de Jesús Carranza, Ver., con la finalidad de tener un mejor control sobre los préstamos de libros realizados por la institución.

### 2.4.2 OBJETIVOS ESPECÍFICOS

* Levantamiento de los requerimientos.
* Hacer uso de un SGBD para la administración de base de datos.
* Usar una base de datos relacional (MariaDB)
* Usar la arquitectura cliente-servidor.
* Diseñar la parte del Backend (servidor) usando los lenguajes PHP y SQL.
* Diseño de interfaces de usuario amigables usando el Framework Bootstrap 5.
* Realizar pruebas y corrección de errores del sistema.
* Implementar el sistema.

## JUSTIFICACIÓN

Actualmente la escuela de bachilleres “24 de febrero” cuenta con una biblioteca de tamaño pequeño-mediano, lo cual es preocupante para la administración a largo plazo. La ausencia de un sistema informático que permita un control efectivo de los libros genera una carga constante y repetitiva en el trabajo de los administradores. Este ciclo manual no solo es propenso a errores, sino que también plantea el riesgo de pérdida de material del instituto.

La implementación de un software administrativo de biblioteca surge como una solución imprescindible ante esta problemática. Este sistema se convierte en una herramienta fundamental para optimizar la gestión de libros, eliminando la carga manual y reduciendo significativamente los errores que podrían conllevar a pérdidas materiales.

La importancia y necesidad de este sistema se basa en los siguientes aspectos:

* **Agilización de la administración:** Un software especializado agiliza significativamente los procesos administrativos de la biblioteca, desde el registro de préstamos hasta la gestión de devoluciones, facilitando un flujo de trabajo más eficiente y reduciendo la carga operativa del personal.
* **Acceso instantáneo a la información:** La disponibilidad de datos en tiempo real permite a los administradores acceder al catálogo de la biblioteca en cualquier momento, proporcionando una visión clara sobre la disponibilidad de los libros, su ubicación y su estado.
* **Manejo apropiado de la información:** La digitalización de registros y la estructuración de una base de datos ordenada y centralizada garantizan un manejo apropiado de la información, esto facilita la búsqueda y recuperación de datos evitando así perdida de registros importantes.
* **Seguridad de los datos:** La seguridad de los datos es crucial, es por ello que utilizando un sistema digital se proporciona un entorno seguro para almacenar y gestionar información, minimizando el riesgo de pérdida o daño de registros debido a desastres naturales o errores humanos.

CAPITULO III MARCO TEÓRICO



## FUNDAMENTO TEÓRICO

### 3.1.1 DISEÑO WEB

Como usuarios de internet, exploramos numerosos sitios web a diario con la finalidad de revisar correos, mantenernos informados, compartir imágenes, realizar compras, relatar nuestras actividades o, simplemente, disfrutar. Nos hemos habituado a navegar por estos sitios de manera casi automática, utilizando interfaces que, en general, resultan accesibles y fáciles de manejar sin necesitar un gran esfuerzo o atención.

La composición y diseño de las páginas web han experimentado una evolución a lo largo del tiempo. En sus inicios, surgieron empleando metáforas provenientes del cine, la televisión, los libros o las galerías de arte, al igual que los sistemas operativos previamente se basaron en la metáfora del escritorio. Estas analogías ayudaron a los usuarios a comprender las funciones de estos productos interactivos, marcando así el inicio de una disciplina que, con el tiempo, se transformó en un campo de trabajo interdisciplinario, activo y creativo (Carballeiro, 2012).

**Análisis y diseño de sistemas**

Dentro de las empresas, el análisis y diseño de sistemas implica evaluar la situación actual de la organización con el fin de mejorarla mediante métodos y procesos más apropiados.

El desarrollo de sistemas consta principalmente de dos partes: análisis y diseño. El diseño implica la planificación, sustitución o complemento de un sistema existente. Antes de iniciar este proceso, es crucial comprender a fondo el sistema actual y determinar cómo se pueden utilizar las computadoras para mejorar la eficiencia operativa, si es viable. Por otro lado, el análisis de sistemas implica la identificación e interpretación de información, el diagnóstico de problemas y el uso de esos datos para sugerir mejoras al sistema. Esta tarea recae en el analista de sistemas. (Senn, 2001).

**Características de diseño**

Las representaciones visuales tienen un estilo específico con el propósito de facilitar la comprensión de los temas por parte de los estudiantes. Se emplean esquemas conceptuales para mostrar las distintas herramientas disponibles para los analistas de sistemas. Este ejemplo específico ejemplifica las diferencias entre los diagramas de flujo de datos lógicos y físicos. Los esquemas conceptuales se presentan utilizando colores que identifican claramente sus funciones, permitiendo a los estudiantes reconocer fácilmente sus elementos. Se incluyen numerosas herramientas esenciales como los diagramas de caso de uso, secuencia y clases, entre otros.

Las pantallas de computadora revelan aspectos clave del software que resultan valiosos para el analista. Por ejemplo, muestran cómo evaluar los enlaces rotos en un sitio web utilizando herramientas como Microsoft Visio. Estas pantallas resaltan elementos importantes del diseño, y los analistas están constantemente enfocados en mejorar la apariencia de las interfaces y páginas web que desarrollan. Los ejemplos coloridos son útiles para demostrar por qué ciertos diseños de pantalla son particularmente efectivos.

Los formularios en papel se emplean para representar el diseño de las entradas y salidas, así como para estructurar cuestionarios. La tinta azul es la elección predominante para escribir o ingresar datos, facilitando la identificación de la información completada por los usuarios. Aunque muchas organizaciones tienen como meta la automatización de procesos manuales, aún se utiliza ampliamente formularios en papel para la captura de datos. La mejora en el diseño de estos formularios permite a los analistas asegurar la precisión y la integridad de los datos de entrada y salida, además de optimizar los nuevos flujos de trabajo que surgen debido a la automatización de aplicaciones de negocio a consumidor (B2C) para el comercio electrónico en línea.

Las tablas se emplean cuando se requiere destacar una lista importante o para organizar y clasificar información. También complementan la comprensión del lector de una manera diferente a la presentación en el cuerpo del texto. La mayoría de los analistas encuentran que las tablas son una herramienta útil para organizar números y texto, ofreciendo una vista "instantánea" y significativa de la información (KENDALL & KENDALL, 2011).

**Algoritmo**

Un algoritmo consiste en una serie de pasos organizados y secuenciales diseñados para alcanzar un objetivo específico. Estos pasos deben ejecutarse en secuencia, uno después del otro, y siguiendo un orden determinado, que en la mayoría de los casos es esencial para su correcta ejecución. En resumen, un algoritmo es una herramienta para alcanzar un objetivo mediante pasos organizados (Buriticá, 1999).

**Algoritmo computacional**

Los algoritmos computacionales son aquellos diseñados preferentemente para ser implementados en computadoras, aprovechando la velocidad de procesamiento que ofrecen estos dispositivos. Estos algoritmos están orientados a ser ejecutados en entornos computacionales para mejorar la eficiencia y rapidez en el logro de los objetivos (Buriticá, 1999).

**Lenguajes de programación**

Cuando un algoritmo se ejecuta en un procesador de computadora, se debe expresar en un formato llamado programa, ya que la computadora no comprende directamente pseudocódigos o diagramas de flujo, aunque estos puedan ser entendidos por cualquier programador. Un programa se escribe en un lenguaje de programación y el proceso de transformar un algoritmo en un programa se conoce como programación. Los lenguajes empleados para escribir estos programas son los lenguajes de programación y aquellos que los crean son los programadores, los diseñadores de software.

El paso de convertir un algoritmo desde pseudocódigo a un lenguaje de programación se llama codificación, y el resultado, el algoritmo expresado en un lenguaje de programación, es conocido como código fuente. Sin embargo, las computadoras no entienden directamente los lenguajes de programación, por lo que se requiere un programa adicional que traduzca el código fuente a un lenguaje comprensible por la máquina, llamado lenguaje máquina, pero extremadamente complejo para las personas. Este código correspondiente es conocido como código máquina.

Los programas encargados de realizar esta traducción del código fuente, como en el caso de C++, al código máquina, son denominados traductores (Aguilar, 2008).

### 3.1.2 UML (UNIFIED MODELING LANGUAGE)

El Lenguaje Unificado de Modelado (UML) se considera un lenguaje, al igual que Pascal, C# (C sharp), el alemán, el inglés y el latín; y posiblemente sea uno de los lenguajes más nuevos creados por la humanidad, surgiendo alrededor de 1997.

Al igual que otros lenguajes, UML nació por necesidad. Emplea símbolos para comunicar significados, pero a diferencia de los idiomas naturales como el inglés o el alemán, que evolucionan a lo largo del tiempo mediante el uso común y la adaptación, UML fue creado por científicos. Esta particularidad presenta una dificultad: los científicos, aunque muy inteligentes, a menudo enfrentan problemas al explicar conceptos en su campo (KIMMEL, 2008).

**Modelos**

Los modelos se componen de representaciones gráficas o diagramas. Su propósito es ser más económicos de producir y probar en comparación con la escritura de código. No obstante, si se invierte mucho tiempo en la planificación de los modelos, en decidir cuándo pasar de los modelos a la codificación o en buscar la perfección en ellos, se notará gradualmente una disminución en su valor y en la eficiencia temporal.

Aunque se puede describir un sistema con texto simple, las imágenes transmiten una cantidad mayor de información. La metodología de eXtreme Programming (XP) aboga por codificar y ajustar conforme se avanza, pero la complejidad de los detalles en el código supera a la de las imágenes. Los programadores se aferran al código y no tanto a las representaciones visuales. Una vez que el código está escrito, modificarlo puede resultar difícil de aceptar para el programador o los administradores, especialmente si se considera que el código ya funciona. Por otro lado, las personas suelen trabajar más cómodamente con modelos de manera informal y están abiertas a sugerencias.

Además, debido a la simplicidad de los símbolos utilizados en los modelos, más personas pueden participar en el diseño del sistema. Mostrar cien líneas de código a un usuario final probablemente genere confusión, pero presentarle un diagrama de actividades facilita que esa persona evalúe si se ha captado correctamente el proceso que se pretende describir. Este enfoque se centra en los modelos dentro del contexto del UML (KIMMEL, 2008).

### 3.1.3 TECNOLOGÍAS FRONTEND

**HTML (HYPERTEXT MARKUP LANGUAGE)**

HTML (Lenguaje de Marcas de Hipertexto, del inglés HyperText Markup Language) es el componente más básico de la Web. Define el significado y la estructura del contenido web. Además de HTML, generalmente se utilizan otras tecnologías para describir la apariencia/presentación de una página web (CSS) o la funcionalidad/comportamiento (JavaScript) (Foundation M. , HTML: Lenguaje de etiquetas de hipertexto - MDN Web Docs, 2023).

"Hipertexto" hace referencia a los vínculos que conectan distintas páginas web, ya sea dentro de un mismo sitio o entre sitios web. Estos enlaces representan un elemento esencial en la estructura de la Web. Al compartir contenido en Internet y asociarlo a las páginas creadas por otros usuarios, te conviertes en un partícipe activo de la "World Wide Web" (Red Informática Mundial).

El lenguaje HTML utiliza "marcas" para etiquetar elementos como texto, imágenes y otros contenidos, permitiendo su visualización en un navegador web. Entre las marcas HTML se incluyen elementos especiales como <head>, <title>, <body>, <header>, <footer>, <article>, <section>, <p>, <div>, <span>, <img>, <aside>, <audio>, <canvas>, <datalist>, <details>, <embed>, <nav>, <output>, <progress>, <video>, <ul>, <ol>, <li> y una amplia variedad más.

Cada elemento HTML se diferencia en un documento mediante "etiquetas", las cuales consisten en el nombre del elemento encerrado entre "<" y ">". Este nombre, dentro de la etiqueta, no distingue entre mayúsculas y minúsculas, permitiendo su escritura en mayúsculas, minúsculas o una combinación de ambas. Por ejemplo, la etiqueta <title> puede escribirse como <Title>, <TITLE> o de cualquier otra manera. (Foundation M. , HTML: Lenguaje de etiquetas de hipertexto - MDN Web Docs, 2023).

Toda página web está conformada por un código estructurado, denominado HTML (HyperText Markup Language). Básicamente, este cuenta con una serie de instrucciones para indicarle al navegador que estemos utilizando la manera en que la página debe ser visualizada y representada (Carballeiro, 2012).

Una de las cualidades distintivas de este lenguaje radica en su simplicidad, ya que para generar un documento HTML, solo se requiere un editor de texto básico como WordPad o el Bloc de notas de Windows. Estas simples herramientas permiten la creación de una página web mediante la inclusión de etiquetas específicas del lenguaje. Estas etiquetas señalan tanto el inicio como el final de los componentes del documento, desde las propiedades del texto hasta elementos multimedia como imágenes, videos y audio.

A pesar de ser considerado un lenguaje, HTML no se clasifica como un lenguaje de programación; más bien, se trata de un lenguaje de marcado. Consiste en un conjunto de etiquetas que instruyen al navegador sobre cómo presentar ciertos elementos, como destacar un encabezado, especificar el color del texto o dónde mostrar una imagen. En caso de que el navegador detecte algún error en el código, mostrará el documento interpretándolo de la mejor manera posible. (Carballeiro, 2012).

Los componentes HTML forman una estructura organizada tipo árbol con el elemento <html> como su punto de origen. Esta disposición representa múltiples niveles de jerarquía, donde ciertos elementos definen secciones generales del documento mientras que otros representan secciones más específicas o el contenido mismo. A continuación se enumeran los elementos disponibles para establecer la base estructural y proporcionar la información necesaria al navegador para visualizar la página en la pantalla.

<html>: Este elemento encierra el código HTML y puede incorporar el atributo 'Lang' para especificar el idioma del contenido del documento.

<head>: Utilizado para definir la información esencial para la configuración de la página web, como el título, la codificación de caracteres y los archivos externos necesarios para el documento.

<body>: Delimita el contenido visible del documento, es decir, la parte visible de la página web. (Gauchad, 2017).

**Archivos**

Exactamente, cuando hablamos de sitios web, cada página individual es un documento que un navegador web descarga y muestra al usuario cuando este lo solicita. Cada uno de estos documentos se llama página web. El proceso de pasar de una página a otra dentro de un sitio web se conoce comúnmente como navegar o navegación (el usuario "navega" a través de las páginas del sitio).

Para desarrollar un sitio web, se requiere crear un archivo separado para cada página que se desea incluir en el sitio. Estos archivos suelen estar escritos en lenguajes como HTML, CSS y JavaScript. Además de los archivos de las páginas, también se deben incluir recursos adicionales como imágenes, videos, hojas de estilo (CSS), archivos de scripts (JavaScript) u otros elementos multimedia que se deseen mostrar o utilizar en estas páginas.

El conjunto de estos archivos, junto con su estructura de carpetas y subcarpetas, conforma la estructura del sitio web y permite a los navegadores web cargar y mostrar correctamente el contenido al usuario cuando accede al sitio (Gauchad, 2017).

**CSS3 (CASCADING STYLE SHEETS)**

Hojas de Estilo en Cascada (del inglés Cascading Style Sheets) o CSS es el lenguaje de estilos utilizado para describir la presentación de documentos HTML o XML (en-US) (incluyendo varios lenguajes basados en XML como SVG, MathML o XHTML). CSS describe como debe ser renderizado el elemento estructurado en la pantalla, en papel, en el habla o en otros medios.

CSS forma parte de los lenguajes fundamentales de la Open Web y cuenta con una especificación estandarizada por el W3C. En el pasado, el desarrollo de distintas partes de las especificaciones de CSS se llevaba a cabo de manera sincrónica, permitiendo la creación de versiones para sus recomendaciones, como CSS1, CSS2.1 y CSS3. Sin embargo, CSS4 nunca se ha lanzado como una versión oficial.

A partir de CSS3, se amplió significativamente el alcance de las especificaciones y el avance de los distintos módulos de CSS empezó a mostrar notables diferencias. Esto llevó a un enfoque más efectivo de desarrollar y publicar recomendaciones separadas por módulos. En lugar de asignar versiones a las especificaciones de CSS, en la actualidad, el W3C captura las últimas especificaciones estables de CSS (Foundation M. , CSS - MDN Web Docs, 1998-2023).

CSS se ha convertido en el estándar predefinido para la presentación de páginas creadas en HTML5, gracias a su capacidad para mejorar el diseño y crear una estética más atractiva. Reconocido como una herramienta de diseño gráfico, permite definir la apariencia visual de los sitios web, siendo adaptable a una amplia gama de dispositivos, desde pantallas grandes hasta dispositivos móviles y tablets.

Su independencia de HTML y su compatibilidad con varios lenguajes basados en XML amplían su utilidad. La separación de HTML y CSS trae consigo beneficios significativos, como la simplificación del mantenimiento de sitios, la facilidad para intercambiar hojas de estilo entre diferentes páginas y la capacidad para personalizar la presentación en distintos entornos de visualización. (Villa, 2019).

En CSS, los estilos personalizados se establecen mediante propiedades. Cada estilo se define al declarar el nombre de la propiedad y su respectivo valor, separados por dos puntos. Por ejemplo, **font-size: 24px;** establece una propiedad que cambia el tamaño de la fuente a 24 píxeles (ya que algunas propiedades pueden contener múltiples valores separados por espacios es por ello que se finaliza la línea con un punto y coma).

La mayoría de las propiedades CSS se utilizan para modificar un solo aspecto de un elemento, como el tamaño de la fuente en este caso. Sin embargo, si se desea cambiar varios estilos simultáneamente, es necesario declarar múltiples propiedades. CSS ofrece una sintaxis que simplifica este proceso al asignar múltiples propiedades a un elemento. Esta estructura se conoce como 'regla'. Una regla consiste en una serie de propiedades delimitadas por llaves y asociadas a un selector. El selector especifica qué elementos serán afectados por las propiedades definidas. (Gauchad, 2017).

Las propiedades y reglas en CSS determinan los estilos que deseamos asignar a uno o varios elementos, pero estos estilos no surten efecto hasta que los incorporamos en el documento. Existen tres métodos disponibles para este propósito: la utilización de estilos en línea, estilos incrustados o el uso de hojas de estilos externas (Gauchad, 2017).

**JAVASCRIPT**

JavaScript (JS) es un lenguaje de programación ligero, interpretado, o compilado justo-a-tiempo (just-in-time) con funciones de primera clase. Si bien es más conocido como un lenguaje de scripting (secuencias de comandos) para páginas web, y es usado en muchos entornos fuera del navegador, tal como Node.js, Apache CouchDB y Adobe Acrobat JavaScript es un lenguaje de programación basada en prototipos, multiparadigma, de un solo hilo, dinámico, con soporte para programación orientada a objetos, imperativa y declarativa (por ejemplo programación funcional). Lee más en acerca de JavaScript (Foundation M. , JavaScript - MDN Web Docs - Mozilla, 1998-2023).

JavaScript es el lenguaje de programación empleado en el desarrollo de aplicaciones web del lado del cliente. Su origen se remonta a 1995, cuando Netscape Corporation lo introdujo como un lenguaje de script en la primera versión de su cliente de WWW. Paralelamente, Microsoft, al crear su propio cliente web, Internet Explorer, adoptó el lenguaje de Netscape pero bajo el nombre de jScript. A pesar de ser muy similares, presentaban diferencias sutiles.

Desde su inicio, surgieron discrepancias en su uso, especialmente en la interacción con el DOM (modelo de objetos del documento), en el sistema de eventos y en varias otras particularidades que los distinguían. Esto generó un escenario en el que el lenguaje debía interactuar con modelos de clases y sistemas de eventos diferentes.

En sus inicios, la programación del lado del cliente era considerablemente compleja, ya que era necesario adaptarse a las especificaciones de cada navegador. Esto ocasionaba que el código resultara poco sólido y difícil de mantener, a menudo con bifurcaciones para cada especificación de navegador en pequeñas funciones.

Para abordar estos problemas de compatibilidad entre el lenguaje y los navegadores, surgieron bibliotecas cuyo propósito era establecer una API común que fuera compatible con distintos navegadores. (Puig, 2018).

**FRAMEWORK**

El término framework, hace referencia a una estructura de software compuesta por componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, un framework se puede considerar como una aplicación genérica incompleta y configurable a la que se le puede añadir las últimas piezas para construir una aplicación concreta.

Los objetivos principales que persigue un framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones, en la actualidad existen frameworks para lenguajes como html5, css y JavaScript.

Los frameworks surgieron a partir de los patrones de diseño y se construyen sobre la base de lenguajes orientados a objetos. Esto facilita una mejor organización de los componentes y una mayor reutilización de código. Además, la mayoría de los frameworks implementan uno o más patrones de diseño de software que garantizan la escalabilidad del producto. Estos patrones de diseño son metodologías probadas para resolver problemas comunes en el diseño de aplicaciones.

Por lo tanto, los frameworks suelen representar implementaciones de patrones de diseño reconocidos, complementados con funciones que asisten al desarrollador. Sin embargo, en ocasiones, un framework puede ser más complejo de lo necesario para determinados proyectos (Aponte, 2014).

**BOOTSTRAP 5**

Bootstrap es un marco de trabajo (framework) de código abierto creado por Twitter y diseñado para facilitar el desarrollo web frontend. Fue desarrollado por Mark Otto y Jacob Thornton en 2011 y desde entonces se ha convertido en una de las herramientas más populares para crear sitios web responsivos y amigables con dispositivos móviles.

Bootstrap ofrece una amplia gama de componentes prediseñados, como botones, formularios, navegación y cuadros de diálogo, junto con un sistema de rejilla adaptable que ayuda a los desarrolladores a crear diseños flexibles y receptivos. Además, incluye CSS y JavaScript predefinidos que pueden ser fácilmente personalizados para adaptarse a las necesidades específicas de un proyecto.

La comunidad de Bootstrap es activa y ofrece una serie de recursos adicionales, como temas personalizados, complementos y documentación detallada, lo que hace que sea más accesible para desarrolladores de diferentes niveles de habilidad.

**Características**

Bootstrap presenta una serie de atributos que lo distinguen de otros frameworks:

* Simplifica la adaptación de la interfaz a diversos navegadores.
* Se vincula con las principales bibliotecas de JavaScript.
* Proporciona un diseño atractivo mediante Sass, cumpliendo los estándares de CSS.
* Es altamente liviano y se ajusta fácilmente a cualquier tipo de proyecto en desarrollo.
* Ofrece múltiples diseños predefinidos, ya sea con estructuras fijas de 940 píxeles o con diseños fluidos, que se adaptan a diferentes columnas o configuraciones (team, 2023).

**BIBLIOTECA JQUERY**

jQuery es una biblioteca de JavaScript rápida, pequeña y rica en funciones. Hace que cosas como el recorrido y la manipulación de documentos HTML, el manejo de eventos, y Ajax es mucho más simple con una API fácil de usar que funciona en una multitud de navegadores. Con una combinación de versatilidad y extensibilidad, jQuery ha cambiado la forma en que millones de personas escriben JavaScript (Foundation O. , 2023).

**PLUGIN DATATABLE**

DataTables es un complemento para la biblioteca jQuery Javascript. Es una herramienta altamente flexible, construida sobre las bases de la mejora progresiva, que agrega todas estas características avanzadas a cualquier tabla HTML (Ltd, 2007-2023).

DataTables está diseñado para mejorar las capacidades de las tablas HTML. Simplifica operaciones como búsqueda, ordenamiento, paginación y filtrado de datos, ofreciendo la posibilidad de gestionar de manera dinámica y eficiente grandes conjuntos de información en tablas en páginas web (Ltd, 2007-2023).

**BIBLIOTECA SWEETALERT2**

SweetAlert2 es una biblioteca JavaScript que posibilita la creación sencilla de ventanas modales personalizadas y visualmente atractivas. Proporciona una manera intuitiva de presentar alertas, confirmaciones y diálogos interactivos en sitios web, mejorando la experiencia de navegación del usuario (sweetalert2, 2023).

### 3.1.4 TECNOLOGÍAS BACKEND

**PHP (PHP: HYPERTEXT PREPROCESSOR)**

PHP es una sigla, un acrónimo de “PHP: Hypertext Preprocessor”, o sea, “Pre-procesador de Hipertexto marca PHP”. El hecho de que sea un “pre” procesador es lo que marca la diferencia entre el proceso que sufren las páginas web programadas con PHP del de aquellas páginas web comunes, escritas en el lenguaje HTML (Beati, 2011).

Para comprender la función de un preprocesador, es útil analizar el proceso previo a la visualización de una página desarrollada en PHP.

1. Al solicitar la visualización de una página con extensión .php desde el navegador, se inicia el proceso.
2. El servidor web en el hosting recibe la solicitud y, al detectar la extensión .php, redirige la petición a otro programa dentro del mismo servidor, conocido como intérprete de PHP. Este intérprete es una entidad invisible pero esencial para programar en PHP.
3. El intérprete busca y lee el archivo .php solicitado en el disco duro del hosting. Examina el código línea por línea en busca de marcas o etiquetas predefinidas por los programadores, que contienen instrucciones para el intérprete de PHP.
4. Cuando el intérprete de PHP encuentra estas instrucciones, las ejecuta y reemplaza las instrucciones del código HTML original, que estaban dentro de las etiquetas de PHP, con los datos resultantes de dichas instrucciones.
5. El software de PHP procesa las instrucciones y devuelve al servidor web el texto y el código HTML generado para que sean entregados al navegador, interpretándolos como si fuesen parte original del código HTML.
6. En el código que finalmente recibe el navegador, las instrucciones previamente escritas en el software PHP desaparecen, sustituidas por "el resultado de la ejecución de dichas instrucciones", ocultando la manipulación realizada por el software PHP (Beati, 2011).

**SQL (STRUCTURED QUERY LANGUAGE)**

SQL tuvo su origen en un proyecto llevado a cabo en el IBM San José Research Laboratory, liderado por E.F. Codd entre 1969 y 1970. Durante este período se publicaron documentos que introducían el concepto de bases de datos relacionales, presentando lenguajes de interfaz como SQUARE y SEQUEL para acceder a estas bases de datos. Este proyecto se conoció como sistema R y vio la luz a mediados de la década de los 70. El lenguaje SEQUEL evolucionó hasta convertirse en SQL a finales de esa misma década. Inicialmente, SQL se implementó en DB2, la primera base de datos relacional de IBM, orientada hacia las máquinas IBM. Sin embargo, rápidamente se adaptó para ser utilizado en microcomputadoras por parte de Oracle Corporation & Tecnology Relational.

A lo largo del tiempo, SQL se ha convertido en un lenguaje de interfaz estándar para bases de datos relacionales, inicialmente implementado en sistemas IBM y más tarde extendido a microcomputadoras. En febrero de 1987, el American National Standards Institute (ANSI) adoptó SQL como un lenguaje estándar para acceder a sistemas de gestión de bases de datos relacionales. Posteriormente, la International Standard Organization (ISO) hizo lo propio, lo cual fue reconocido por muchas compañías de software que comenzaron a desarrollar productos basados en este estándar.

SQL es un lenguaje "no-procedural" diseñado específicamente para la creación, mantenimiento y manipulación de bases de datos relacionales. (Alcala Castañeda, 1991).

### 3.1.5 BASES DE DATOS

Una base de datos representa un conjunto de datos conectados entre sí. Estos datos son hechos conocidos que tienen un significado implícito. Por ejemplo, podrían ser los nombres, números de teléfono y direcciones de personas que hemos registrado, ya sea en una libreta de direcciones organizada o utilizando software como DBASE IV o V, PARADOX o EXCEL en un computador personal.

Aunque la definición anterior es bastante amplia, es importante señalar que una base de datos suele ser más específica. En términos generales, una base de datos posee ciertas propiedades implícitas:

* Representa un aspecto del mundo real, a veces denominado "minimundo" o "universo de discurso". Los cambios en el mundo real se reflejan en la base de datos.
* Es un conjunto de datos lógicamente coherente, con un significado inherente. Una simple colección de datos aleatorios no puede considerarse una base de datos.
* Cada base de datos se crea, estructura y llena con datos para un propósito específico. Está orientada a un grupo de usuarios y tiene aplicaciones específicas que les interesan.

En resumen, una base de datos tiene una fuente de donde provienen los datos, interactúa de cierta manera con el mundo real y está dirigida a un público interesado activamente en su contenido (Elmasri & Navathe).

**Tipos de bases de datos**

Las bases de datos pueden ser clasificadas de diversas formas según el criterio que se elija para dicha clasificación.

**En relación a la variabilidad de los datos almacenados:**

* Las bases de datos estáticas, destinadas exclusivamente a la lectura, conservan datos históricos para análisis temporal y toma de decisiones.
* Las bases de datos dinámicas permiten la modificación de la información a lo largo del tiempo, facilitando operaciones de actualización y edición, además de las consultas básicas.

**En cuanto a su contenido:**

* Las bases de datos bibliográficas contienen registros de fuentes primarias para su ubicación, ofreciendo información sobre el autor, fecha de publicación, etc. Pueden incluir resúmenes, pero no el texto completo, a diferencia de las bases de datos a texto completo que almacenan fuentes primarias.
* Las bases de datos numéricas se enfocan en cifras o números, como conjuntos de resultados de análisis de laboratorio, entre otros.
* Las bases de datos de texto completo almacenan las fuentes originales, como todo el contenido de todas las ediciones de revistas científicas (Navarro López, 2009).

**Bases de datos relacionales**

Las bases de datos relacionales se basan en el modelo relacional y usan un conjunto de tablas para representar tanto los datos como las relaciones entre ellos. También incluyen un LMD y un LDD. La mayor parte de los sistemas de base de datos relacionales comerciales emplean el lenguaje SQL (Silberschatz y otros, 2006).

**Lenguaje de definición de datos (DDL)**

Los principales comandos en el lenguaje de definición de datos (DDL) de SQL incluyen:

* CREATE TABLE
* CREATE INDEX
* ALTER TABLE
* RENAME TABLE
* DROP TABLE
* DROP INDEX

Estos comandos se emplean para crear, modificar y eliminar las estructuras lógicas que conforman el modelo lógico de una base de datos. Son útiles en cualquier momento para realizar ajustes en la estructura de la base de datos. Además, existen otros comandos adicionales que permiten especificar detalles físicos de almacenamiento, pero no serán abordados aquí debido a su naturaleza específica para cada sistema .

**Lenguaje de manipulación de datos (DML)**

El lenguaje de consulta de SQL es declarativo, lo que se conoce también como no procedural. Esto implica que permite especificar qué datos se recuperan sin detallar los procedimientos para obtenerlos. Puede ser utilizado de manera interactiva para consultas, integrado dentro de un lenguaje de programación principal, o como un lenguaje completo por sí mismo, permitiendo cálculos mediante el uso de SQL/PSM (Persistent Stored Modules = Módulos de Almacenamiento Persistentes).

Los enunciados principales de manipulación de datos (DML) en SQL son:

* SELECT
* UPDATE
* INSERT
* DELETE

Estos comandos son fundamentales para realizar diversas operaciones, como la selección, actualización, inserción y eliminación de datos en una base de datos (Ricardo, 2009).

### 3.1.6 MOTOR DE BASE DE DATOS

MariaDB Server es un sistema de gestión de bases de datos relacionales que opera bajo un modelo de código abierto. Reconocido como uno de los servidores de bases de datos más populares globalmente, cuenta con usuarios notables como Wikipedia, WordPress.com y Google. Su distribución se realiza bajo la licencia GPLv2, asegurando así su naturaleza abierta y disponible para la comunidad.

Este servidor es versátil y se adapta a diversas funciones: desde manejar datos de transacciones con alta disponibilidad hasta análisis de datos. Es utilizado como servidor integrado y cuenta con un amplio respaldo de herramientas y aplicaciones que lo respaldan en su funcionamiento .

**Historia**

Cuando Oracle adquirió MySQL en 2009, el creador de MySQL, Michael “Monty” Widenius, tuvo preocupaciones sobre el rumbo que podría tomar el proyecto bajo el control de Oracle. Por esta razón, Monty decidió crear una bifurcación (fork) del proyecto, al que llamó MariaDB. La elección del nombre fue simbólica: MySQL se nombró en honor a su primera hija, My, mientras que MariaDB honra a su segunda hija, María.

La mayoría de los desarrolladores originales se unieron a este nuevo proyecto, y desde entonces, MariaDB Server ha experimentado un desarrollo constante y acelerado (Foundation M. , 2009-2023).

### 3.1.7 SISTEMA GESTOR DE BASES DE DATOS (SGBD)

Un Sistema de Gestión de Bases de Datos (DBMS, por sus siglas en inglés) es un sistema de software diseñado para facilitar el acceso a la información almacenada en una base de datos. Su objetivo principal radica en ofrecer un método efectivo para definir, almacenar y recuperar datos de manera eficiente.

El DBMS interactúa con programas de aplicación, permitiendo que los datos almacenados en la base de datos sean accesibles por múltiples aplicaciones y usuarios. Además, despliega un control centralizado sobre la base de datos, previniendo el acceso de usuarios no autorizados o fraudulentos, garantizando así la privacidad y la seguridad de los datos almacenados (Gunjal, 2003):

A continuación se describen una serie de características y beneficios de los sistemas de gestión de bases de datos (DBMS):

* **Independencia de los datos:** Permite cambios en un nivel de la base de datos sin afectar a otros niveles, facilitando modificaciones en hardware, almacenamiento o datos adicionales sin reescribir programas de aplicación.
* **Economía:** Busca maximizar la eficiencia en el uso, almacenamiento y modificación de datos para reducir costos.
* **Exactitud e integridad:** A pesar de eliminar la redundancia, el control centralizado de la base de datos ayuda a mantener la precisión, mientras que los controles de integridad detectan y corrigen errores.
* **Recuperación de fallas:** Es crucial para sistemas multiusuario, asegurando una recuperación rápida sin pérdida de transacciones para mantener la precisión e integridad de los datos.
* **Privacidad y seguridad:** El DBMS proporciona control sobre el acceso no autorizado, garantizando la privacidad y la seguridad mediante un control centralizado.
* **Rendimiento:** Se enfoca en el tiempo de respuesta a las consultas y depende de la interacción entre el usuario y la base de datos.
* **Recuperación, análisis y almacenamiento:** Facilita estas operaciones fundamentales en la gestión de datos.
* **Compatibilidad:** Garantiza que el hardware y el software sean funcionales con las computadoras existentes.
* **Control de concurrencia:** Permite el acceso simultáneo a la base de datos, asegurando la integridad de los datos.
* **Apoyo y normas:** Brinda soporte para estructuras de archivos complejas, rutas de acceso y estándares de datos para facilitar el intercambio entre sistemas y usuarios (Gunjal, 2003).

### 3.1.8 PROGRAMAS ADICIONALES

**XAMPP**

Mucha gente conoce de primera mano que no es fácil instalar un servidor de web Apache y la tarea se complica si le añadimos MariaDB, PHP y Perl. El objetivo de XAMPP es crear una distribución fácil de instalar para desarrolladores que se están iniciando en el mundo de Apache. XAMPP viene configurado por defecto con todas los opciones activadas. XAMPP es gratuito tanto para usos comerciales como no comerciales. En caso de usar XAMPP comercialmente, asegúrate de que cumples con las licencias de los productos incluidos en XAMPP. Actualmente XAMPP tiene instaladores para Windows, Linux y OS X.

**Fundadores**

**Kai 'Oswald' Seidler**

Oswald es uno de los co-fundadores originales de Apache Friends. Se graduó en 1999 de la Technical University of Berlin con un Diplom-Informatiker degree (equivalente a una ingeniería en software). En los 90s creó y administró el servidor de IRCnet más grande irc.fu-berlin.de, y co-administró uno de los servidores anónimos de FTP más grandes del mundo ftp.cs.tu-berlin.de. Desde 1993 a 1998 fue miembro del grupo internacional "Projektgruppe Kulturraum Internet", un proyecto basado en Berlin sobre cultura y organizaciones basadas en internet. En 2006 publicó su tercer libro, Das XAMPP-Handbuch, con Addison Wesley. De 2009 a 2011, fue evangelista de tecnología para productos de web en Sun Microsystems/Oracle.

**Kay Vogelgesang**

Junto con Oswald, Kay co-fundó Apache Friends en 2002. Actualmente trabaja como ingeniero de sistemas freelance y ha escrito varios libros sobre tecnologías web como Apache, MySQL y XAMPP.

**La Licencia**

XAMPP es una compilación de software libre (similar a una distribución de Linux). Es gratuita y puede ser copiada libremente de acuerdo a la licencia GNU GPL. Únicamente la compilación de XAMPP está publicada bajo la licencia GPL (Friends, 2023).

**APACHE WEB SERVER**

El Proyecto Apache HTTP Server es un esfuerzo para desarrollar y mantener un servidor HTTP de código abierto para sistemas operativos modernos, incluyendo UNIX y Windows. El objetivo de este proyecto es proporcionar un servidor seguro, eficiente y extensible que ofrezca servicios HTTP en concordancia con los estándares actuales de HTTP.

El Servidor HTTP Apache ("httpd") fue lanzado en 1995 y ha sido el servidor web más popular en Internet desde abril de 1996. Celebró su 25 aniversario como proyecto en febrero de 2020.

El Servidor HTTP Apache es un proyecto de The Apache Software Foundation (Foundation T. A., 1997-2023).

**PHPMYADMIN**

phpMyAdmin es una herramienta de software libre escrita en PHP, diseñada para gestionar la administración de MySQL a través de la web. phpMyAdmin admite una amplia gama de operaciones en MySQL y MariaDB. Las operaciones frecuentemente utilizadas (gestión de bases de datos, tablas, columnas, relaciones, índices, usuarios, permisos, etc.) se pueden realizar a través de la interfaz de usuario, manteniendo aún la capacidad de ejecutar directamente cualquier declaración SQL.

El proyecto phpMyAdmin es miembro de Software Freedom Conservancy. SFC es una organización sin fines de lucro que ayuda a promover, mejorar, desarrollar y defender proyectos de Software Libre, Libre y de Código Abierto (FLOSS, por sus siglas en inglés) (Contributors, 2003-2023).



## MARCO METODOLÓGICO

**Ciclo de vida del software**

Se llama ciclo de vida del software a las fases por las que pasa un proyecto software desde que es concebido, hasta que está listo para usarse.

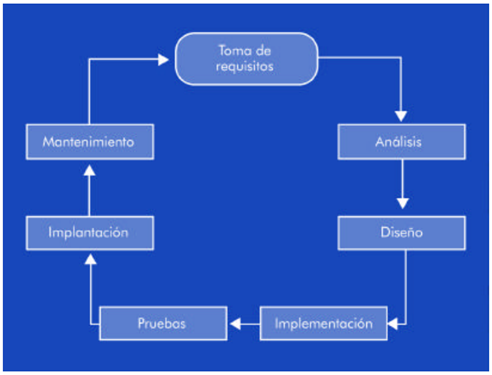
Típicamente, incluye las siguientes actividades: toma de requisitos, análisis, diseño, desarrollo, pruebas (validación, aseguramiento de la calidad), instalación (implantación), uso, mantenimiento y obsolescencia (Mas y otros, 2005).

Ilustración 2 Ciclo evolutivo del software

El proyecto tiende a pasar iterativamente por estas fases, en lugar de

hacerlo de forma lineal. Así pues, se han propuesto varios modelos (en cascada, incremental, evolutivo, en espiral, o concurrente, por citar algunos) para describir el progreso real del proyecto (Mas y otros, 2005).

### 3.2.1 MODELOS DE PROCESO PRESCRIPTIVO

**Modelos de proceso prescriptivo**

En sus inicios, los modelos de proceso prescriptivo surgieron como soluciones para organizar el caos del desarrollo de software. A lo largo de la historia, estos modelos tradicionales han aportado cierta estructura beneficiosa al trabajo de la ingeniería de software, sirviendo como una guía razonablemente efectiva para los equipos. A pesar de esto, el trabajo de ingeniería de software y el producto resultante aún se encuentran en un estado constantemente desafiante, justo en el límite entre el orden y el caos (Pressman, 2010).

**Modelo de la cascada**

El modelo de cascada, también conocido como ciclo de vida clásico, propone un enfoque sistemático y secuencial en el desarrollo de software. Este método inicia con la especificación de requerimientos por parte del cliente y progresa mediante fases de planificación, modelado, construcción y despliegue, culminando con el soporte al software finalizado. Este es un modelo secuencial donde cada fase (requisitos, diseño, implementación, pruebas, mantenimiento) se completa antes de pasar a la siguiente. Cada fase tiene sus propios hitos y entregables (Pressman, 2010).

**Modelo de proceso incremental**

En situaciones donde los requisitos iniciales del software están definidos, pero el tamaño del proyecto hace que un proceso lineal sea inviable, se opta por un enfoque incremental. Este modelo se elige cuando se necesita entregar rápidamente una funcionalidad limitada a los usuarios y se planea expandirla en entregas posteriores.

El enfoque incremental combina elementos de procesos lineales en etapas a lo largo del calendario de actividades. Cada secuencia lineal produce "incrementos" de software que pueden entregarse de manera similar a los pasos progresivos de un flujo de proceso evolutivo (Pressman, 2010).

**Modelo de proceso evolutivo**

En el desarrollo del software, al igual que en otros sistemas complejos, los requerimientos del negocio y del producto tienden a cambiar a lo largo del tiempo. Esta evolución hace que trazar una línea recta hacia el producto final sea poco realista. En ocasiones, las restricciones de tiempo en el mercado impiden la finalización de un software perfecto, lo que lleva a lanzar versiones limitadas para mitigar la presión de la competencia o del negocio. En situaciones donde se comprenden los requerimientos básicos pero los detalles o extensiones del sistema aún no están definidos, se requiere un modelo de proceso adaptable a esta evolución constante del producto.

Los modelos evolutivos son iterativos y se enfocan en desarrollar versiones progresivamente más completas del software. A continuación, se presentan dos modelos comunes de procesos evolutivos (Pressman, 2010).

### 3.2.2 METODOLOGÍA A EMPLEAR

**Modelo de cascada**

Hay veces en las que los requerimientos para cierto problema se comprenden bien: cuando el trabajo desde la comunicación hasta el despliegue fluye en forma razonablemente lineal. Esta situación se encuentra en ocasiones cuando deben hacerse adaptaciones o mejoras bien definidas a un sistema ya existente (por ejemplo, una adaptación para software de contabilidad que es obligatorio hacer debido a cambios en las regulaciones gubernamentales). También ocurre en cierto número limitado de nuevos esfuerzos de desarrollo, pero sólo cuando los requerimientos están bien definidos y tienen una estabilidad razonable.

El modelo de la cascada, a veces llamado ciclo de vida clásico, sugiere un enfoque sistemático y secuencial para el desarrollo del software, que comienza con la especificación de los requerimientos por parte del cliente y avanza a través de planeación, modelado, construcción y despliegue, para concluir con el apoyo del software terminado (Pressman, 2010).



Ilustración 3 Representación de la metodología en cascada

**Fases del modelo de cascada**

* **Requerimientos:** En esta etapa, se recopilan y documentan todos los requisitos del software, incluyendo las necesidades y expectativas del cliente.
* **Diseño:** Se crea la arquitectura del sistema y se diseña la estructura del software basándose en los requerimientos recopilados. Aquí se definen los componentes, interfaces y módulos del sistema.
* **Implementación (Codificación):** En esta fase se lleva a cabo la codificación del software según el diseño establecido. Se traduce el diseño en código fuente ejecutable.
* **Pruebas:** Se realiza la verificación del software para asegurarse de que cumple con los requisitos establecidos. Se ejecutan pruebas para identificar errores y garantizar su corrección.
* **Despliegue (Instalación):** Una vez que el software ha pasado por las pruebas satisfactoriamente, se implementa en el entorno de producción o se entrega al cliente para su uso.
* **Mantenimiento:** Esta etapa implica realizar actualizaciones, correcciones de errores y mejoras continuas al software en respuesta a las necesidades cambiantes del usuario o del entorno.

Cada fase se lleva a cabo de manera secuencial y se espera que una fase no comience hasta que la anterior haya concluido. Este enfoque lineal y estructurado es característico del modelo de cascada.

**Desventajas**

El modelo de cascada ha sido uno de los paradigmas más antiguos en ingeniería de software, pero a lo largo de las últimas tres décadas, las críticas hacia este modelo han llevado incluso a sus más fervientes defensores a cuestionar su efectividad. Algunos de los problemas que surgen al aplicar el modelo de cascada incluyen:

* **Secuencialidad rígida:** En la práctica, es raro que los proyectos reales sigan estrictamente el flujo secuencial propuesto por el modelo. Aunque el modelo lineal permite repeticiones, lo hace de manera indirecta, lo que lleva a confusiones cuando se realizan cambios a medida que avanza el equipo del proyecto.
* **Dificultades en la definición de requerimientos:** Frecuentemente, resulta difícil para el cliente expresar de manera explícita todos los requerimientos. El modelo de cascada requiere esta claridad desde el inicio y enfrenta dificultades para manejar la incertidumbre natural que acompaña a muchos proyectos en sus etapas iniciales.
* **Falta de versiones funcionales tempranas:** Este modelo requiere que el cliente tenga paciencia, ya que no se dispone de una versión funcional del programa hasta que el proyecto está bastante avanzado. Esto puede ser problemático, ya que un error significativo descubierto tarde en el proceso puede ser catastrófico una vez que el programa está en funcionamiento.



## MARCO LEGAL

Vittorio Frosini destaca la conexión entre la informática y el derecho, donde se reconoce a la computadora como una herramienta utilizada por los juristas para crear bancos de datos legales y facilitar la administración de la justicia. Sin embargo, el uso de la computadora también plantea una serie de problemas que deben regularse mediante la ley.

Para muchos expertos legales, el derecho informático es un campo en desarrollo. La legislación mexicana relacionada con sistemas informáticos y, más recientemente, con redes de información como internet, ha sido limitada. En 1984, con el Acuerdo 114 de la Secretaría de Educación Pública, se permitió la inclusión de programas de cómputo en el Registro del Derecho de Autor, marcando la primera mención explícita de esta tecnología en la legislación mexicana. El área de mayor interés para la industria del software actualmente radica en la protección de los derechos de autor. (Salgado, 2014).

### 3.3.1 LEY FEDERAL DEL DERECHO DE AUTOR

(Publicada en el Diario Oficial de la Federación el 26 de diciembre de 1996)

TITULO SEGUNDO

Del Derecho de Autor

CAPITULO I

Reglas generales

**Artículo 13**

Los derechos de autor a que se refiere esta Ley se reconocen respecto de las obras de las siguientes ramas:

XI. Programas de cómputo;

XIV. De compilación, integrada por las colecciones de obras, tales como las enciclopedias, las antologías, y de obras u otros elementos como las bases de datos, siempre que dichas colecciones, por su selección o la disposición de su contenido o materias, constituyan una creación intelectual.

Las demás obras que por analogía puedan considerarse obras literarias o artísticas se incluirán en la rama que les sea más afín a su naturaleza.

TITULO QUINTO

De los Derechos Conexos

CAPITULO IV

De los Programas de Computación y las Bases de Datos

**Artículo 101**

Se entiende por programa de computación la expresión original en cualquier forma, lenguaje o código, de un conjunto de instrucciones que, con una secuencia, estructura y organización determinada, tiene como propósito que una computadora o dispositivo realice una tarea o función específica.

**Artículo 102**

Los programas de computación se protegen en los mismos términos que las obras literarias. Dicha protección se extiende tanto a los programas operativos como a los programas aplicativos, ya sea en forma de código fuente o de código objeto. Se exceptúan aquellos programas de cómputo que tengan por objeto causar efectos nocivos a otros programas o equipos.

**Artículo 103**

Salvo pacto en contrario, los derechos patrimoniales sobre un programa de computación y su documentación, cuando hayan sido creados por uno o varios empleados en el ejercicio de sus funciones o siguiendo las instrucciones del empleador, corresponden a éste.

Como excepción a lo previsto por el artículo 33 de la presente Ley, el plazo de la cesión de derechos en materia de programas de computación no está sujeto a limitación alguna.

**Artículo 104**

Como excepción a lo previsto en el artículo 27 fracción IV, el titular de los derechos de autor sobre un programa de computación o sobre una base de datos conservará, aún después de la venta de ejemplares de los mismos, el derecho de autorizar o prohibir el arrendamiento de dichos ejemplares. Este precepto no se aplicará cuando el ejemplar del programa de computación no constituya en sí mismo un objeto esencial de la licencia de uso.

**Artículo 105**

El usuario legítimo de un programa de computación podrá realizar el número de copias que le autorice la licencia concedida por el titular de los derechos de autor, o una sola copia de dicho programa siempre y cuando:

I. Sea indispensable para la utilización del programa, o

II. Sea destinada exclusivamente como resguardo para sustituir la copia legítimamente adquirida, cuando ésta no pueda utilizarse por daño o pérdida. La copia de respaldo deberá ser destruida cuando cese el derecho del usuario para utilizar el programa de computación.

**Artículo 106**

El derecho patrimonial sobre un programa de computación comprende la facultad de autorizar o prohibir:

I. La reproducción permanente o provisional del programa en todo o en parte, por cualquier medio y forma;

II. La traducción, la adaptación, el arreglo o cualquier otra modificación de un programa y la reproducción del programa resultante;

III. Cualquier forma de distribución del programa o de una copia del mismo, incluido el alquiler, y IV. La descompilación, los procesos para revertir la ingeniería de un programa de computación y el desensamblaje.

**Artículo 107**

Las bases de datos o de otros materiales legibles por medio de máquinas o en otra forma, que por razones de selección y disposición de su contenido constituyan creaciones intelectuales, quedarán protegidas como compilaciones. Dicha protección no se extenderá a los datos y materiales en sí mismos.

**Artículo 108**

Las bases de datos que no sean originales quedan, sin embargo, protegidas en su uso exclusivo por quien las haya elaborado, durante un lapso de 5 años.

**Artículo 109**

El acceso a información de carácter privado relativa a las personas contenida en las bases de datos a que se refiere el artículo anterior, así como la publicación, reproducción, divulgación, comunicación pública y transmisión de dicha información, requerirá la autorización previa de las personas de que se trate.

Quedan exceptuados de lo anterior, las investigaciones de las autoridades encargadas de la procuración e impartición de justicia, de acuerdo con la legislación respectiva, así como el acceso a archivos públicos por las personas autorizadas por la ley, siempre que la consulta sea realizada conforme a los procedimientos respectivos.

**Artículo 110**

El titular del derecho patrimonial sobre una base de datos tendrá el derecho exclusivo, respecto de la forma de expresión de la estructura de dicha base, de autorizar o prohibir:

I. Su reproducción permanente o temporal, total o parcial, por cualquier medio y de cualquier forma;

II. Su traducción, adaptación, reordenación y cualquier otra modificación;

III. La distribución del original o copias de la base de datos;

IV. La comunicación al público, y

V. La reproducción, distribución o comunicación pública de los resultados de las operaciones mencionadas en la fracción II del presente artículo.

**Artículo 111**

Los programas efectuados electrónicamente que contengan elementos visuales, sonoros, tridimensionales o animados quedan protegidos por esta Ley en los elementos primigenios que contengan.

**Artículo 112**

Queda prohibida la importación, fabricación, distribución y utilización de aparatos o la prestación de servicios destinados a eliminar la protección técnica de los programas de cómputo, de las transmisiones a través del espectro electromagnético y de redes de telecomunicaciones y de los programas de elementos electrónicos señalados en el artículo anterior.

**Artículo 113**

Las obras e interpretaciones o ejecuciones transmitidas por medios electrónicos a través del espectro electromagnético y de redes de telecomunicaciones y el resultado que se obtenga de esta transmisión estarán protegidas por esta Ley.

**Artículo 114**

La transmisión de obras protegidas por esta Ley mediante cable, ondas radioeléctricas, satélite u otras similares, deberán adecuarse, en lo conducente, a la legislación mexicana y respetar en todo caso y en todo tiempo las disposiciones sobre la materia.

**Artículo 231**

Constituyen infracciones en materia de comercio las siguientes conductas cuando sean realizadas con fines de lucro directo o indirecto:

V. Importar, vender, arrendar o realizar cualquier acto que permita tener un dispositivo o sistema cuya finalidad sea desactivar los dispositivos electrónicos de protección de un programa de computación;

(Unión, 1996).



## ANTECEDENTES

A

CAPÍTULO IV DESARROLLO

## PROCEDIMIENTO Y DESCRIPCIÓN DE LAS ACTIVIDADES REALIZADAS

* **Ingeniería y análisis del sistema:**

Durante esta etapa, se llevó a cabo un análisis de las necesidades y requerimientos del sistema para comprender a fondo las necesidades de la empresa.

* **Análisis de los requisitos del software:**

Se documentaron y analizaron los requisitos del software, identificando características clave y restricciones necesarias para el desarrollo del sistema.

* **Diseño del sistema:**

Ya teniendo claras las necesidades de la empresa se diseñaron las interfaces de usuario teniendo como objetivo ser lo más simples e intuitivas para que los usuarios finales puedan interactuar con los diferentes módulos de forma ágil.

* **Desarrollo del software (codificación):**

En esta etapa se procedió con la codificación del software siguiendo las directrices establecidas en la fase de diseño. Se emplearon las mejores prácticas de programación y se realizaron revisiones periódicas para asegurar la calidad del código.

* **Prueba del sistema:**

Se llevaron a cabo pruebas exhaustivas para garantizar el correcto funcionamiento del sistema. Esto incluyó pruebas de sistema para detectar y corregir posibles fallos, asegurando así, la coherencia con los requisitos establecidos. También se realizaron pruebas de rendimiento para ofrecer una mejor experiencia al usuario.

* **Implementación y evaluación del software:**

Una vez completadas las pruebas, se procedió a la implementación del software en el entorno de la escuela. Se monitoreó su desempeño y se realizaron evaluaciones para ajustar y mejorar su funcionalidad de acuerdo con las necesidades educativas.

* **Mantenimiento del software:**

Se estableció un plan de mantenimiento preventivo y correctivo para garantizar la estabilidad y actualización continua del software. Se atendieron solicitudes del personal educativo y administrativo, se corrigieron errores y se realizaron mejoras según las necesidades identificadas en el contexto escolar.

CAPÍTULO V RESULTADOS

## RESULTADOS, PLANOS, GRAFICAS, PROTOTIPOS, MANUALES, ETC.

Resultados obtenidos mediante la aplicación de encuestas:

Gráfica 1 Pregunta 1

Gráfica 2 Pregunta 2

Gráfica 3 Pregunta 3

Gráfica 4 Pregunta 4

Gráfica 5 Pregunta 5

Gráfica 6 Pregunta 6

Gráfica 7 Pregunta 7

Gráfica 8 Pregunta 8

CAPÍTULO VI CONCLUSIONES Y RECOMENDACIONES

## CONCLUSIONES

El primer paso del proyecto consistió en identificar y documentar los requerimientos del sistema a desarrollar. Para ello se llevó a cabo un proceso exhaustivo de recolección de datos relevantes y análisis de las necesidades de la empresa. La información recopilada se tradujo en una documentación clara y concisa de los requerimientos funcionales y no funcionales del sistema.

Se optó por la implementación de un Sistema de Gestión de Bases de Datos (SGBD) para administrar y organizar la información del sistema de manera eficiente. Se evaluaron diferentes alternativas y se seleccionó MySQL como SGBD debido a sus ventajas en la gestión de datos estructurados. Todo esto siguiendo prácticas de diseño y normalización de la base de datos para garantizar la coherencia e integridad de los datos.

Se realizaron diseños de interfaces de usuario intuitivas y responsivas utilizando el Framework Bootstrap 5. El enfoque en el diseño centrado en el usuario permitió la creación de interfaces atractivas y funcionales, mejorando la experiencia del usuario final.

Se realizaron rigurosas pruebas en busca de errores para prevenir futuros fallos en el sistema asegurando así un software de calidad y confianza. Se corrigieron los errores encontrados de manera proactiva, asegurando la estabilidad del software antes de su implementación.

Una vez completadas las fases de desarrollo, pruebas y correcciones, el sistema fue implementado y puesto en producción.

## RECOMENDACIONES

* Mantener siempre actualizada la información del sistema.
* Ofrecer sesiones de capacitación cada que haya nuevo personal encargado de administrar el sistema. Esto ayudará a maximizar la adopción del sistema y a asegurar su uso efectivo evitando confusiones y/o alteraciones en registros importantes.
* Mantener la cantidad de administradores regulada, si bien el sistema es capaz de operar con múltiples cuentas administrativas, es recomendable tener controlado la cantidad de usuarios que pueden realizar alteraciones en el sistema.
* Realizar un análisis exhaustivo del rendimiento del sistema para identificar posibles cuellos de botella y optimizar la velocidad de respuesta. Esto podría incluir mejoras en la consulta a la base de datos o la optimización del código fuente.
* Evaluar la posibilidad de agregar funcionalidades adicionales que enriquezcan la gestión de la biblioteca, como la implementación de un sistema de recomendación de libros, gestión de préstamos avanzada.

CAPÍTULO VII COMPETENCIAS DESARROLLADAS

## COMPETENCIAS DESARROLLADAS Y/O APLICADAS

**MATERIAS APLICADAS:**

A continuación se enlistan las materias que fueron base fundamental en la realización del proyecto:

* **AEF1031 -** FUNDAMENTOS DE BASE DE DATOS
* **SCD1027 -** TÓPICOS AVANZADOS DE PROGRAMACIÓN WEB
* **SCC1007 -** FUNDAMENTOS DE INGENIERÍA DE SOFTWARE
* **SCA1025 -** TALLER DE BASE DE DATOS
* **SCB1001 -** ADMINISTRACIÓN DE BASE DE DATOS
* **SCD1011 -** INGENIERÍA DE SOFTWARE
* **AEB1055 -** PROGRAMACIÓN WEB
* **SCG1009 -** GESTIÓN DE PROYECTOS DE SOFTWARE
* **ACA0909 -** TALLER DE INVESTIGACIÓN I
* **ISD2101 -** TÓPICOS AVANZADOS DE PROGRAMACIÓN WEB
* **ISB2105 -** INGENIERÍA DE REQUERIMIENTO
* **ACA0910 -** TALLER DE INVESTIGACIÓN II

Para la elaboración de este proyecto se aplicaron conocimientos adquiridos en las materias anteriormente mencionadas, mismas que fueron abordadas durante cuatro años de estudio de la carrera de Ingeniería en Sistemas Computacionales.

FUENTES DE INFORMACIÓN

Aguilar, L. J. (2008). *FUNDAMENTOS DE PROGRAMACIÓN. Algoritmos, estructuras de datos y objetos* (Cuarta ed.). (J. L. García, & C. Sánchez, Edits.) España: McGRAW-HILL/INTERAMERICANA DE ESPAÑA, S. A. U.

Alcala Castañeda, M. (1991). *Estudio sobre el lenguaje estructurado de consulta SQL, y bases de datos relacionales.* Tesis de licenciatura, Universidad Nacional Autónoma de México, Facultad de Ciencias, UNAM. Retrieved 2023, from https://hdl.handle.net/20.500.14330/TES01000161305

Aponte, Á. M. (2014). *GUÍA COMPARATIVA DE FRAMEWORKS PARA LOS LENGUAJES HTML 5, CSS Y JAVASCRIPT PARA EL DESARROLLO DE APLICACIONES WEB.* Monografía de licenciatura, Universidad Tecnológica de Pereira, Facultad de ingenierías. Ingeniería de Sistemas y Computación Pereira. Retrieved 2023, from https://hdl.handle.net/11059/4577

Beati, H. (2011). *PHP Creación de páginas Web dinámicas* (Primera ed.). (D. Fernández, Ed.) Buenos Aires: Alfaomega Grupo Editor Argentino.

Buriticá, O. I. (1999). *La Esencia de la Lógica de Programación - Básico.* Colombia: Papiro.

Carballeiro, G. (2012). *Diseño web con HTML y CSS.* Creative Andina Corp.

Contributors, p. (2003-2023). *phpMyAdmin*. Retrieved 2023, from https://www.phpmyadmin.net/

Elmasri, R., & Navathe, S. B. (s.f.). *SISTEMAS DE BASES DE DATOS Conceptos fundamentales* (Segunda ed.). (R. E. García, F. L. Gamino, & A. Illarramendi, Trads.) Addison-Wesley Iberoamericana.

Foundation, M. (1998-2023). *CSS - MDN Web Docs*. Retrieved 2023, from https://developer.mozilla.org/es/docs/Web/css

Foundation, M. (1998-2023). *JavaScript - MDN Web Docs - Mozilla*. Retrieved 2023, from https://developer.mozilla.org/es/docs/Web/JavaScript

Foundation, M. (2009-2023). *MariaDB en resumen*. Retrieved 2023, from https://mariadb.org/es/

Foundation, M. (2023). *HTML: Lenguaje de etiquetas de hipertexto - MDN Web Docs*. Retrieved 2023, from https://developer.mozilla.org/es/docs/Web/HTML

Foundation, O. (2023). *JQuery write less, do more*. Retrieved 2023, from https://jquery.com/

Foundation, T. A. (1997-2023). *Welcome! - The Apache HTTP Server Project*. Retrieved 2023, from https://httpd.apache.org/

Friends, A. (2023). *About the XAMPP project*. Retrieved 2023, from https://www.apachefriends.org/es/about.html

Gauchad, J. D. (2017). *El gran libro de HTML5, CSS3 y JavaScript.* MARCOMBO, S.A.

Gunjal, B. (2003). Database Management: Concepts and Design. https://www.researchgate.net/publication/257298522\_Database\_System\_Concepts\_and\_Design

KENDALL, K. E., & KENDALL, J. E. (2011). *ANÁLISIS Y DISEÑO DE SISTEMAS* (Octava ed.). (L. M. Castillo, Ed.) Estado de México, México: Pearson Educación de México, S.A. DE C.V.

KIMMEL, P. (2008). *Manual de UML.* D.F., México: McGRAW-HILL INTERAMERICANA EDITORES S.A. DE C.V.

Ltd, S. (2007-2023). *DataTables | Table plug-in for JQuery*. Retrieved 2023, from https://datatables.net/

Mas, J., Jiménez, D. M., Ginestà, M. G., & González, A. P. (2005). *Ingeniería del software en entornos de SL.* Fundació per a la Universitat Oberta de Catalunya.

Navarro López, R. (2009). *Teoría y conceptos de la administración de bases de datos conjuntando su aplicación basada en software libre de un sistema de seguimiento y control de errores del manejador de bases de datos Sybase.* Tesis de licenciatura, Universidad Nacional Autónoma de México, Facultad de Estudios Superiores Aragón, UNAM. Retrieved 2023, from https://hdl.handle.net/20.500.14330/TES01000649796

Pressman, R. S. (2010). *Ingeniería de software. Un enfoque práctico* (Séptima ed.). D.F., México: McGRAW-HILL INTERAMERICANA EDITORES, S.A. DE C.V.

Puig, J. C. (2018). *CSS3 y Javascript avanzado.* FUOC. Fundació per a la Universitat Oberta de Catalunya.

Ricardo, C. M. (2009). *BASES DE DATOS.* D. F., México: McGRAW-HILL INTERAMERICANA EDITORES, S.A. de C.V.

Salgado, L. L. (2014). *DERECHO INFORMÁTICO* (Primera ed.). (J. E. Callejas, Ed.) D.F., México: GRUPO EDITORIAL PATRIA S.A. DE C.V.

Senn, J. A. (2001). *ANÁLISIS Y DISEÑO DE SISTEMAS DE INFORMACIÓN.* México: MCGRAW-HILL.

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2006). *FUNDAMENTOS DE BASES DE DATOS* (Quinta ed.). (C. S. González, Ed.) McGraw-Hill/Interamericana de España, S. A. U.

sweetalert2. (2023). *Sweetalert2 - a beautiful, responsive, customizable, accessible (WAI-ARIA) replacement for JavaScript's popup boxes*. Retrieved 2023, from https://sweetalert2.github.io/

team, B. (2023). *Bootstrap · The most popular HTML, CSS, and JS library in the world*. Retrieved 2023, from https://getbootstrap.com/docs/5.2/getting-started/introduction/

Unión, C. d. (1996). *Ley Federal del Derecho de Autor.* Retrieved 2023, from https://www.diputados.gob.mx/LeyesBiblio/pdf/LFDA.pdf

Villa, A. I. (2019). *IMPLEMENTACIÓN DE BOOTSTRAP EN EL DESARROLLO WEB.* Tesis de licenciatura, Universidad Autónoma de México, Facultad de Estudios Superiores Cuautitlán. Retrieved 2023, from https://repositorio.unam.mx/contenidos/3444033

ANEXOS

**REQUERIMIENTOS**

**FUNCIONALES**

**USUARIO ADMINISTRADOR:**

Tabla 1 Requerimientos funcionales

|  |  |  |
| --- | --- | --- |
|  | **Requerimiento** | **Descripción** |
| Acceso y Seguridad | Iniciar sesión en el sistema | Se requiere obligatoriamente iniciar sesión con credenciales de administrador para acceder al panel principal para la gestión del sistema. |
| Visualización | Visualización de préstamos, libros, alumnos, etc. | El administrador podrá acceder a los diferentes módulos y visualizar los registros almacenados. |
| Búsqueda | Búsqueda de libros | El administrador podrá consultar si algún libro se encuentra dentro de la biblioteca o en dado caso consultar que libros se encuentran en estado de préstamo. |
| Gestión | Dar de alta nuevos libros | Con la sesión iniciada, el administrador podrá registrar nuevos libros dentro del sistema. |
| Dar de alta a nuevos usuarios | En caso de ser necesario se podrá dar de alta a más administradores para gestionar el sistema, estas cuentas se podrán dar de baja temporalmente una vez ya no sean requeridas. |
| Registrar prestamos | El administrador tiene la capacidad de registrar préstamos de libros y la fecha de entrega de los mismos. |
| Recibir libros | Solo usuarios con acceso al sistema pueden modificar los datos de recepción de libros (en préstamo). |
| Actualizar datos de registros | Se podrá modificar la información de los diferentes registros. Esto es útil cuando se cometen errores al momento de crear los registros o para actualizar la información de los mismos. |
| Dar de baja registros | El administrador podrá remover del sistema información que ya no se considere necesaria, como por ejemplo, cuando determinados libros, usuarios inactivos, etc. que ya no formen parte de la biblioteca. |

**NO FUNCIONALES**

**RNF1: Multiusuario.**

Aplicación multiusuario que permite la creación de múltiples cuentas de administración para mayor flexibilidad durante la gestión de la biblioteca.

**RNF2: Interfaz.**

El sistema cuenta con una interfaz intuitiva y fácil de utilizar, tiene un aspecto minimalista para no saturar al usuario con información irrelevante.

**RNF3: Seguridad.**

La información registrada en el sistema se encuentra almacenada de forma local en el equipo, evitando así que terceras personas puedan manipular los registros.

**RNF4: Escalabilidad.**

Es una herramienta altamente escalable debido a que se compone de módulos diseñados para cada tarea especifica, dichos módulos trabajan de manera independiente lo cual permite nuevas implementaciones para expandir el software.

**RNF5: Rendimiento.**

El sistema utiliza tecnologías actuales para otorgar el mejor rendimiento posible en cualquier dispositivo, logrando así operar de forma óptima y fluida.

**ENCUESTAS**

Encuesta #1:

1. ¿Considera factible que los administradores de la biblioteca utilicen el sistema propuesto?

R: Sí

1. ¿Cree usted que almacenar datos en medios digitales es una forma segura?

R: No

1. ¿Conoce algún sistema existente que cumpla con las características del software propuesto?

R: No

1. ¿Está a favor de implementar un sistema administrativo para la biblioteca escolar?

R: Sí

1. ¿Cree usted que es importante tener un buen control de los préstamos de libros realizados en una biblioteca?

R: Sí

1. ¿Considera que se le dará uso al sistema propuesto?

R: Sí

1. ¿Considera usted que el software propuesto cumple con características importantes de gestión?

R: Sí

1. ¿Cree usted que el software propuesto va a ser de utilidad para la escuela?

R: Sí

**DISEÑO DE LA BASE DE DATOS**

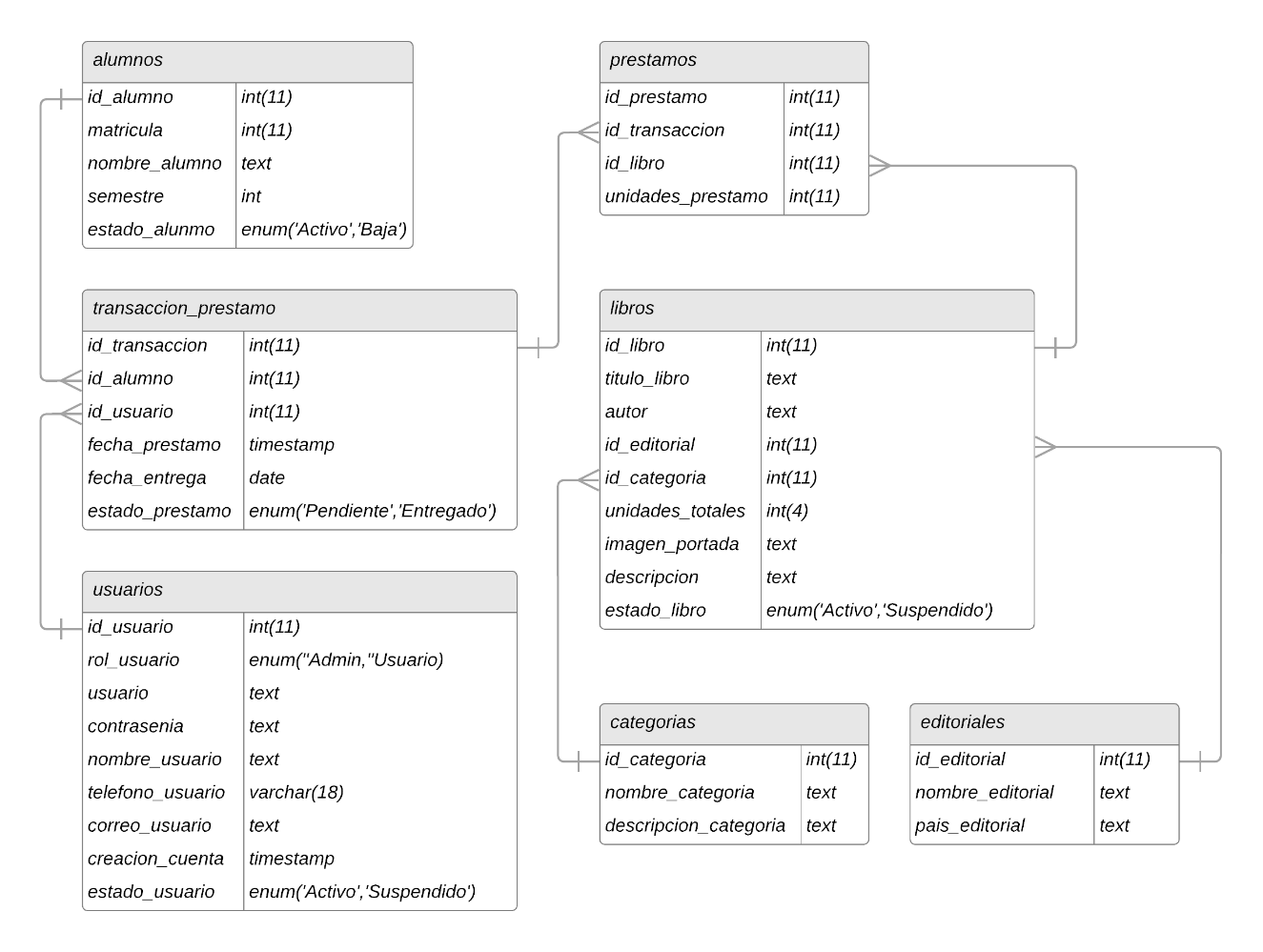


Ilustración 4 Diseño de la base de datos

**DICCIONARIO DE DATOS**

**Tabla alumnos**

Tabla 2 Diccionario de datos de la tabla alumnos

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Columna** | **Tipo de dato** | **Tamaño** | **Descripción** | **NULL** |
| PK | id\_alumno | int | 11 | Número de identificación para alumnos. | No |
|  | matricula | int | 11 | Matricula de los estudiantes. | No |
|  | nombre\_alumno | text |  | Nombre completo del alumno. | No |
|  | semestre | int | 11 | Semestre actual del alumno. | No |
|  | estado\_alumno | enum | 2 | Estatus de los alumnos (Activo/Baja) | No |

**Tabla categorias**

Tabla 3 Diccionario de datos de la tabla categorias

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Columna** | **Tipo de dato** | **Tamaño** | **Descripción** | **NULL** |
| PK | id\_categoria | int | 11 | Número de identificación para categorías. | No |
|  | nombre\_categoria | text |  | Nombre de las categorías. | No |
|  | descripcion\_categoria | text |  | Breve descripción de cada categoría. | No |

**Tabla editoriales**

Tabla 4 Diccionario de datos de la tabla editoriales

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Columna** | **Tipo de dato** | **Tamaño** | **Descripción** | **NULL** |
| PK | id\_editorial | int | 11 | Número de identificación para editoriales. | No |
|  | nombre\_editorial | text |  | Nombre de las editoriales. | No |
|  | país\_editorial | text |  | País de origen de las editoriales. | No |

**Tabla libros**

Tabla 5 Diccionario de datos de la tabla libros

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Columna** | **Tipo de dato** | **Tamaño** | **Descripción** | **NULL** |
| PK | id\_libro | int | 11 | Número de identificación para libros. | No |
|  | titulo\_libro | text |  | Título de los libros. | No |
|  | autor | text |  | Autor o autores de los libros. |  |
| FK | id\_editorial | int | 11 | Número de identificación para las editoriales. | No |
| FK | id\_categoria | int | 11 | Número de identificación para las categorías. | No |
|  | unidades\_totales | int | 4 | Unidades totales de cada libro. | No |
|  | unidades\_restantes | int | 11 | Unidades disponibles para préstamos. | No |
|  | imagen\_portada | text |  | Imagen de portada y referencia del libro. | No |
|  | descripcion | text |  | Breve descripción de los libros. | No |
|  | estado\_libro | enum | 2 | Estatus de los libros (Activo/Inactivo) | No |

**Tabla prestamos**

Tabla 6 Diccionario de datos de la tabla prestamos

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Columna** | **Tipo de dato** | **Tamaño** | **Descripción** | **NULL** |
| PK | id\_prestamo | int | 11 | Número de identificación para préstamos. | No |
| FK | id\_transaccion | int | 11 | Número de identificación para transacciones. | No |
| FK | id\_libro | int | 11 | Número de identificación para libros. | No |
|  | unidades\_prestamo | int | 11 | Cantidad de libros prestados. | No |

**Tabla transaccion\_prestamo**

Tabla 7 Diccionario de datos de la tabla transaccion\_prestamo

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Columna** | **Tipo de dato** | **Tamaño** | **Descripción** | **NULL** |
| PK | id\_transaccion | int | 11 | Número de identificación para transacciones de préstamo. | No |
| FK | id\_alumno | int | 11 | Número de identificación para alumnos. | No |
| FK | id\_usuario | int | 11 | Número de identificación para usuarios. | No |
|  | fecha\_prestamo | timestamp |  | Fecha de registro del préstamo. | No |
|  | fecha\_entrega | date |  | Fecha estimada de devolución. | No |
|  | estado\_prestamo | enum | 2 | Estatus de los prestamos (Pendiente/Entregado) |  |

**Tabla usuarios**

Tabla 8 Diccionario de datos de la tabla usuarios

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Columna** | **Tipo de dato** | **Tamaño** | **Descripción** | **NULL** |
| PK | id\_usuario | int | 11 | Número de identificación para usuarios. | No |
|  | rol­­\_usuario | enum | 2 | Nivel de privilegio de los usuarios. | No |
|  | usuario | text |  | Nombre corto de usuario (para iniciar sesión) | No |
|  | contrasenia | text |  | Contraseña de los usuarios. | No |
|  | nombre\_usuario | text |  | Nombre completo de los usuarios. | No |
|  | teléfono\_usuario | varchar | 18 | Teléfono de contacto. | No |
|  | correo\_usuario | text |  | Correo de contacto. | No |
|  | creacion\_cuenta | timestamp |  | Fecha de creación de las cuentas. | No |
|  | estado\_usuario | enum | 2 | Estatus de los usuarios (Activo/Suspendido) | No |

**DIAGRAMA ENTIDAD-RELACIÓN**

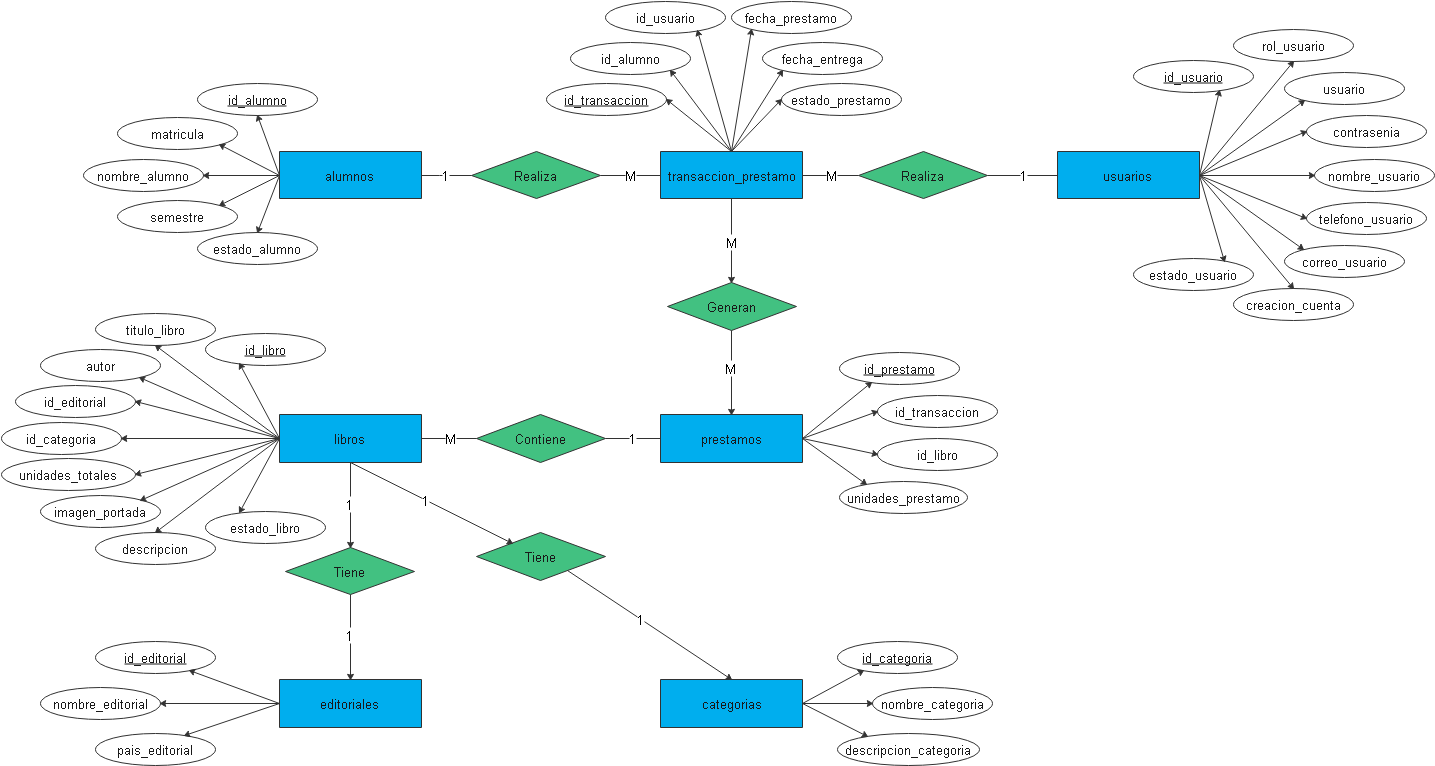


Ilustración 5 Diagrama Entidad-Relación

**PSEUDOCÓDIGOS**

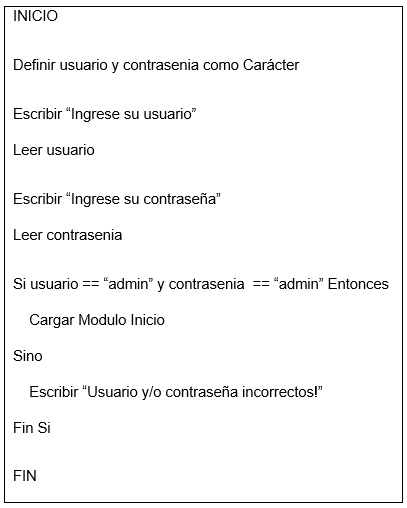
ALGORITMO PARA INICIO DE SESIÓN.

Ilustración 6 Pseudocódigo inicio sesión

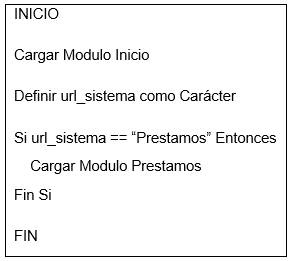
ALGORITMO PARA VISUALIZAR LISTA DE PRÉSTAMOS.

Ilustración 7 Pseudocódigo visualización módulo préstamos

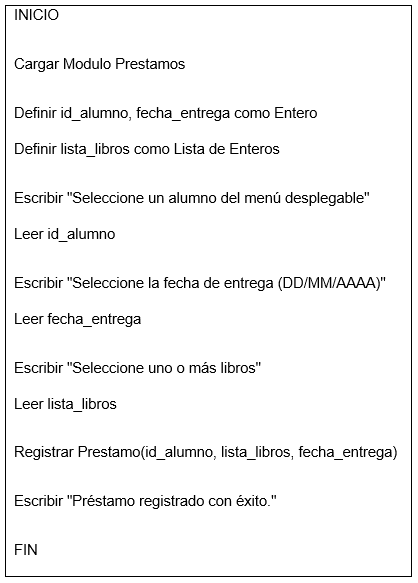
ALGORITMO PARA REGISTRAR PRÉSTAMO.

Ilustración 8 Pseudocódigo registro de préstamo

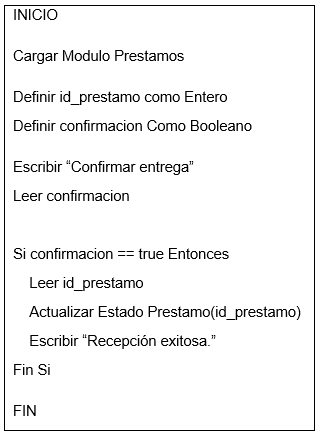
ALGORITMO PARA ACTUALIZAR ESTADO DE PRÉSTAMO.

Ilustración 9 Pseudocódigo actualizar estado préstamo

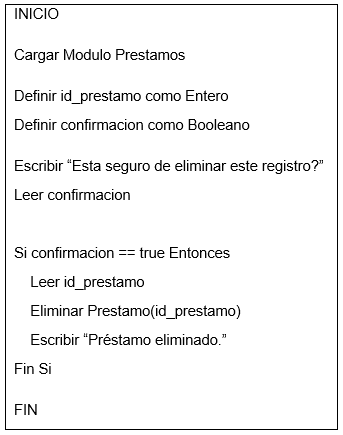
ALGORITMO PARA ELIMINAR REGISTRO DE PRÉSTAMO.

Ilustración 10 Pseudocódigo eliminar registro préstamo

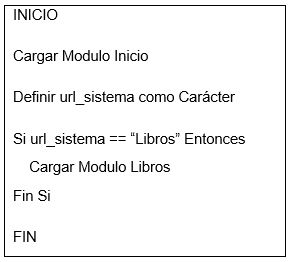
ALGORITMO PARA VISUALIZAR LISTA DE LIBROS.

Ilustración 11 Pseudocódigo visualización módulo libros

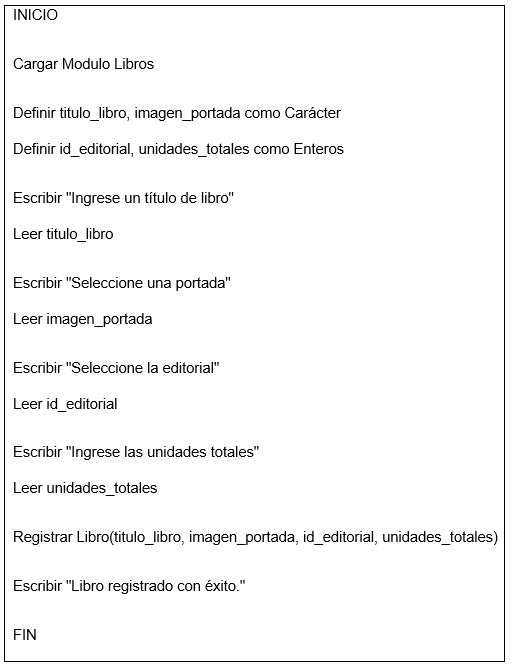
ALGORITMO PARA AGREGAR LIBRO.

Ilustración 12 Pseudocódigo registro de libro

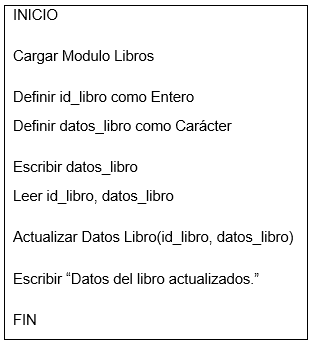
ALGORITMO PARA ACTUALIZAR DATOS DE LIBRO.

Ilustración 13 Pseudocódigo actualizar datos libro

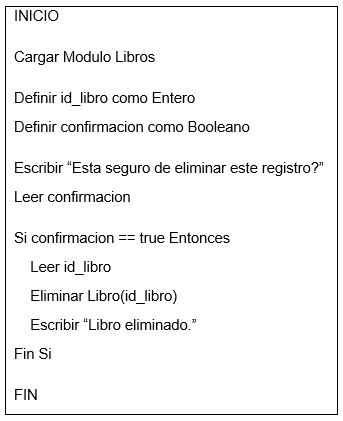
ALGORITMO PARA ELIMINAR REGISTRO DE LIBRO.

Ilustración 14 Pseudocódigo eliminar registro libro

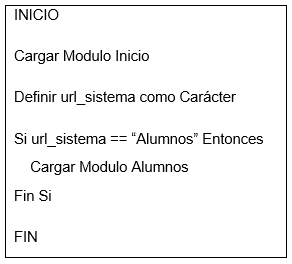
ALGORITMO PARA VISUALIZAR LISTA DE ALUMNOS.

Ilustración 15 Pseudocódigo visualización módulo alumnos

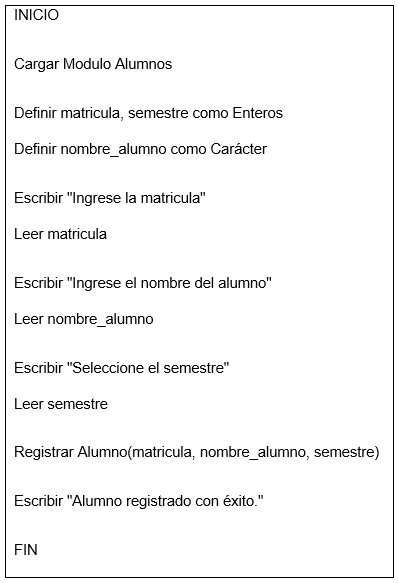
ALGORITMO PARA REGISTRAR ALUMNO.

Ilustración 16 Pseudocódigo registro de alumno

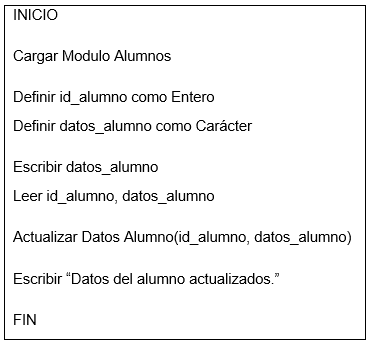
ALGORITMO PARA ACTUALIZAR DATOS DE ALUMNO.

Ilustración 17 Pseudocódigo actualizar datos alumno

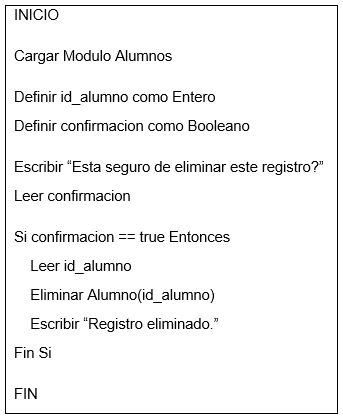
ALGORITMO PARA ELIMINAR REGISTRO DE ALUMNO.

Ilustración 18 Pseudocódigo eliminar registro alumno

**DIAGRAMAS DE FLUJO**

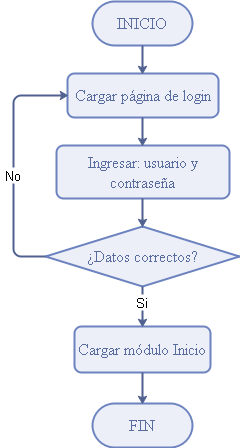
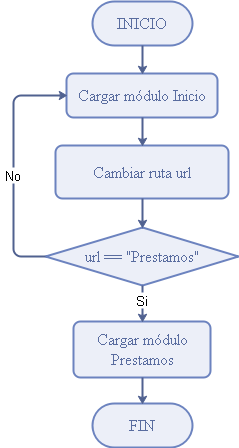
DIAGRAMA DE FLUJO PARA INICIO DE SESIÓN.

Ilustración 19 Diagrama de flujo 1

Ilustración 20 Diagrama de flujo 2

DIAGRAMA DE FLUJO PARA VISUALIZAR LISTA DE PRÉSTAMOS.

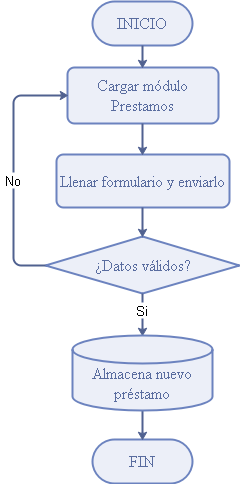
DIAGRAMA DE FLUJO PARA REGISTRAR PRÉSTAMO.

Ilustración 21 Diagrama de flujo 3

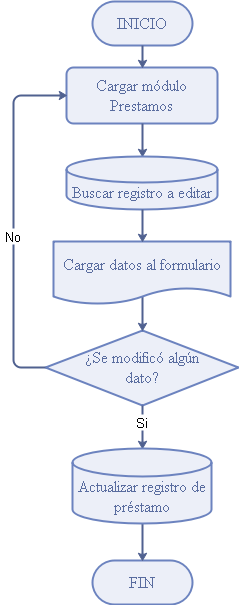
DIAGRAMA DE FLUJO PARA ACTUALIZAR ESTADO DE PRÉSTAMO.

Ilustración 22 Diagrama de flujo 4

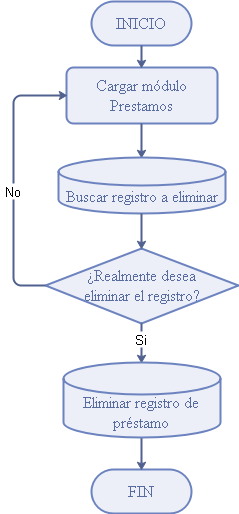
DIAGRAMA DE FLUJO PARA ELIMINAR REGISTRO DE PRÉSTAMO.

Ilustración 23 Diagrama de flujo 5

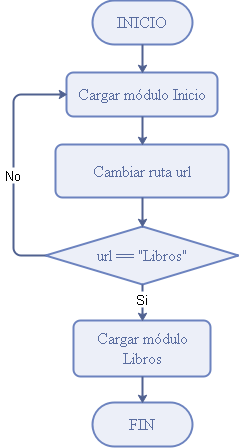
DIAGRAMA DE FLUJO PARA VISUALIZAR LISTA DE LIBROS.

Ilustración 24 Diagrama de flujo 6

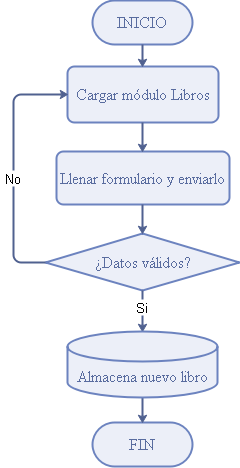
DIAGRAMA DE FLUJO PARA AGREGAR LIBRO.

Ilustración 25 Diagrama de flujo 7

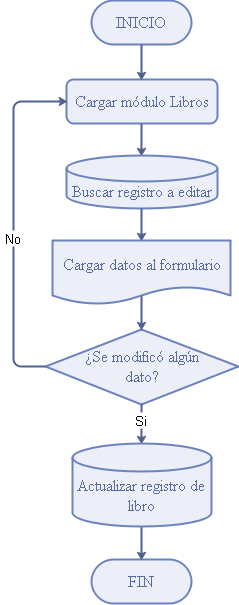
DIAGRAMA DE FLUJO PARA ACTUALIZAR DATOS DE LIBRO.

Ilustración 26 Diagrama de flujo 8

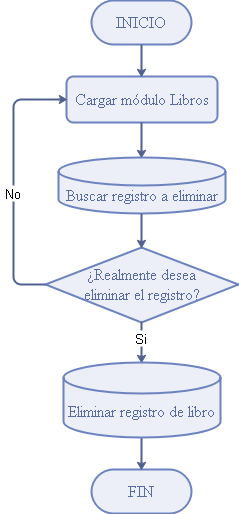
DIAGRAMA DE FLUJO PARA ELIMINAR REGISTRO DE LIBRO.

Ilustración 27 Diagrama de flujo 9

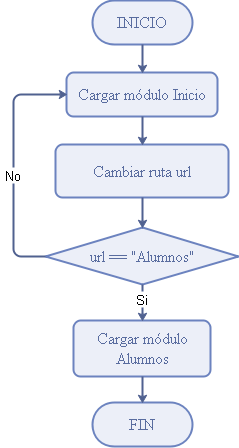
DIAGRAMA DE FLUJO PARA VISUALIZAR LISTA DE ALUMNOS.

Ilustración 28 Diagrama de flujo 10

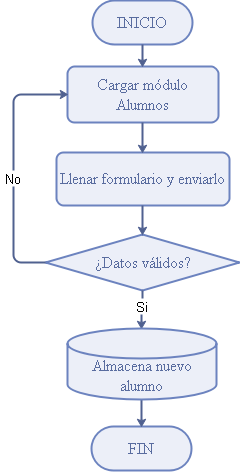
DIAGRAMA DE FLUJO PARA REGISTRAR ALUMNO.

Ilustración 29 Diagrama de flujo 11

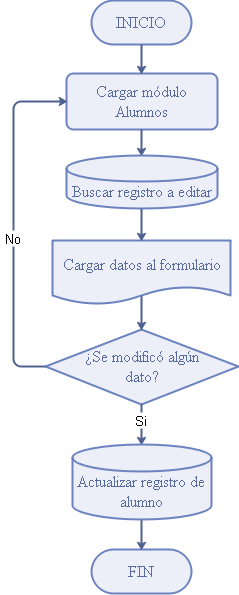
DIAGRAMA DE FLUJO PARA ACTUALIZAR DATOS DE ALUMNO.

Ilustración 30 Diagrama de flujo 12

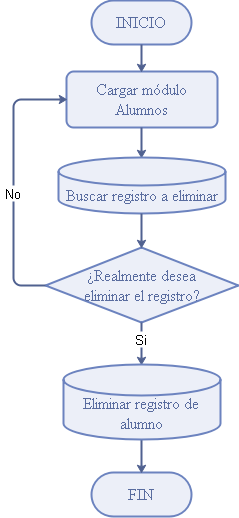
DIAGRAMA DE FLUJO PARA ELIMINAR REGISTRO DE ALUMNO.

Ilustración 31 Diagrama de flujo 13

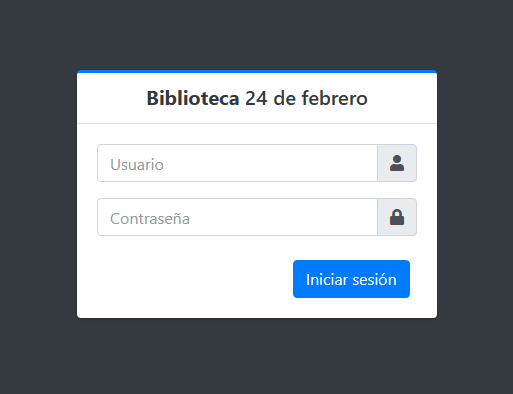
**DISEÑO DE INTERFACES**

Ilustración 32 Interfaz inicio de sesión

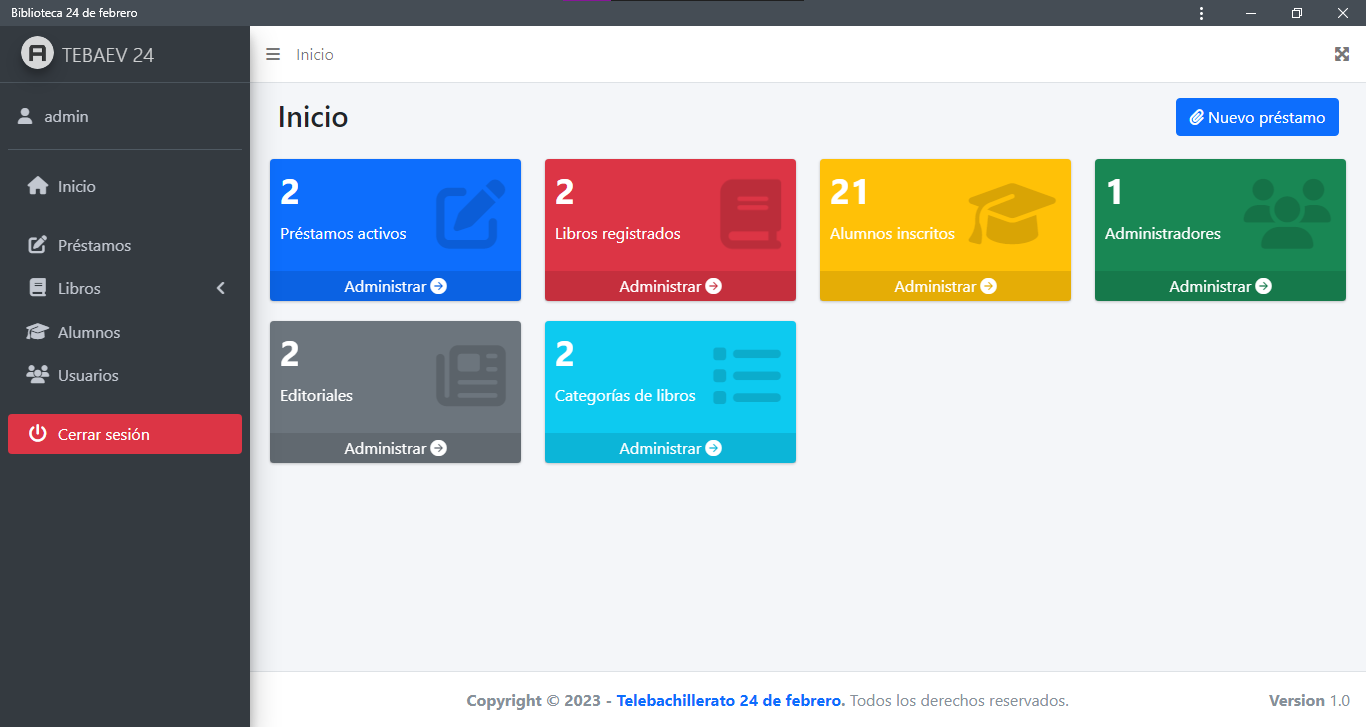


Ilustración 33 Interfaz principal

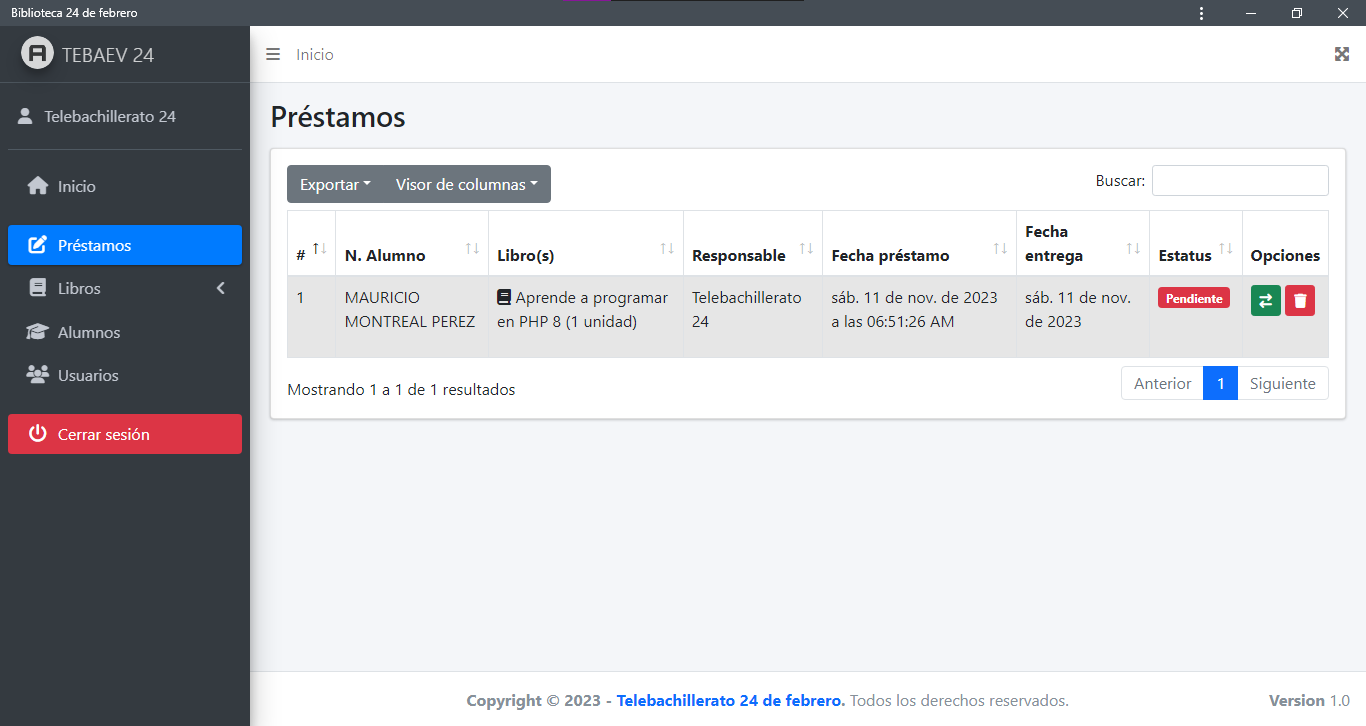


Ilustración 34 Interfaz del módulo préstamos

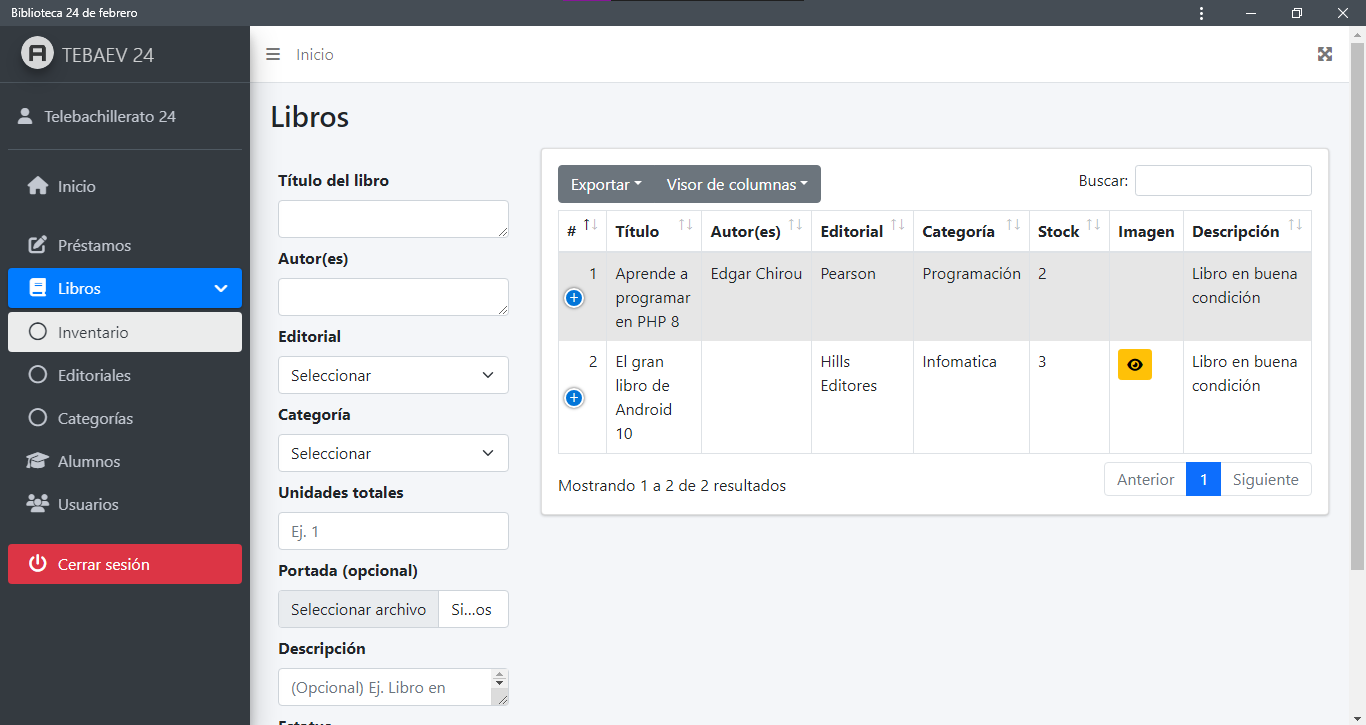


Ilustración 35Interfaz del módulo libros

**DISEÑO MODULAR**

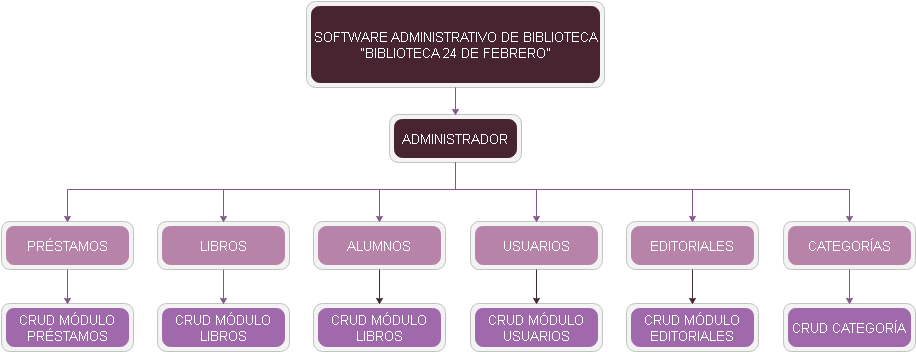


Ilustración 36 Diseño modular

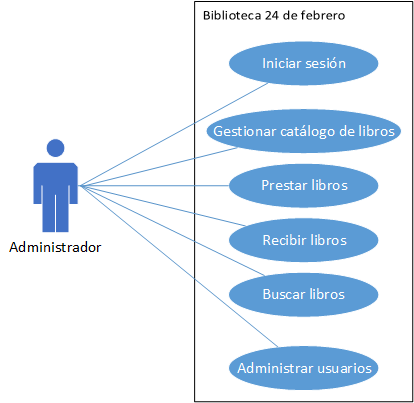
**DIAGRAMAS DE CASO DE USO**

Ilustración 37 Diagrama de casos de uso

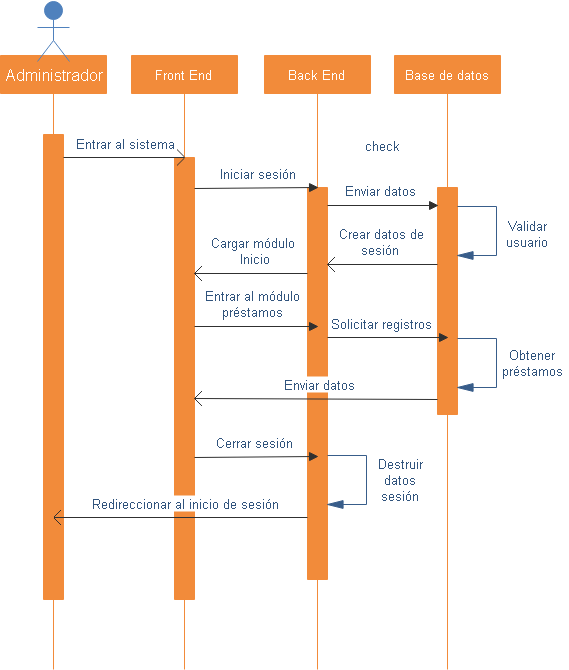
**DIAGRAMA DE SECUENCIA**

Ilustración 38 Diagrama de secuencia

**CORRECCIÓN DE ERRORES**

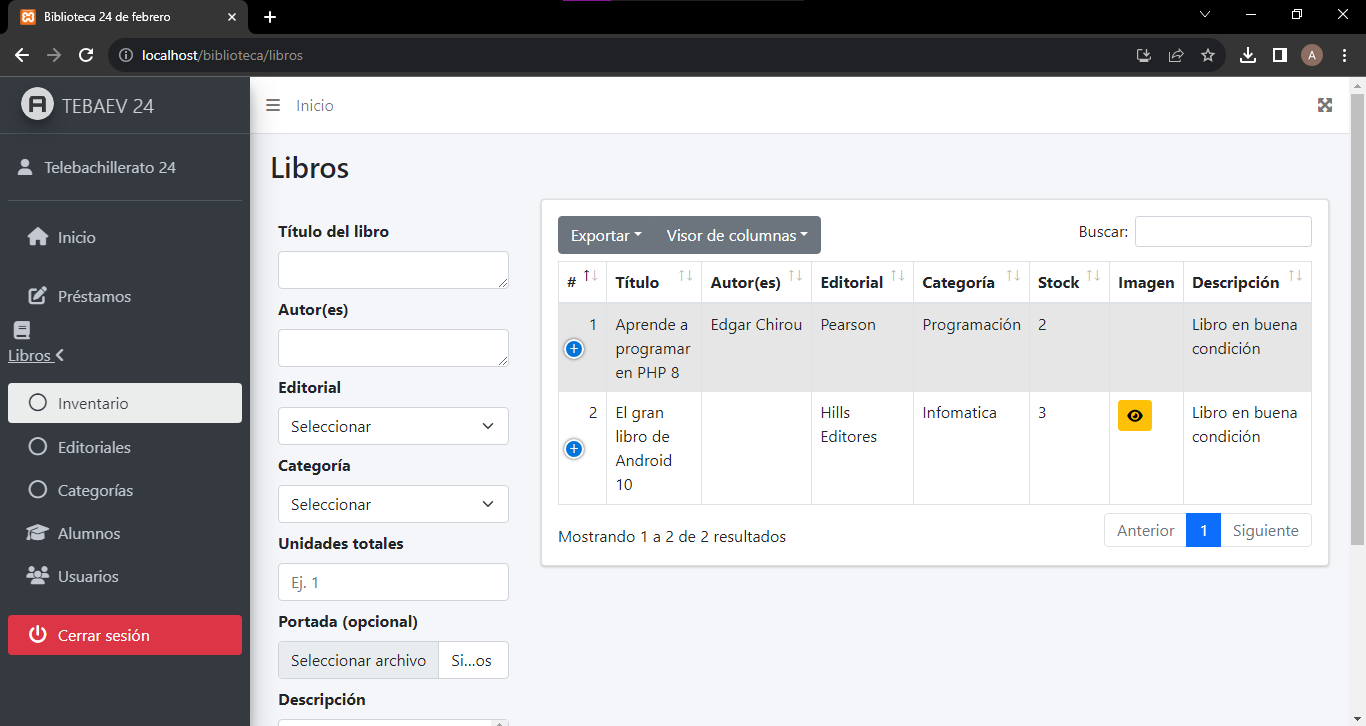


Ilustración 39 Error 1 barra lateral

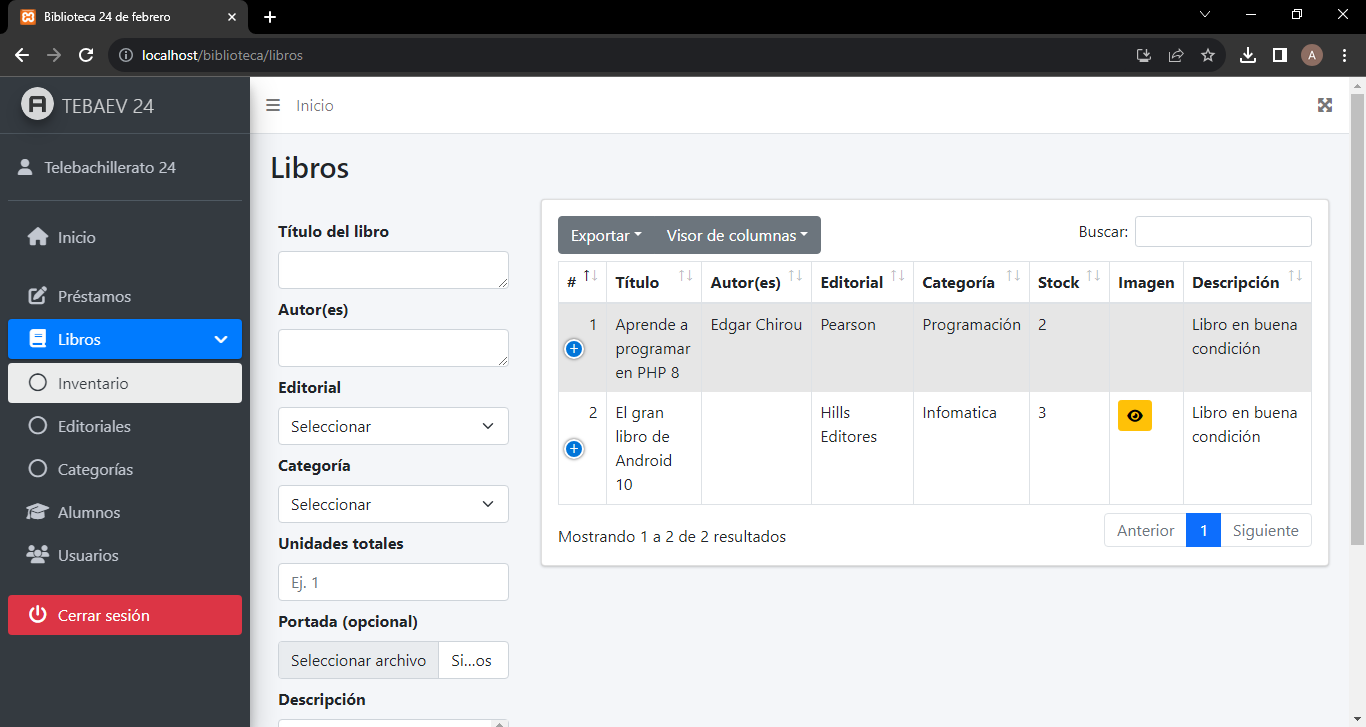


Ilustración 40 Corrección 1 barra lateral

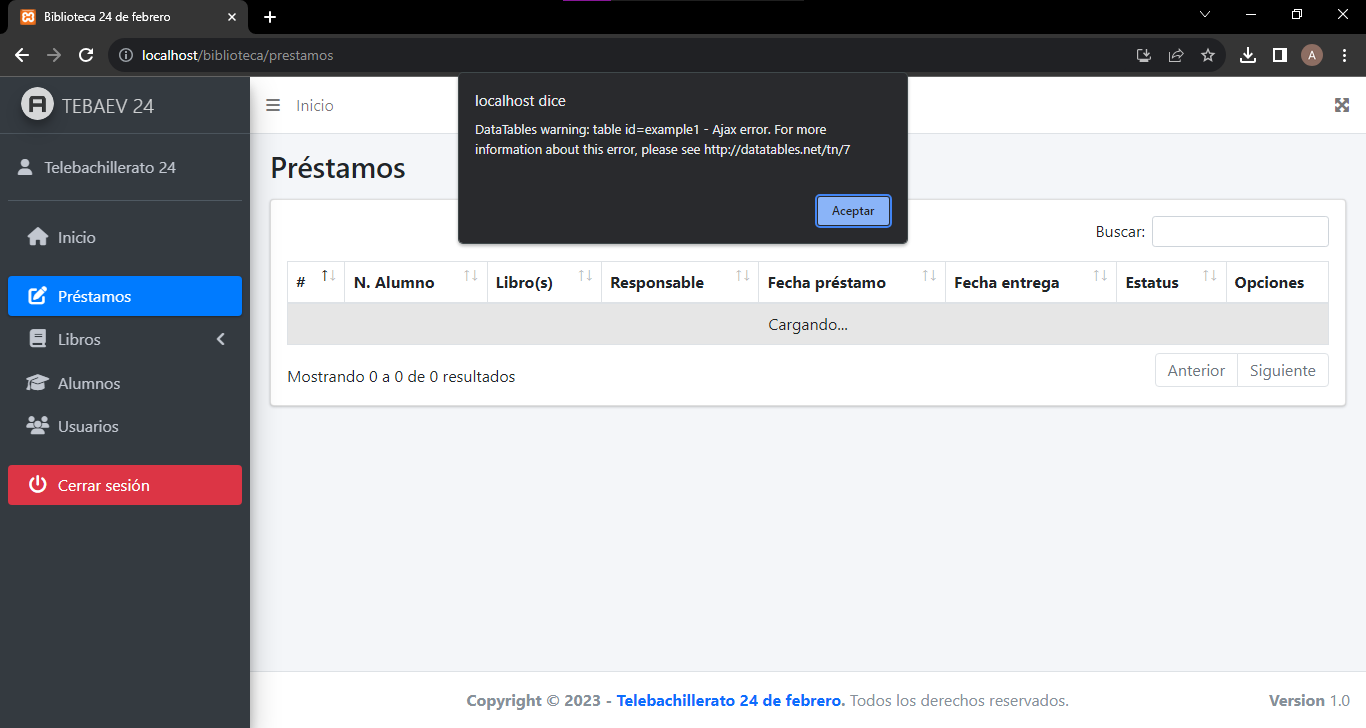


Ilustración 41 Error 2 interfaz prestamos

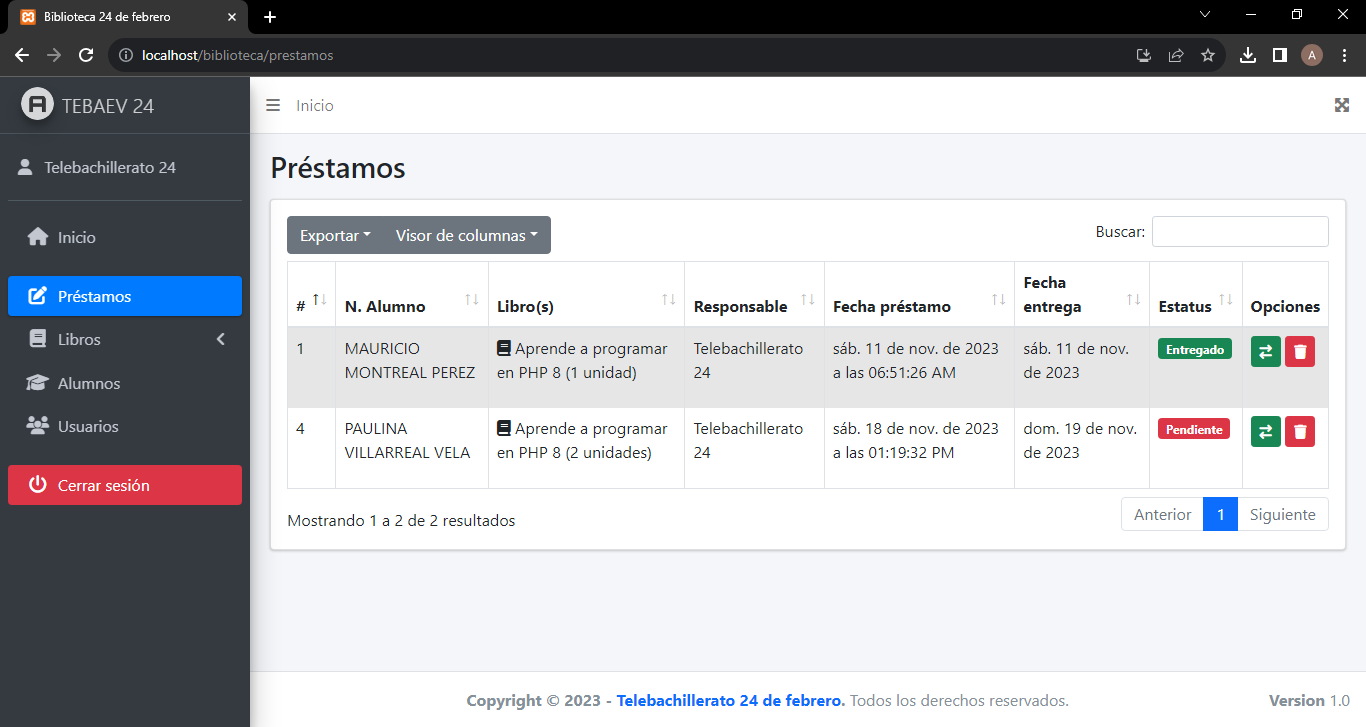


Ilustración 42 Corrección 2 interfaz prestamos

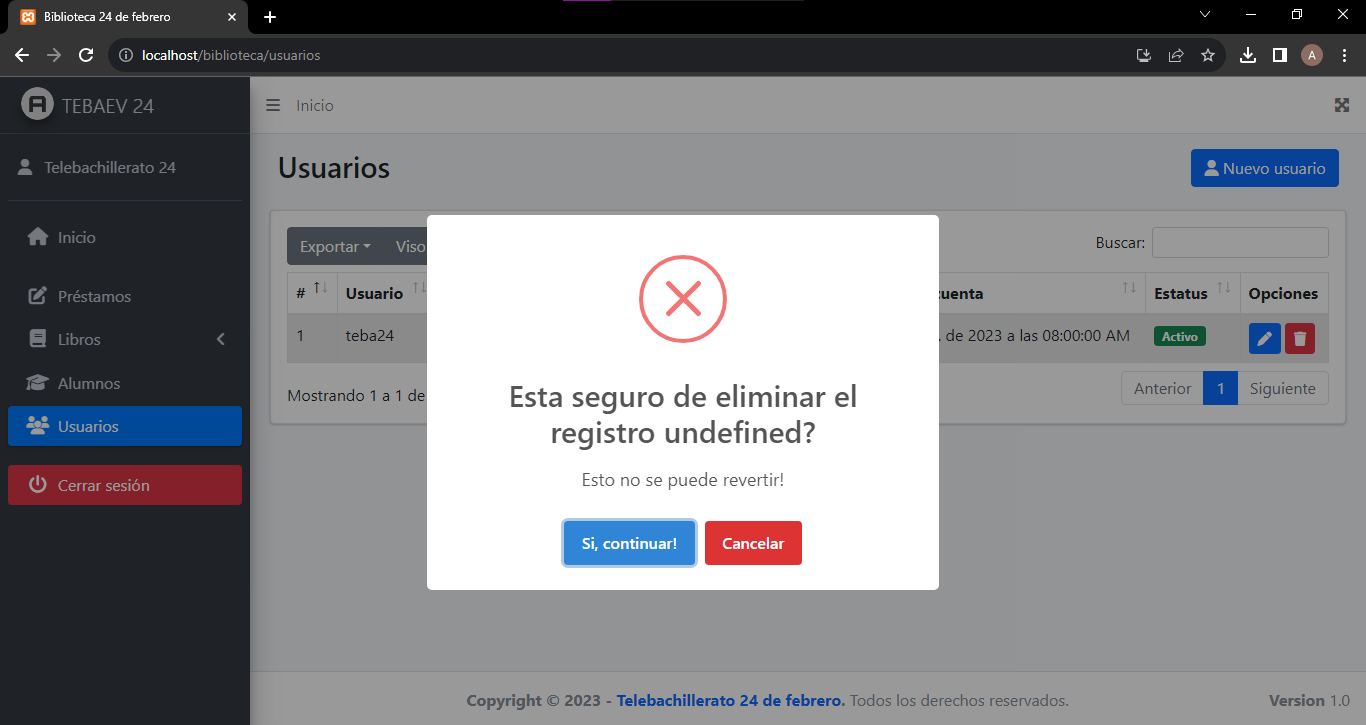


Ilustración 43 Error 3 módulo usuarios

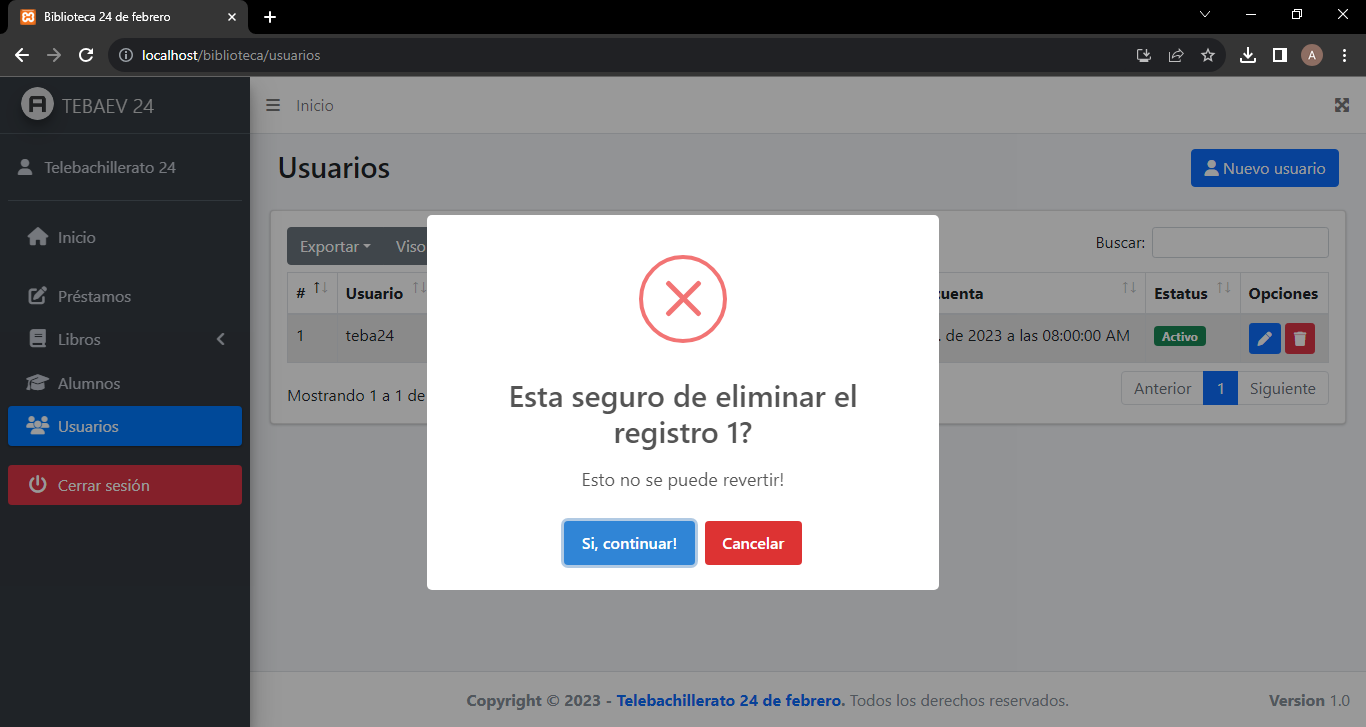


Ilustración 44 Corrección 3 módulo usuarios

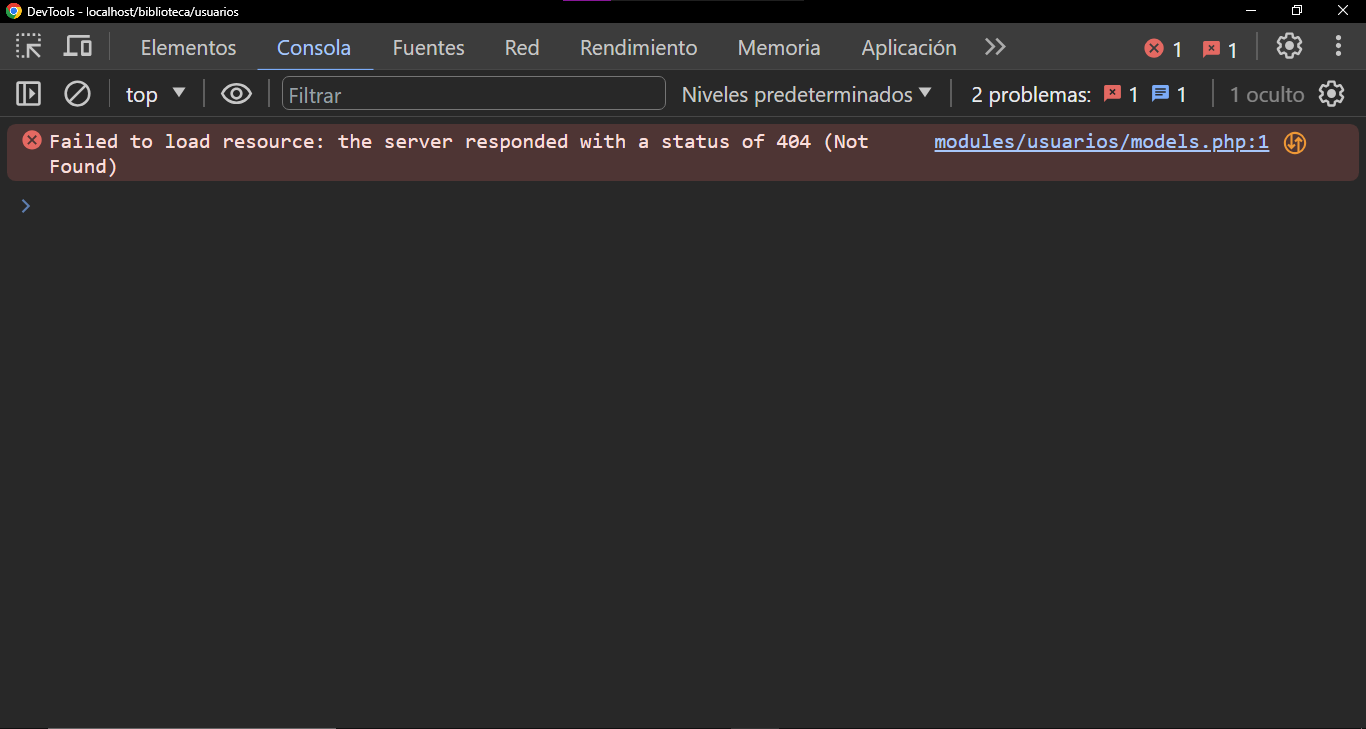


Ilustración 45 Error 4 módulo usuarios

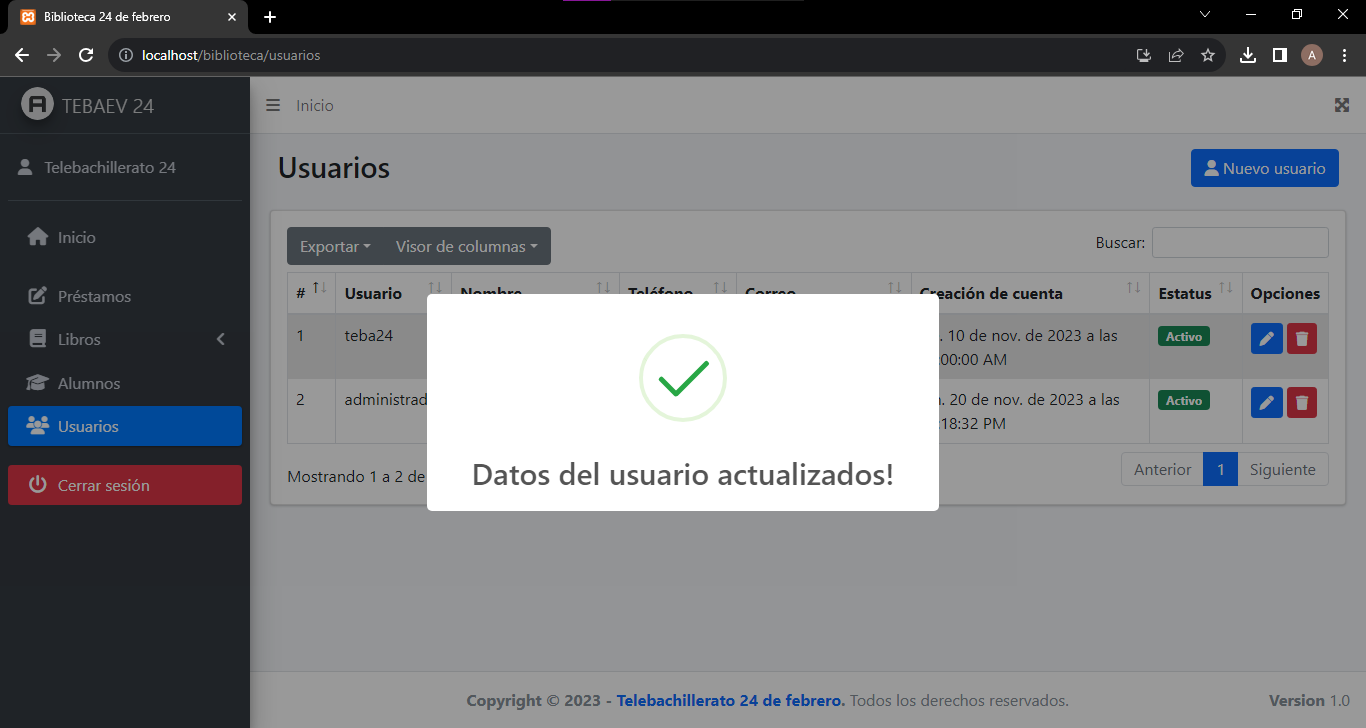


Ilustración 46 Corrección 4 módulo usuarios