# BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA COMPUTACIÓN

TESIS

## SISTEMA DE RECOMENDACIÓN DE LIBROS

PARA OBTENER EL TÍTULO DE
INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN

PRESENTADA POR

**MARTÍN GONZÁLEZ ESCAMILLA**

ASESOR

**M.C. MELIZA CONTRERAS GONZÁLEZ**

PUEBLA, PUE. ENERO 2013

# ÍNDICE

# ÍNDICE DE FIGURAS

# CAPÍTULO 1. INTRODUCCIÓN

## 1.1 Resumen.

Se planea crear un sistema que permita realizar altas, generar opiniones y hacer recomendaciones referentes a libros de distintas disciplinas y editoriales para así considerar su consulta, uso o compra de aquellos recomendados que cumplan con las necesidades del lector.

El sistema será una aplicación Web, a la cual podrá acceder el público en general sin importar si se encuentra registrado o no en la página, el público en general tendrá la posibilidad de realizar búsquedas sobre un libro en específico.

El público podrá hacer consultas de los libros, por tema, por autor, por título, así como por ISBN, los cuales estarán ordenados por el mayor número de recomendaciones obtenidas.

Para realizar la recomendación de un libro se debe estar registrado en el sistema, al estar registrado se convierte en parte de los usuarios del sistema, quienes aparte de tener la facultad de realizar recomendaciones pueden hacer opiniones de un libro y pueden administrar sus datos de cuenta.

Se creó una interfaz lo más amigable posible, ya que la página va dirigida al público general, por lo que se pretende que sea fácil de utilizar para cualquier persona ya sea un niño o un maestro, el único requisito para utilizar el sistema es que el usuario tenga conocimientos mínimos de computación. El usuario debe contar con una PC con acceso a internet y un navegador web, basta con escribir la dirección de la página web para empezar a utilizar el sistema.

## 1.2 Planteamiento del problema.

La intención de crear este sistema surge debido a la necesidad que presentan la mayoría de los estudiantes así como profesores, ya que muchas veces no se cuenta con la información necesaria y es entonces cuando recurrimos a los libros, pero muchas veces no sabemos cual consultar, cual contendrá la información que buscamos o cual será mas entendible para nosotros.

Este sistema pretende proporcionarle al usuario toda esta información, la cual no proviene de los autores ni de las editoriales, sino de los lectores mismos y con esto, ahorrar tiempo, dinero y esfuerzo en encontrar la información que el usuario necesita.

Hablando de libros electrónicos, los cuales ya se pueden comprar por internet, los lectores no pueden tener acceso a la información sin haberlos comprado, por lo que no pueden estar seguros si les servirá la información del libro, muchas veces

eso hace que los lectores descarten este tipo de compras. Mediante este sistema aquellos lectores pueden consultar la página y si les convence las opiniones y recomendaciones de los demás lectores pueden proceder a comprarlos vía internet.

En cuanto a libros de papel es necesario acudir a una biblioteca o librería para consultar su contenido y verificar libro por libro si contiene la información que requerimos lo cual toma muchísimo tiempo, por lo que con este sistema se agilizaría la búsqueda de los libros.

## 1.3 Objetivo general

El objetivo general de la presente tesis es proporcionar un sistema que permita realizar un seguimiento sobre libros de distintas disciplinas y editoriales para consulta y uso de los más recomendados.

### 1.3.1 Objetivos específicos.

Los objetivos específicos son:

1) Registrar el libro considerando el título, autor, ISBN, áreas relacionadas e idioma.

2) Registrar al usuario considerando nombre, cargo, teléfono de contacto o celular, correo electrónico, empresa, carrera, experiencia en el área.

3) Generar la recomendación del libro considerando en que cursos pueden ser empleados, aspectos relevantes de la obra, que elementos le faltan por cubrir.

4) Realizar historial de recomendaciones.

# CAPÍTULO 2. MARCO TEÓRICO

## 2.1. Ingeniería de software

La ingeniería de software es un área importante al crear sistemas, subsistemas, grupos y subgrupos en el contexto del Trabajo Colaborativo Soportado por Computadoras (CSCW). El campo de aplicación del CSCW es muy amplio. Una de las aplicaciones está dirigida a estudiar y dar soporte a los procesos de Ingeniería de Software, gestión de flujo de trabajo, soporte documental, toma de requisitos, coordinación de procesos, comunicación formal e informal dentro de la empresa, todo ello visto desde el punto de vista de la cooperación, en el que dos o más personas trabajan de forma conjunta para realizar una misma actividad [1].

En la construcción y desarrollo de proyectos se aplican métodos y técnicas para resolver los problemas, la informática aporta herramientas y procedimientos sobre los que se apoya la Ingeniería de Software:

- Mejoran la calidad de los productos de software.
- Aumentar la productividad y el trabajo de los ingenieros de software.
- Facilitar el control del proceso de desarrollo de software.
- Suministrar a los desarrolladores las bases para construir software de alta calidad en forma eficiente.
- Definir una disciplina que garantice la producción y el mantenimiento de los productos de software desarrollados en el plazo fijado y dentro del costo estimado.

Los métodos de la Ingeniería de Software indican cómo construir técnicamente el software, abarcan una gran gama de tareas que incluyen análisis de requisitos, diseño, construcción de programas, pruebas y mantenimiento. Los métodos de Ingeniería de Software dependen de un conjunto de principios básicos que gobiernan cada área de la tecnología e incluyen actividades de modelado y otras técnicas descriptivas.

Entre los objetivos que abarca la ingeniería de software están los siguientes: “Las cinco “C”:

Capacidad: las actividades de la organización están influenciadas por la capacidad de esta para procesar transacciones con rapidez y eficiencia. Los sistemas de información mejoran esta capacidad en tres formas:

- Aumentan la velocidad de procesamiento.
- Aumento en el volumen.
- Recuperación más rápida de la información.

Comunicación: la falta de comunicación es una fuente común de dificultades que afectan tanto al cliente como al empleado. Sin embargo, los sistemas de

información bien desarrollados amplían la comunicación y facilitan la integración de funciones individuales.

Costo: para determinar si la compañía evoluciona en la forma esperada, de acuerdo con lo presupuestado, se debe de llevar a cabo el seguimiento de los costos de mano de obra, bienes y gastos generales.

Control: este se obtiene al tomar en cuenta que debemos tener mayor seguridad de información y un menor margen de error, ya que algunas veces el hecho de que los datos pueden ser guardados de una manera adecuada para su lectura por medio de una máquina, es una seguridad difícil de alcanzar en un medio ambiente donde no existen computadoras.

Competitividad: los sistemas de información computacionales son un arma estratégica, capaces de cambiar la forma en que la compañía compite en el mercado, en consecuencias estos sistemas mejoran la organización y le ayudan a ganar “ventaja competitiva”.

## 2.2. UML

El Lenguaje de Modelado Unificado (LUM o UML, por sus siglas en inglés, Unified Modeling Language) es la sucesión de una serie de métodos de análisis y diseño orientadas a objetos que aparecen a fines de los 80's y principios de los 90s.

El lenguaje de modelado es la notación (principalmente gráfica) que usan los métodos para expresar un diseño. El proceso indica los pasos que se deben seguir para llegar a un diseño.

El Lenguaje Unificado de Modelado es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables [2].

Es importante resaltar que UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo.

Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software, pero no especifica en sí mismo qué metodología o proceso usar.

UML no puede compararse con la programación estructurada, pues UML significa Lenguaje Unificado de Modelado, no es programación, solo se diagrama la realidad de una utilización en un requerimiento. Mientras que, programación estructurada, es una forma de programar como lo es la orientación a objetos, sin embargo, la programación orientada a objetos viene siendo un complemento perfecto de UML, pero no por eso se toma UML sólo para lenguajes orientados a objetos.

### 2.2.1. Modelos de UML

Un modelo representa a un sistema software desde una perspectiva específica. Al igual que la planta y el alzado de una figura en dibujo técnico nos muestran la misma figura vista desde distintos ángulos, cada modelo nos permite fijarnos en un aspecto distinto del sistema.

En la figura 2.1 se muestran los diagramas definidos en UML.

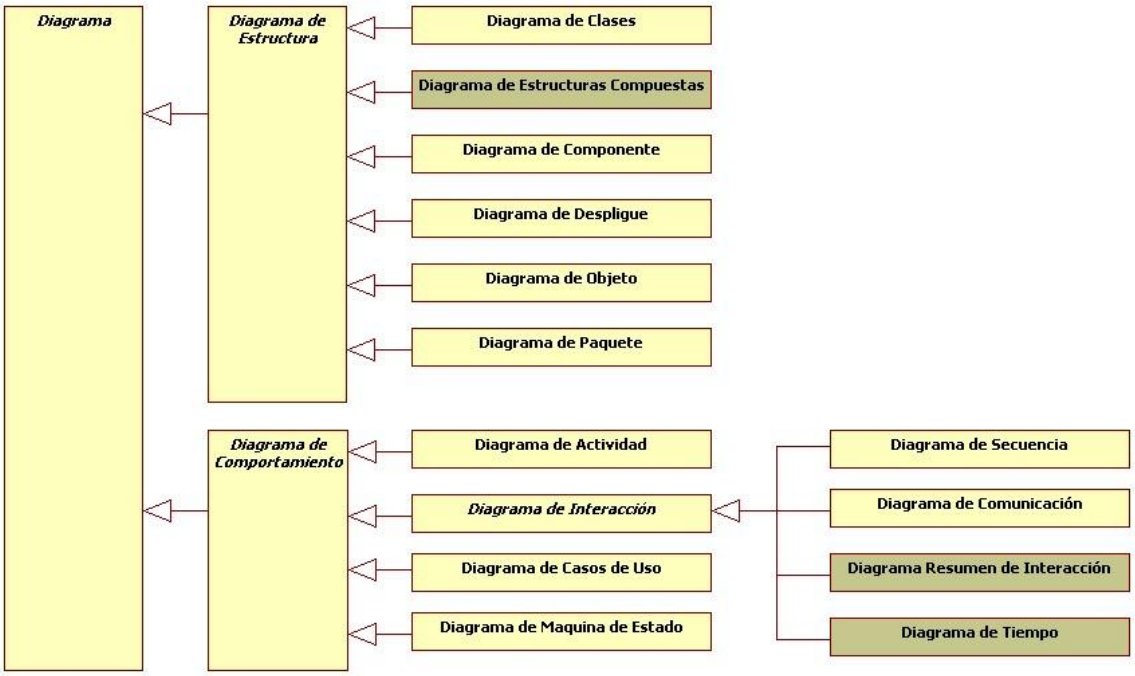


Figura 2.1 Esquema de Diagramas UML

Los modelos de UML que más comúnmente se utilizan son los siguientes:

- Diagrama de Estructura Estática.
- Diagrama de Casos de Uso.
- Diagrama de Secuencia.
- Diagrama de Estados.

## 2.3. Modelos de proceso de software

El modelo de ciclo de vida escogido es un factor principal para conseguir los objetivos buscados, una mala elección del ciclo de vida puede hacer que se retrase el trabajo enormemente o que se tenga una planificación perfecta del trabajo [3].

### 2.3.1. Modelo Lineal

En Ingeniería de software el desarrollo en cascada, también llamado modelo en cascada, es el enfoque metodológico que ordena rigurosamente las etapas del ciclo de vida del software, de forma tal que el inicio de cada etapa debe esperar a la finalización de la inmediatamente anterior [4]. Un ejemplo de una metodología de desarrollo en cascada es:

1. Análisis de requisitos
2. Diseño del Sistema
3. Diseño del Programa
4. Codificación
5. Pruebas
6. Implantación
7. Mantenimiento

De esta forma, cualquier error de diseño detectado en la etapa de prueba conduce necesariamente al rediseño y nueva programación del código afectado, aumentando los costos del desarrollo. La palabra *cascada* sugiere, mediante la metáfora de la fuerza de la gravedad, el esfuerzo necesario para introducir un cambio en las fases más avanzadas de un proyecto.

Si bien ha sido ampliamente criticado desde el ámbito académico y la industria, sigue siendo el paradigma más seguido al día de hoy.

Análisis de requisitos

> Se analizan las necesidades de los usuarios finales del software para determinar qué objetivos debe cubrir. De esta fase surge una memoria llamada SRD (documento de especificación de requisitos), que contiene la especificación completa de lo que debe hacer el sistema sin entrar en detalles internos.
>
> Es importante señalar que en esta etapa se deben consensuar todo lo que se requiere del sistema y será aquello lo que seguirá en las siguientes etapas, no pudiéndose requerir nuevos resultados a mitad del proceso de elaboración del software.

Diseño del Sistema

Se descompone y organiza el sistema en elementos que puedan elaborarse por separado, aprovechando las ventajas del desarrollo en equipo. Como resultado surge el SDD (Documento de Diseño del Software), que contiene la descripción de la estructura relacional global del sistema y la especificación de lo que debe hacer cada una de sus partes, así como la manera en que se combinan unas con otras.

Diseño del Programa

Es la fase en donde se realizan los algoritmos necesarios para el cumplimiento de los requerimientos del usuario así como también los análisis necesarios para saber que herramientas usar en la etapa de Codificación.

Codificación

Es la fase de programación o implementación propiamente dicha. Aquí se implementa el código fuente, haciendo uso de prototipos así como pruebas y ensayos para corregir errores.

Dependiendo del lenguaje de programación y su versión se crean las librerías y componentes reutilizables dentro del mismo proyecto para hacer que la programación sea un proceso mucho más rápido.

Pruebas

Los elementos, ya programados, se ensamblan para componer el sistema y se comprueba que funciona correctamente antes de ser puesto en explotación.

Implantación

El software obtenido se pone en producción. Se implantan los niveles software y hardware que componen el proyecto. La implantación es la fase con más duración y con más cambios en el ciclo de elaboración de un proyecto. Es una de las fases finales del proyecto.

Durante la explotación del sistema pueden surgir cambios, bien para corregir errores o bien para introducir mejoras. Todo ello se recopila en los Documentos de Cambios.

Variantes

Existen variantes de este modelo; especialmente destacamos la que hace uso de prototipos y en la que se establece un ciclo antes de llegar a la fase de mantenimiento, verificando que el sistema final este libre de fallos.

Desventajas

En la vida real, un proyecto rara vez sigue una secuencia lineal, esto crea una mala implementación del modelo, lo cual hace que lo lleve al fracaso.

Difícilmente un cliente va a establecer al principio todos los requerimientos necesarios, por lo que provoca un gran atraso trabajando en este modelo, ya que este es muy restrictivo y no permite movilizarse entre fases.

Los resultados y/o mejoras no son visibles, el producto se ve recién cuando este esté finalizado, lo cual provoca una gran inseguridad por parte del cliente que anda ansioso de ver avances en el producto. Esto también implica toparse con requerimientos que no se habían tomado en cuenta, y que surgieron al momento de la implementación, lo cual provocara que se regrese nuevamente a la fase de requerimientos.

### 2.3.2. Modelo en Espiral

El Desarrollo en Espiral es un modelo de ciclo de vida desarrollado por Barry Boehm en 1985, utilizado generalmente en la Ingeniería de software. Las actividades de este modelo son una espiral, cada bucle es una actividad como se puede apreciar en la figura 2.2. Las actividades no están fijadas a prioridad, sino que las siguientes se eligen en función del análisis de riesgo, comenzando por el bucle interior [5].

Para cada actividad habrá cuatro tareas:

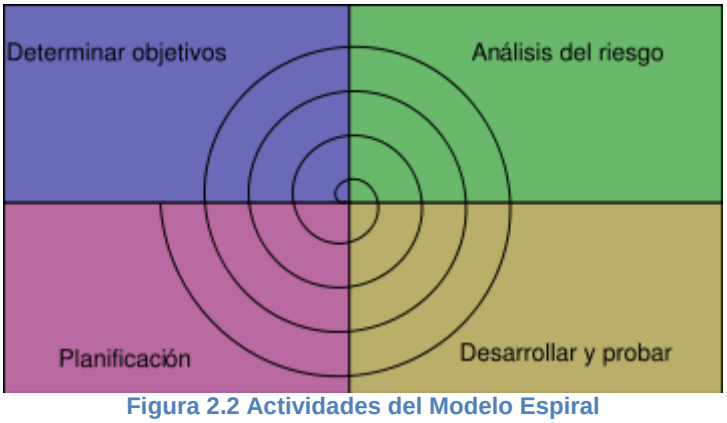


Figura 2.2 Actividades del Modelo Espiral

Determinar o fijar objetivos

- Fijar también los productos definidos a obtener: requerimientos, especificación, manual de usuario.
- Fijar las restricciones.
- Identificación de riesgos del proyecto y estrategias alternativas para evitarlos.
- Hay una cosa que solo se hace una vez: planificación inicial o previa.

Análisis del riesgo

- Se estudian todos los riesgos potenciales y se seleccionan una o varias alternativas propuestas para reducir o eliminar los riesgos.

Desarrollar y probar

- Tareas de la actividad propia y de prueba.
- Análisis de alternativas e identificación resolución de riesgos.
- Dependiendo del resultado de la evaluación de los riesgos, se elige un modelo para el desarrollo, el que puede ser cualquiera de los otros existentes, como formal, evolutivo, cascada, etc. Así si por ejemplo si los riesgos en la interfaz de usuario son dominantes, un modelo de desarrollo apropiado podría ser la construcción de prototipos evolutivos. Si lo riesgos de protección son la principal consideración, un desarrollo basado en transformaciones formales podría ser el más apropiado.

Planificar

- Revisamos todo lo hecho, evaluándolo, y con ello decidimos si continuamos con las fases siguientes y planificamos la próxima actividad.

Mecanismos de control

- La dimensión radial mide el costo.
- La dimensión angular mide el grado de avance del proyecto.

Ventajas

- El análisis del riesgo se hace de forma explícita y clara. Une los mejores elementos de los restantes modelos. - Reduce riesgos del proyecto - Incorpora objetivos de calidad - Integra el desarrollo con el mantenimiento, etc. Además es posible tener en cuenta mejoras y nuevos requerimientos sin romper con la metodología, ya que este ciclo de vida no es rígido ni estático.

Desventajas

- Genera mucho tiempo en el desarrollo del sistema - Modelo costoso - Requiere experiencia en la identificación de riesgos [6].

### 2.3.3. Modelo de Prototipos

Este modelo consiste en un procedimiento que permite al equipo de desarrollo diseñar y analizar una aplicación que representa el sistema que será implementado. Dicha aplicación llamada prototipo, está compuesta por los componentes que se desean evaluar. Las etapas del modelo son:

- Investigación preliminar.
- Recolección y refinamiento de los requisitos y proyecto rápido.
- Diseño técnico.
- Diseño y construcción del prototipo.
- Evaluación del prototipo por el cliente.
- Renacimiento del prototipo.

El empleo de prototipos para el desarrollo de software es útil para comunicar, discutir y definir las ideas de los diseñadores y las partes responsables para poder formar un ciclo como se puede apreciar en la figura 2.3. Es frecuente que los clientes no sepan lo que quieren, pero cuando ven algo y lo utilizan, pronto saben lo que no quieren.

Los prototipos responden a preguntas y apoyan al trabajo de los diseñadores probando ideas, clarificando requisitos o definiendo alternativas.

Con el modelo de prototipos se comienza diseñando y construyendo las partes más importantes de la aplicación en un prototipo que posteriormente se refinará y ampliará hasta que el prototipo se termine.

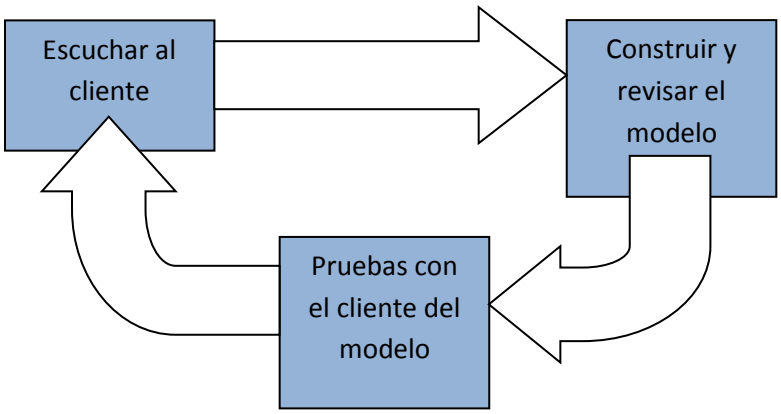


**Figura 2.3 Ciclo de Vida del Modelo de Prototipos**

### 2.3.4. Proceso Unificado de Desarrollo del Software

El Proceso Unificado es un proceso de desarrollo de software: “conjunto de actividades necesarias para transformar los requisitos del usuario en un sistema software” [7].

- El Proceso Unificado de Rational (Rational Unified Process en inglés, habitualmente resumido como RUP) es un marco genérico que puede especializarse para una variedad de tipos de sistemas, diferentes áreas de aplicación, tipos de organizaciones, niveles de aptitud y diferentes tamaños de proyectos.
- RUP está basado en componentes. El software está formado por componentes software interconectados a través de interfaces.
- RUP está dirigido por casos de uso, centrado en la arquitectura, y es iterativo e incremental.

Dirigido por casos de uso

- Un caso de uso es un fragmento de funcionalidad del sistema que proporciona un resultado de valor a un usuario. Los casos de uso modelan los requerimientos funcionales del sistema.

- Todos los casos de uso juntos constituyen el modelo de casos de uso.

- Los casos de uso también guían el proceso de desarrollo (diseño, implementación, y prueba). Basándose en los casos de uso los desarrolladores crean una serie de modelos de diseño e implementación que llevan a cabo los casos de uso. De este modo los casos de uso no solo inician el proceso de desarrollo sino que le proporcionan un hilo conductor, avanza a través de una serie de flujos de trabajo que parten de los casos de uso.

Centrado en la arquitectura

La arquitectura de un sistema software se describe mediante diferentes vistas del sistema en construcción. El concepto de arquitectura software incluye los aspectos estáticos y dinámicos más significativos del sistema. La arquitectura es una vista del diseño completo con las características más importantes resaltadas, dejando los detalles de lado.

Arquitectura

Conjunto de decisiones significativas acerca de la organización de un sistema software, la selección de los elementos estructurales a partir de los cuales se compone el sistema, las interfaces entre ellos, su comportamiento, sus colaboraciones, y su composición.

Los casos de uso y la arquitectura están profundamente relacionados. Los casos de uso deben encajar en la arquitectura, y a su vez la arquitectura debe permitir el desarrollo de todos los casos de uso requeridos, actualmente y a futuro.

El arquitecto desarrolla la forma o arquitectura a partir de la comprensión de un conjunto reducido de casos de uso fundamentales o críticos (usualmente no más del
10 % del total). En forma resumida, podemos decir que el arquitecto:

- Crea un esquema en borrador de la arquitectura comenzando por la parte no específica de los casos de uso (por ejemplo la plataforma) pero con una comprensión general de los casos de uso fundamentales.

- A continuación, trabaja con un conjunto de casos de uso, claves o fundamentales. Cada caso de uso es especificado en detalle y realizado en términos de subsistemas, clases, y componentes.

- A medida que los casos de uso se especifican y maduran, se descubre más de la arquitectura, y esto a su vez lleva a la maduración de más casos de uso. Este proceso continúa hasta que se considere que la arquitectura es estable.

Iterativo e Incremental

Es práctico dividir el esfuerzo de desarrollo de un proyecto de software en partes más pequeñas o mini proyectos. Cada mini proyecto es una iteración que resulta en un incremento. Las iteraciones hacen referencia a pasos en el flujo de trabajo, y los incrementos a crecimientos en el producto. Las iteraciones deben estar controladas. Esto significa que deben seleccionarse y ejecutarse de una forma planificada.

Los desarrolladores basan la selección de lo que implementarán en cada iteración en dos cosas: el conjunto de casos de uso que amplían la funcionalidad, y en los riesgos más importantes que deben mitigarse.

En cada iteración los desarrolladores identifican y especifican los casos de uso relevantes, crean un diseño utilizando la arquitectura seleccionada como guía, para implementar dichos casos de uso. Si la iteración cumple sus objetivos, se continúa con la próxima. Si no deben revisarse las decisiones previas y probar un nuevo enfoque.

Beneficios del enfoque iterativo

- La iteración controlada reduce el riesgo a los costes de un solo incremento.
- Reduce el riesgo de retrasos en el calendario atacando los riesgos más importantes primero.
- Acelera el desarrollo. Los trabajadores trabajan de manera más eficiente al obtener resultados a corto plazo.

- Tiene un enfoque más realista al reconocer que los requisitos no pueden definirse completamente al principio.

### 2.3.5. Modelo Incremental

Este modelo combina elementos del modelo de cascada (aplicados repetitivamente) con la filosofía interactiva de construcción de prototipo. El primer incremento es un producto esencial (núcleo), se afrontan requisitos básicos y muchas funciones extras (conocidas o no) quedan para los siguientes incrementos [8].

El cliente usa el producto central y en base a la utilización y/o evolución se desarrolla un plan para el incremento siguiente. Este proceso se repite hasta que se elabore el producto completo.

Es interactivo, al igual que el de construcción de prototipos y otros enfoques evolutivos. Pero a diferencia del modelo de construcción de prototipos, el modelo incremental entrega un producto operacional en cada incremento (véase figura 2.4).

Es útil cuando la dotación de personal no está disponible para una implementación completa. El primer incremento se puede realizar con pocas personas. Si el producto central es aceptado, se puede recibir a más personal.

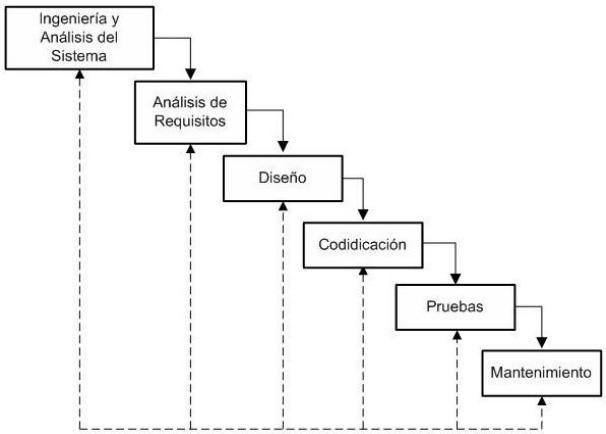


**Figura 2.4 Diagrama del Modelo Incremental**

## 2.4. Bases de datos

Una base de datos es un almacén que permite guardar grandes cantidades de información de forma organizada para que luego se utilice fácilmente. El término de base de datos fue escuchado por primera vez en 1963, en un simposio celebrado en California, USA. Una base de datos se puede definir como un conjunto de información relacionada que se encuentra agrupada o estructurada.

Desde el punto de vista informático, las bases de datos es un sistema formado por un conjunto de datos almacenados en un disco que permiten el acceso directo a ellos y un conjunto de programas que manipulan ese conjunto de datos [9].

Cada base de datos se compone de una o más tablas que guardan un conjunto de datos. Cada tabla tiene una o más columnas y filas. Las columnas guardan información sobre cada elemento que queremos guardar en la tabla, cada fila de la tabla conforma un registro.

### 2.4.1. Introducción y definición de Bases de Datos

Las bases de datos no son un fenómeno nuevo. De hecho, Herman Hollerith mecanizó el almacenamiento del Censo de EEUU de 1890 en lo que de alguna forma se considera la primera base de datos significativa “computarizada”. En la actualidad, y gracias al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos tienen formato electrónico, que ofrece un amplio rango de soluciones al problema de almacenar datos.

En informática existen los sistemas gestores de bases de datos (SGBD), que permiten almacenar y posteriormente acceder a los datos de forma rápida y estructurada. Las propiedades de los sistemas gestores de bases de datos se estudian en informática.

Las aplicaciones más usuales son para la gestión de empresas e instituciones públicas. También son ampliamente utilizadas en entornos científicos con el objeto de almacenar la información experimental.

Aunque las bases de datos pueden contener muchos tipos de datos, algunos de ellos se encuentran protegidos por las leyes de varios países. Por ejemplo en España, los datos personales se encuentran protegidos por la Ley Orgánica de Protección de Datos de Carácter Personal.

### 2.4.2. Sistemas de Gestión de Bases de Datos

Los Sistemas de Gestión de Bases de Datos son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta [10].
Consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a dichos datos. La colección de datos, normalmente denominada base de datos, contiene información relevante para una empresa. El objetivo principal de una SGBD es proporcionar una forma de almacenar y recuperar la información de una base de datos de manera que sea tanto practica como eficiente.

Los sistemas de bases de datos se diseñan para gestionar grandes cantidades de información. La gestión de datos implica tanto la definición de estructuras para almacenar la información como proporcionar mecanismos para la manipulación de la información. Además, los sistemas de bases de datos deben de garantizar la fiabilidad de la información almacenada, a pesar de la caída del sistema o de los accesos no autorizados. Si los datos van a ser compartidos entre diferentes usuarios, el sistema debe evitar posibles resultados anormales.

Existen distintos objetivos que deben cumplir los SGBD:

- Abstracción de la información. Los SGBD ahorran a los usuarios detalles acerca del almacenamiento físico de los datos. Da lo mismo si una base de datos ocupa uno o cientos de archivos, este hecho se hace transparente al usuario. Así, se definen varios *niveles de abstracción*.

- Independencia. La independencia de los datos consiste en la capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.

- Consistencia. En aquellos casos en los que no se ha logrado eliminar la redundancia, será necesario vigilar que aquella información que aparece repetida se actualice de forma coherente, es decir, que todos los datos repetidos se actualicen de forma simultánea. Por otra parte, la base de datos representa una realidad determinada que tiene determinadas condiciones, por ejemplo que los menores de edad no pueden tener licencia de conducir. El sistema no debería aceptar datos de un conductor menor de edad. En los SGBD existen herramientas que facilitan la programación de este tipo de condiciones.

- Seguridad. La información almacenada en una base de datos puede llegar a tener un gran valor. Los SGBD deben garantizar que esta información se encuentra segura de permisos a usuarios y grupos de usuarios, que permiten otorgar diversas categorías de permisos.

- Manejo de Transacciones. Una transacción es un programa que se ejecuta como una sola operación. Esto quiere decir que luego de una ejecución en la que se produce una falla es el mismo que se obtendría si el programa no se hubiera ejecutado. Los SGBD proveen mecanismos para programar las modificaciones de los datos de una forma mucho más simple que si no se dispusiera de ellos.

- Tiempo de respuesta. Lógicamente, es deseable minimizar el tiempo que el SGBD tarda en darnos la información solicitada y en almacenar los cambios realizados.

## 2.5. Modelo de bases de datos

Un modelo de base de datos o esquema de base de datos es la estructura o el formato de una base de datos, descrita en un lenguaje formal soportada por el sistema de gestión de bases de datos. En otras palabras, un "modelo de base de datos" es la aplicación de un modelo de datos usado en conjunción con un sistema de gestión de bases de datos.

Los esquemas generalmente son almacenados en un diccionario de datos. Aunque un esquema se defina en un lenguaje de base de datos de texto, el término a menudo es usado para referirse a una representación gráfica de la estructura de la base de datos.

### 2.5.1. Modelo de Datos

Este modelo es el más utilizado actualmente, ya que utiliza tablas bidimensionales para la representación lógica de los datos y sus relaciones.

Algunas de sus principales características son:

- Puede ser entendido y usado por cualquier usuario.
- Permite ampliar el esquema conceptual sin modificar las aplicaciones de gestión.
- Los usuarios no necesitan saber donde se encuentran los datos físicamente.

El elemento principal de este modelo es la relación que se representa mediante una tabla.

### 2.5.2. Modelo Entidad- Relación (E-R)

El modelo de datos E-R se basa en una percepción del mundo real que consiste en una colección de objetos básicos, denominados entidades, y de las relaciones entre ellos. Este modelo se desarrolló para facilitar el diseño de bases de datos permitiendo la especificación de un esquema de la empresa que representa la estructura lógica global de la base de datos [11].

El modelo de datos E-R es uno de los diferentes modelos de datos semánticos; el aspecto semántico del modelo radica en la representación del significado de los datos. El modelo E-R resulta muy útil para relacionar los significados e interacciones de las empresas reales con el esquema conceptual. Debido a esta utilidad, muchas herramientas de diseño de bases de datos se basan en los conceptos del modelo E-R.
El modelo de datos E-R emplea tres conceptos básicos: los conjuntos de entidades, los conjuntos de relaciones y los atributos.

Llamándose entidad a una “cosa” u “objeto” del mundo real que es distinguible de otros objetos. Por ejemplo, cada persona de una empresa es una entidad. Una entidad tiene un conjunto de propiedades, y los valores de algún conjunto de propiedades pueden identificar cada entidad de forma unívoca.

Cada entidad se representa mediante un conjunto de atributos, los cuales son propiedades descriptivas que posee cada miembro de un conjunto de entidades.

Una relación es una asociación entre varias entidades.

### 2.5.3. Modelo Relacional

Éste es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Tras ser postulados sus fundamentos en 1970 por Edgar Frank Codd, de los laboratorios IBM en San José (California), no tardó en consolidarse como un nuevo paradigma en los modelos de base de datos. Su idea fundamental es el uso de "relaciones" [12].
Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados "tuplas". Pese a que ésta es la teoría de las bases de datos relacionales creadas por Edgar Frank Codd, la mayoría de las veces se conceptualiza de una manera más fácil de imaginar.

Esto es pensando en cada relación como si fuese una tabla que está compuesta por registros (las filas de una tabla), que representarían las tuplas, y campos (las columnas de una tabla).

En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario esporádico de la base de datos. La información puede ser recuperada o almacenada mediante "consultas" que ofrecen una amplia flexibilidad y poder para administrar la información.

El lenguaje más habitual para construir las consultas a bases de datos relacionales es SQL, Structured Query Language o Lenguaje Estructurado de Consultas, un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales.

Durante su diseño, una base de datos relacional pasa por un proceso al que se le conoce como normalización de una base de datos.

### 2.5.4. Llaves Primarias y Foráneas

En el diseño de bases de datos relacionales, se llama Llave Primaria a un campo o a una combinación de campos que identifica de forma única a cada fila de una

tabla. Una clave primaria comprende de esta manera una columna o conjunto de columnas. No puede haber dos filas en una tabla que tengan la misma clave primaria.

Una clave primaria debe identificar unívocamente a todas las posibles filas de una tabla y no solo a las filas que se encuentran en un momento determinado. Ejemplos de claves primarias son DNI (asociado a una persona) o ISBN (asociado a un libro). Las guías telefónicas y diccionarios no pueden usar nombres o palabras o números del sistema decimal de Dewey como claves candidatas, porque no identifican unívocamente números de teléfono o palabras.

Una clave primaria es un caso especial de clave única. La mayor diferencia es que para claves únicas, no se impone automáticamente la restricción implícita NOT NULL, mientras que para claves primarias, sí. Así, los valores en columnas de clave única pueden o no ser NULL. Otra diferencia es que las claves primarias deben definirse por medio de otra sintaxis.

El modelo relacional, según se lo expresa mediante cálculo relacional y álgebra relacional, no distingue entre clave primaria y otros tipos de claves. Las claves primarias fueron agregadas al estándar SQL principalmente para conveniencia del programador.

Tanto claves únicas como claves primarias pueden referenciarse con claves foráneas.

Llave Foránea

En el contexto de bases de datos relacionales, una Llave Foránea o Llave Ajena (o Foreign Key FK) es una limitación referencial entre dos tablas. La clave foránea identifica una columna o grupo de columnas en una tabla (tabla hija o referendo) que se refiere a una columna o grupo de columnas en otra tabla (tabla maestra o referenciada). Las columnas en la tabla referendo deben ser la clave primaria u otra clave candidata en la tabla referenciada.

Los valores en una fila de las columnas referendo deben existir solo en una fila en la tabla referenciada. Así, una fila en la tabla referendo no puede contener valores que no existen en la tabla referenciada. De esta forma, las referencias pueden ser creadas para vincular o relacionar información. Esto es una parte esencial de la normalización de base de datos. Múltiples filas en la tabla referendo pueden hacer referencia, vincularse o relacionarse a la misma fila en la tabla referenciada. Mayormente esto se ve reflejado en una relación uno (tabla maestra o referenciada) a muchos (tabla hija o referendo).

## 2.6. Normalización

La normalización es el proceso mediante el cual se transforman datos complejos a un conjunto de estructuras de datos más pequeñas, que además de ser más simples y más estables, son más fáciles de mantener [13].

El proceso de normalización se puede entender como una serie de reglas que sirven para ayudar a los diseñadores de bases de datos a desarrollar un esquema que minimice los problemas de lógica.

### 2.6.1. Definición de Normalización

La normalización es el proceso de organizar los datos de una base de datos. Se incluye la creación de tablas y el establecimiento de relaciones entre ellas según reglas diseñadas tanto para proteger los datos como para hacer que la base de datos sea más flexible al eliminar la redundancia y las dependencias incoherentes.

Los datos redundantes desperdician el espacio de disco y crean problemas de mantenimiento. Si hay que cambiar datos que existen en más de un lugar, se deben cambiar de la misma forma exactamente en todas sus ubicaciones. Un cambio en la dirección de un cliente es mucho más fácil de implementar si los datos sólo se almacenan en la tabla Clientes y no en algún otro lugar de la base de datos.

Aunque es intuitivo para un usuario mirar en la tabla Clientes para buscar la dirección de un cliente en particular, puede no tener sentido mirar allí el salario del empleado que llama a ese cliente. El salario del empleado está relacionado con el empleado, o depende de él, y por lo tanto se debería pasar a la tabla Empleados. Las dependencias incoherentes pueden dificultar el acceso porque la ruta para encontrar los datos puede no estar o estar interrumpida.

### 2.6.2. Formas Normales

Existen reglas en la normalización de una base de datos. Cada regla se denomina una "forma normal". Si se cumple la primera regla, se dice que la base de datos está en la "primera forma normal".

Si se cumplen las tres primeras reglas, la base de datos se considera que está en la "tercera forma normal". Aunque son posibles otros niveles de normalización, la tercera forma normal se considera el máximo nivel necesario para la mayor parte de las aplicaciones.

Al igual que con otras muchas reglas y especificaciones formales, en los escenarios reales no siempre se cumplen los estándares de forma perfecta. En general, la normalización requiere tablas adicionales y algunos clientes consideran éste un trabajo considerable.

Si decide infringir una de las tres primeras reglas de la normalización, asegúrese de que su aplicación se anticipa a los problemas que puedan aparecer, como la existencia de datos redundantes y de dependencias incoherentes.

#### 2.6.2.1. Primera Forma Normal

1. Eliminar los grupos repetidos de las tablas individuales.
2. Crear una tabla independiente para cada conjunto de datos relacionados.
3. Identificar cada conjunto de datos relacionados con una clave principal.

No se debe usar varios campos en una sola tabla para almacenar datos similares. Por ejemplo, para realizar el seguimiento de un elemento del inventario que proviene de dos orígenes posibles, un registro del inventario puede contener campos para el Código de proveedor 1 y para el Código de proveedor 2.

Si se agrega un tercer proveedor Agregar un campo no es la respuesta, requiere modificaciones en las tablas y el programa, y no admite fácilmente un número variable de proveedores. En su lugar, debe colocarse toda la información de los proveedores en una tabla independiente denominada Proveedores y después vincular el inventario a los proveedores con el número de elemento como clave, o los proveedores al inventario con el código de proveedor como clave.

#### 2.6.2.2. Segunda Forma Normal

4. Crear tablas independientes para conjuntos de valores que se apliquen a varios registros.
5. Relacionar estas tablas con una clave externa.

Los registros no deben depender de nada que no sea una clave principal de una tabla, una clave compuesta si es necesario. Por ejemplo, podemos considerar la dirección de un cliente en un sistema de contabilidad. La dirección se necesita en la tabla Clientes, pero también en las tablas Pedidos, Envíos, Facturas, Cuentas por cobrar y Colecciones. En lugar de almacenar la dirección de un cliente como una entrada independiente en cada una de estas tablas, se puede almacenar en un lugar, ya sea en la tabla Clientes o en una tabla Direcciones independiente.

#### 2.6.2.3. Tercera Forma Normal

6. Eliminar los campos que no dependan de la clave.

Los valores de un registro que no sean parte de la clave de ese registro no pertenecen a la tabla. En general, siempre que el contenido de un grupo de campos pueda aplicarse a más de un único registro de la tabla, se debe considerar el colocar estos campos en una tabla independiente. Por ejemplo, en

una tabla Contratación de empleados, puede incluirse el nombre de la universidad y la dirección de un candidato. Pero necesita una lista completa de universidades para enviar mensajes de correo electrónico en grupo. Si la información de las universidades se almacena en la tabla Candidatos, no hay forma de enumerar las universidades que no tengan candidatos en ese momento. Cree una tabla Universidades independiente y vincúlela a la tabla Candidatos con el código de universidad como clave.

EXCEPCIÓN: cumplir la tercera forma normal, aunque en teoría es deseable, no siempre es práctico. Si se tiene una tabla Clientes y se desea eliminar todas las dependencias posibles entre los campos, se deben crear tablas independientes para las ciudades, códigos postales, representantes de venta, clases de clientes y cualquier otro factor que pueda estar duplicado en varios registros. En teoría, la normalización merece el trabajo que supone. Sin embargo, muchas tablas pequeñas pueden degradar el rendimiento o superar la capacidad de memoria o de archivos abiertos.

Puede ser más factible aplicar la tercera forma normal sólo a los datos que cambian con frecuencia. Si quedan algunos campos dependientes, se tiene que diseñar la aplicación para que pida al usuario que compruebe todos los campos relacionados cuando cambie alguno.

## 2.7. Programación Orientada a Objetos

La programación orientada a objetos o POO, es un paradigma de programación que usa objetos y sus interacciones, para diseñar aplicaciones y programas informáticos. Está basado en varias técnicas, incluyendo herencia, abstracción, polimorfismo y encapsulamiento [14]. Su uso se popularizó a principios de la década de los años 1990. En la actualidad, existe variedad de lenguajes de programación que soportan la orientación a objetos.

Los objetos son entidades que tienen un determinado estado, comportamiento (método) e identidad:

- El estado está compuesto de datos, será uno o varios atributos a los que se habrán asignado unos valores concretos (datos).
- El comportamiento está definido por los métodos o mensajes a los que sabe responder dicho objeto, es decir, qué operaciones se pueden realizar con él.
- La identidad es una propiedad de un objeto que lo diferencia del resto, dicho con otras palabras, es su identificador (concepto análogo al de identificador de una variable o una constante).

Un objeto contiene toda la información que permite definirlo e identificarlo frente a otros objetos pertenecientes a otras clases e incluso frente a objetos de una misma clase, al poder tener valores bien diferenciados en sus atributos. A su vez, los objetos disponen de mecanismos de interacción llamados métodos, que

favorecen la comunicación entre ellos. Esta comunicación favorece a su vez el cambio de estado en los propios objetos. Esta característica lleva a tratarlos como unidades indivisibles, en las que no se separa el estado y el comportamiento.

Los métodos (comportamiento) y atributos (estado) están estrechamente relacionados por la propiedad de conjunto. Esta propiedad destaca que una clase requiere de métodos para poder tratar los atributos con los que cuenta. El programador debe pensar indistintamente en ambos conceptos, sin separar ni darle mayor importancia a alguno de ellos. Hacerlo podría producir el hábito erróneo de crear clases contenedoras de información por un lado y clases con métodos que manejen a las primeras por el otro. De esta manera se estaría realizando una programación estructurada camuflada en un lenguaje de programación orientado a objetos.

La POO difiere de la programación estructurada tradicional, en la que los datos y los procedimientos están separados y sin relación, ya que lo único que se busca es el procesamiento de unos datos de entrada para obtener otros de salida. La programación estructurada anima al programador a pensar sobre todo en términos de procedimientos o funciones, y en segundo lugar en las estructuras de datos que esos procedimientos manejan. En la programación estructurada solo se escriben funciones que procesan datos. Los programadores que emplean POO, en cambio, primero definen objetos para luego enviarles mensajes solicitándoles que realicen sus métodos por sí mismos.

## 2.8. Arquitectura cliente – servidor

La arquitectura cliente-servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes [15]. Un cliente realiza peticiones a otro programa, el servidor, que le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema.

La separación entre cliente y servidor es una separación de tipo lógico, donde el servidor no se ejecuta necesariamente sobre una sola máquina ni es necesariamente un sólo programa. Los tipos específicos de servidores incluyen los servidores web, los servidores de archivo, los servidores del correo, etc. Mientras que sus propósitos varían de unos servicios a otros, la arquitectura básica seguirá siendo la misma.

Una disposición muy común son los *sistemas multicapa* en los que el servidor se descompone en diferentes programas que pueden ser ejecutados por diferentes computadoras aumentando así el grado de distribución del sistema.

La *arquitectura cliente-servidor* sustituye a la *arquitectura monolítica* en la que no hay distribución, tanto a nivel físico como a nivel lógico.

La red cliente-servidor es aquella red de comunicaciones en la que todos los clientes están conectados a un servidor, en el que se centralizan los diversos recursos y aplicaciones con que se cuenta; y que los pone a disposición de los clientes cada vez que estos son solicitados. Esto significa que todas las gestiones que se realizan se concentran en el servidor, de manera que en él se disponen los requerimientos provenientes de los clientes que tienen prioridad, los archivos que son de uso público y los que son de uso restringido, los archivos que son de sólo lectura y los que, por el contrario, pueden ser modificados, etc. Este tipo de red puede utilizarse conjuntamente en caso de que se esté utilizando en una red mixta.

### 2.8.1. HTML

HTML (*Hyper Text Markup Language* por sus siglas en ingles) o también conocido como lenguaje de marcado de hipertexto, es el lenguaje de marcado predominante para la elaboración de páginas web [16]. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir un *script* (por ejemplo JavaScript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

El lenguaje HTML puede ser creado y editado con cualquier editor de textos básico, como puede ser Gedit en Linux, el Bloc de notas de Windows, o cualquier otro editor que admita texto sin formato como GNU Emacs, Microsoft Wordpad, TextPad, Vim, Notepad++, entre otros.

Existen además, otros editores para la realización de sitios web con características WYSIWYG (*What You SeeIs What You Get*, o en español: «lo que ves es lo que obtienes»). Estos editores permiten ver el resultado de lo que se está editando en tiempo real, a medida que se va desarrollando el documento. Ahora bien, esto no significa una manera distinta de realizar sitios web, sino que una forma un tanto más simple ya que estos programas, además de tener la opción de trabajar con la vista preliminar, tiene su propia sección HTML la cual va generando todo el código a medida que se va trabajando. Algunos ejemplos de editores WYSIWYG son KompoZer, Microsoft FrontPage, o Adobe Dreamweaver.

HTML utiliza etiquetas o marcas, que consisten en breves instrucciones de comienzo y final, mediante con lo cual se determina la forma en la que debe

aparecer en su navegador el texto, así como también las imágenes y los demás elementos, en la pantalla del ordenador.

### 2.8.2. Apache

El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual [17]. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue rescrito por completo. Su nombre se debe a que Behelendorf quería que tuviese la connotación de algo que es firme y enérgico pero no agresivo, y la tribu Apache fue la última en rendirse al que pronto se convertiría en gobierno de EEUU, y en esos momentos la preocupación de su grupo era que llegasen las empresas y "civilizasen" el paisaje que habían creado los primeros ingenieros de internet. Además Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. Era, en inglés, *a patchy server* (un servidor "parcheado").

El servidor Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation.

Apache presenta entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración.

Apache tiene amplia aceptación en la red: desde 1996, Apache, es el servidor HTTP más usado. Alcanzó su máxima cuota de mercado en 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo, sin embargo ha sufrido un descenso en su cuota de mercado en los últimos años. (Estadísticas históricas y de uso diario proporcionadas por Netcraft ).

La mayoría de las vulnerabilidades de la seguridad descubiertas y resueltas tan sólo pueden ser aprovechadas por usuarios locales y no remotamente. Sin embargo, algunas se pueden accionar remotamente en ciertas situaciones, o explotar por los usuarios locales malévolos en las disposiciones de recibimiento compartidas que utilizan PHP como módulo de Apache.

### 2.8.3. PHP

PHP es un acrónimo recursivo que significa *PHP Hypertext Pre-processor* (inicialmente *PHP Tools*, o, *Personal Home Page Tools*). Fue creado originalmente por Rasmus Lerdorf en 1994; sin embargo la implementación principal de PHP es producida ahora por The PHP Group y sirve como el estándar de facto para PHP al no haber una especificación formal. Publicado bajo la PHP License, la Free Software Foundation considera esta licencia como software libre [18].

Puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. El lenguaje PHP se encuentra instalado en más de 20 millones de sitios web y en un millón de servidores, el número de sitios en PHP ha compartido algo de su preponderante dominio con otros nuevos lenguajes no tan poderosos desde agosto de 2005. El sitio web de Wikipedia está desarrollado en PHP. Es también el módulo Apache más popular entre las computadoras que utilizan Apache como servidor web.

El gran parecido que posee PHP con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores crear aplicaciones complejas con una curva de aprendizaje muy corta. También les permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones.

Aunque todo en su diseño está orientado a facilitar la creación de sitios web, es posible crear aplicaciones con una interfaz gráfica para el usuario, utilizando la extensión PHP-Qt o PHP-GTK. También puede ser usado desde la línea de órdenes, de la misma manera como Perl o Python pueden hacerlo; a esta versión de PHP se la llama PHP-CLI (*Command Line Interface*).

Cuando el cliente hace una petición al servidor para que le envíe una página web, el servidor ejecuta el intérprete de PHP. Éste procesa el script solicitado que generará el contenido de manera dinámica (por ejemplo obteniendo información de una base de datos). El resultado es enviado por el intérprete al servidor, quien a su vez se lo envía al cliente. Mediante extensiones es también posible la generación de archivos PDF, Flash, así como imágenes en diferentes formatos.

Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, PostgreSQL, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite.

PHP también tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos, tales como Unix (y de ese tipo, como Linux o Mac OS X) y Microsoft Windows, y puede interactuar con los servidores de web más populares ya que existe en versión CGI, módulo para Apache, e ISAPI.

PHP es una alternativa a las tecnologías de Microsoft ASP y ASP.NET (que utiliza C# y Visual Basic .NET como lenguajes), a ColdFusion de la empresa Adobe, a JSP/Java y a CGI/Perl. Aunque su creación y desarrollo se da en el ámbito de los sistemas libres, bajo la licencia GNU, existe además un entorno de desarrollo integrado comercial llamado Zend Studio. CodeGear (la división de lenguajes de programación de Borland) ha sacado al mercado un entorno de desarrollo integrado para PHP, denominado 'Delphi for PHP. También existen al menos un par de módulos para Eclipse, uno de los entornos más populares.

### 2.8.4. MySQL

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones [19]. MySQL AB —desde enero de 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009— desarrolla MySQL como software libre en un esquema de licenciamiento dual.

Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C.

Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y los derechos de autor del código están en poder del autor individual, MySQL es patrocinado por una empresa privada, que posee el copyright de la mayor parte del código.

Esto es lo que posibilita el esquema de licenciamiento anteriormente mencionado. Además de la venta de licencias privativas, la compañía ofrece soporte y servicios. Para sus operaciones contratan trabajadores alrededor del mundo que colaboran vía Internet. MySQL AB fue fundado por David Axmark, Allan Larsson y Michael Widenius.

# CAPÍTULO 3: ANÁLISIS Y DISEÑO

## 3.1 Introducción

Se planea crear un sistema que permita realizar altas, generar opiniones y hacer recomendaciones referentes a libros de distintas disciplinas y editoriales para así considerar su consulta, uso o compra de aquellos recomendados que cumplan con las necesidades del lector.

## 3.2 Planteamiento del Problema

A continuación se describen las características principales que debe tener el sistema:

Funciones del Producto

Las funciones principales que el software tiene que realizar son:

- Registro del libro considerando el título, autor, ISBN, áreas relacionadas e idioma.
- Registro del usuario considerando nombre, cargo, teléfono de contacto o celular, correo electrónico, empresa, carrera, experiencia en el área.
- Generación de recomendación del libro considerando en que cursos pueden ser empleados, aspectos relevantes de la obra, que elementos le faltan por cubrir.
- Historial de recomendaciones.

Tipos de Usuarios

Se contará con 2 tipos de usuarios en el sistema:

- Visitante.- Este usuario es aquel que visita la página web, puede realizar búsquedas de libros, búsqueda de recomendaciones, registrarse en la página web o iniciar sesión en ella. Como conocimientos requeridos para utilizar el sistema solo se considera que necesita conocimientos básicos en computación para acceder a la página web.
- Usuario.- Este usuario es aquel que ha iniciado sesión en la pagina web y que previamente ya se había registrado, cuenta con privilegios para agregar un libro nuevo, agregar una recomendación o agregar una opinión de un libro en especifico, así como modificar sus datos de perfil o cerrar sesión. Los conocimientos requeridos por los usuarios son los mismos que los visitantes.

## 3.3 Requerimientos

Para cumplir con las funcionalidades que el sistema presentó en esta fase se capturaron requerimientos de 2 tipos:

Requerimientos funcionales.
Son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares.

Requerimientos no funcionales.
Son restricciones de los servicios o funciones ofrecidos por el sistema.

### 3.3.1 Funcionales

Requerimientos de Negocio

Los requisitos que el software tiene que cumplir son:

- RN1- Registro del libro considerando el título, autor, ISBN, área relacionada e idioma.
- RN2- Registro del usuario considerando nombre, cargo, teléfono de contacto o celular, correo electrónico, empresa, carrera, experiencia en el área.
- RN3- Generación de recomendación del libro considerando en que cursos pueden ser empleados, aspectos relevantes de la obra, que elementos le faltan por cubrir.
- RN4- Historial de las recomendaciones realizadas para cada libro.
- RN5- Búsqueda de libros mediante título, autor, área relacionada.
- RN6- Mostrar los libros mas recomendados.

Requerimientos de Usuario

En este sistema el usuario desempeñará las siguientes actividades:

1. RU1- Visitar la página web.
2. RU2- Registrarse.
3. RU3- Iniciar Sesión.
4. RU4- Buscar Libros.
5. RU5- Agregar Libros.
6. RU6- Realizar recomendaciones de un Libro.

Requerimientos Funcionales

Para realizar cada uno de los Requerimientos que Usuario presenta, se propone lo siguiente:

- RF1- Tendrá opciones en la parte superior como: **Inicio**, **Búsqueda**, **Login**, **Registrarse** y **Acerca de**.
- RF2- Para visitar la página web mencionado en RU1 (Visitar la página web), basta con escribir la dirección en el navegador web.
- RF3-Para poder registrarse RU2 (Registrarse), el usuario debe hacer clic en la opción **Registrarse**, en esta opción el visitante debe ingresar sus datos en el formulario y se le creará una cuenta para después poder iniciar sesión.
- RF4- Una vez registrado, el usuario puede iniciar sesión RU3 (Iniciar Sesión), para iniciar sesión se debe hacer clic en la opción **Login** el usuario deberá introducir sus datos de cuenta para iniciar su sesión.
- RF5- Para realizar la búsqueda de libros RU4(Buscar Libros), el usuario debe hacer clic en la opción **Búsqueda** donde podrá ingresar los datos del libro que busca, en caso que no se encuentre el libro, si el usuario ha iniciado sesión se le mostrará la opción de agregar libros RU5(Agregar Libros). Si al buscar un libro y aparece en el sistema, entonces se mostrarán las opciones para recomendar RU6 (Realizar Recomendación de un Libro), acerca de ese libro.
- RF6- La página **Inicio** tendrá una descripción rápida de la página web.
- RF7-En la página **Acerca de** habrá información del desarrollador de la página.

### 3.3.2 No Funcionales

Requerimientos de Rendimiento

- Se debe contar con una página web que despliegue información de manera rápida y con alta disponibilidad.

Requerimientos de Seguridad

- Se debe mantener segura las contraseñas de los usuarios dentro de la base de datos y en la página web.

Atributos de Calidad de Software

- Adaptabilidad.- El sistema debe adaptarse a diferentes servidores host.
- Disponibilidad.- Debe contar con las características necesarias para que la pagina siempre este disponible.
- Interoperabilidad.- La página web debe desplegarse de manera correcta en diferentes sistemas operativos y diferentes navegadores web.

- Mantenimiento.- Se debe dar mantenimiento a la base de datos. Esta tarea solo será realizada por el Administrador de la Base de Datos.

## 3.4 Alcances

Dentro de los objetivos a cumplir en este proyecto se encuentran los siguientes:

- Se pretende obtener un sistema completo y funcional que provea al público en general el servicio de recomendar libros, de buscar libros más recomendados y de opinar acerca de libros específicos.
- Se pretende generar una tesis que cuente con 5 capítulos que describan al sistema.
- Se pretende crear un manual de usuario para facilitar el uso y entendimiento del sistema.
- Se pretende generar un manual técnico con diagramas y modelos del sistema.

## 3.5 Metodología

En base a la metodología de Software del Modelo en Espiral se desarrollarán este sistema por lo que se dividió en etapas para ir avanzando mientras se corrigen y perfeccionan fases anteriores en cada iteración por lo que para cada ciclo habrá cuatro actividades que define el modelo:

Determinar Objetivos.
Análisis del riesgo.
Planificación.
Desarrollar y probar.

En el primer ciclo, se realizarán las siguientes actividades:

- Realización de la especificación de requerimientos.
- Generación de los casos de uso y diagramas para el análisis del sistema.
- Diseño del modelo entidad relación, diagrama relacional y normalización de la Base de datos.

En la segunda etapa, se realizarán las siguientes actividades:

- Implementación en MySQL del modelo relacional.
- Diseño de la interfaz del sistema.
- Implementación de la conexión de MySQL con PHP.
- Implementación de las funcionalidades del sistema

En la tercera etapa, se realizará las siguientes actividades

- Analizar las vulnerabilidades de seguridad del sistema.
- Implantar las estrategias de seguridad del sistema.
- Generación de pruebas del sistema.

## 3.6 Análisis del Sistema

El Lenguaje Unificado de Modelado (UML) es una técnica para la especificación de sistemas en todas sus fases.

UML es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software y ofrece un estándar para describir un modelo del sistema, incluyendo aspectos conceptuales tales como requerimientos de negocio y funcionales, aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

UML sirve para hacer modelos que permitan:

- Visualizar como es un sistema o como queremos que sea.
- Especificar la estructura y/o comportamiento de un sistema.
- Hacer una plantilla que guíe la construcción de los sistemas
- Documentar las decisiones que hemos tomado.

A continuación se muestran los diagramas que se utilizaran para definir el diseño del sistema.

### 3.6.1 Diagrama de Casos de Uso

Los diagramas de casos de uso describen las relaciones y las dependencias entre un grupo de casos de uso y los actores participantes en el proceso.

Es importante resaltar que los diagramas de casos de uso no están pensados para representar el diseño y no puede describir los elementos internos de un sistema. Los diagramas de casos de uso sirven para facilitar la comunicación con los futuros usuarios del sistema, y con el cliente, y resultan especialmente útiles para determinar las características necesarias que tendrá el sistema. En otras palabras, los diagramas de casos de uso describen qué es lo que debe hacer el sistema, pero no cómo.

En la figura 3.1 se presenta los actores que intervienen en el sistema y las funcionalidades que pueden realizar.

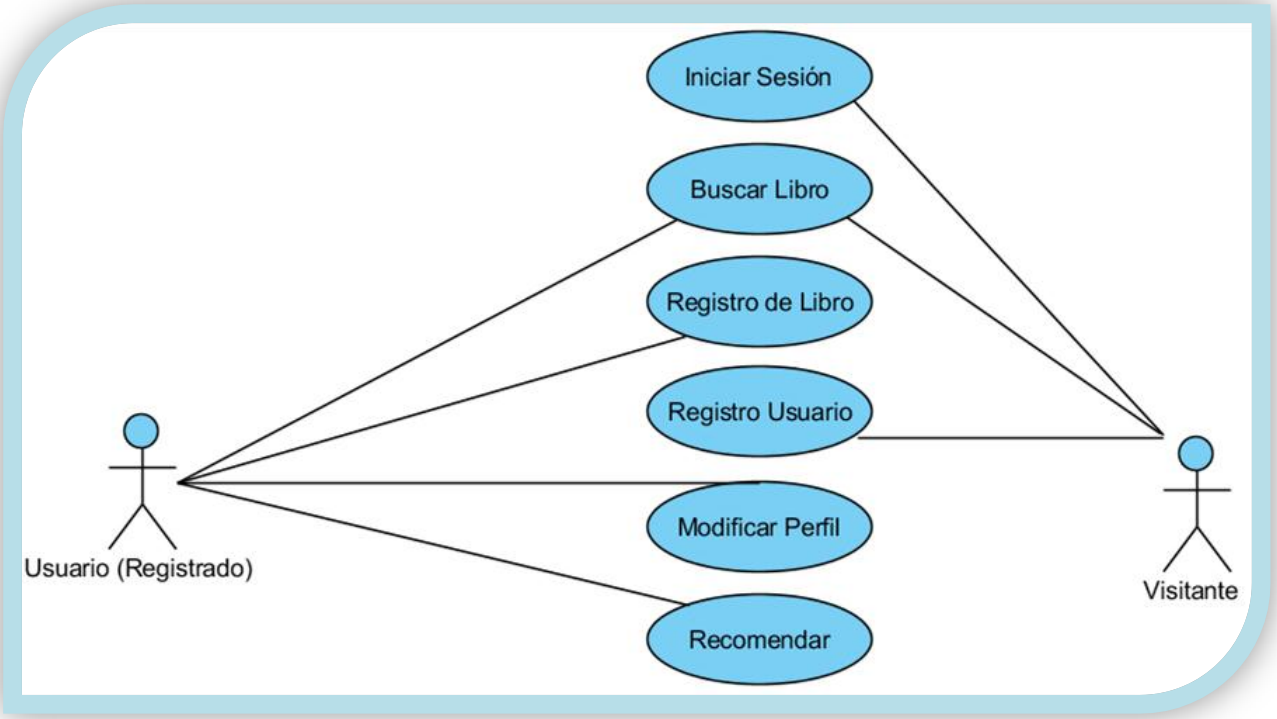


Figura 3.1 Diagrama de Casos de Uso

### 3.6.2 Especificación de Casos de Uso

Un caso de uso es una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo algún proceso. A continuación se muestran los casos de uso identificados en el sistema.

Buscar Libro

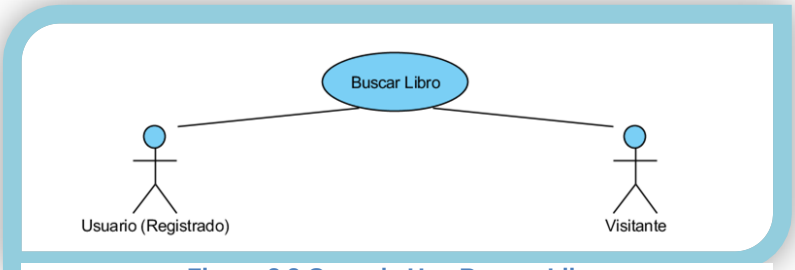


Figura 3.2 Caso de Uso Buscar Libro

Descripción: Es la manera en la que se consultará la disponibilidad así como sus recomendaciones u opiniones de un libro en el sistema. La búsqueda se podrá realizar por usuarios o visitantes y por nombre de autor, título o área (véase figura 3.2).

Precondiciones:

- Se cuente con internet.
- El usuario debe haber ingresado a la página web.

Flujo de eventos normales

| Paso | Acción Usuario | Paso | Acción Sistema | Excepción |
|---|---|---|---|---|
| 1 | Da clic en la opción Buscar | 2 | La pantalla muestra un campo a llenar con el criterio de búsqueda | |
| 3 | Ingresar los datos que se desean buscar | | | |
| 4 | Dar clic en el botón Aceptar | 5 | Se muestran los resultados de la búsqueda | E1,E2 |

Flujo de eventos alternativos

| Id | Nombre | Acción |
|---|---|---|
| E1 | La conexión con la base de datos se interrumpió | Se manda un mensaje de error el cual indique el error causado |
| E2 | Los campos están vacíos o faltó alguno | Se manda un mensaje de error mencionando error al llenar los campos |

Registro de Libro

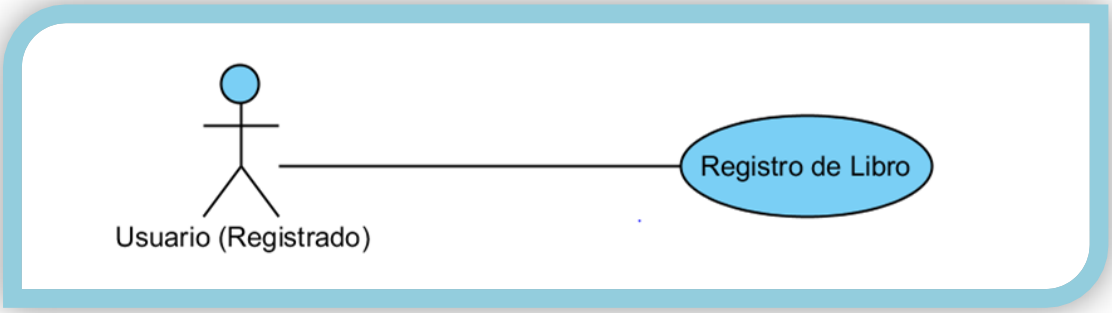


**Figura 3.3 Caso de Uso Registro de Libro**

Descripción: En esta acción el usuario puede dar de alta un nuevo libro para realizar recomendaciones u opiniones del libro (véase figura 3.3).

Precondiciones:

- Se cuente con internet.
- El usuario debe haber ingresado a la página web.
- El usuario debe haber iniciado sesión.

Flujo de eventos normales

| Paso | Acción Usuario | Paso | Acción Sistema | Excepción |
|---|---|---|---|---|
| 1 | Da clic en la opción Agregar un libro | 2 | La pantalla muestra un campo a llenar con los datos del libro | |
| 3 | Se da clic en el botón Agregar | 4 | Se muestra un mensaje de que se agregó con éxito. | E1,E2 |

Flujo de eventos alternativos

| Id | Nombre | Acción |
|---|---|---|
| E1 | La conexión con la base de datos se interrumpió | Se manda un mensaje de error el cual indique el error |
| E2 | Los campos están vacíos o faltó alguno | Se manda un mensaje de error mencionando error al llenar los campos |

Registro Usuario

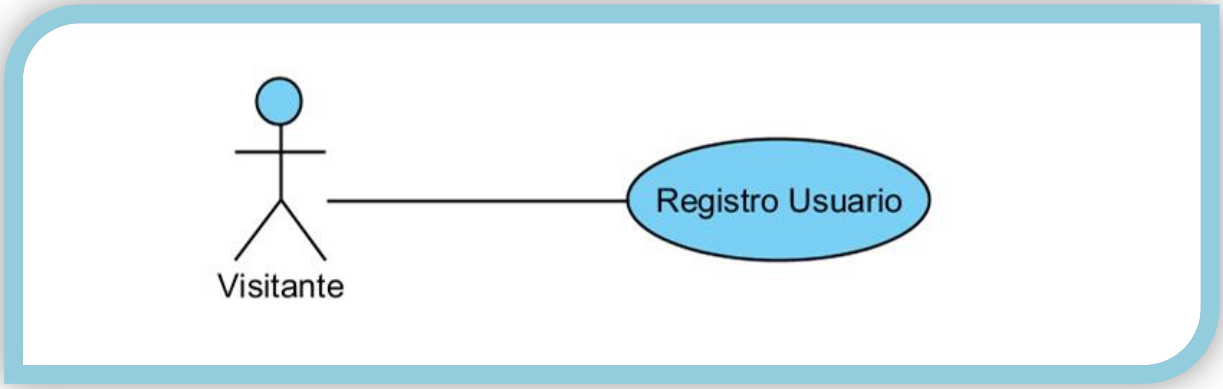


Figura 3.4 Caso de Uso Registro Usuario

Descripción: Es la forma en que un visitante pasa a ser usuario de la página, con lo cual tiene privilegio para realizar recomendaciones, opiniones y agregar libros (véase figura 3.4).

Precondiciones:

- Se cuente con internet.
- El usuario debe haber ingresado a la página web.

Flujo de eventos normales

| Paso | Acción Usuario | Paso | Acción Sistema | Excepción |
|---|---|---|---|---|
| 1 | Da clic en la opción Registrarse | 2 | La pantalla muestra un formulario a llenar con los datos del usuario | |
| 3 | Ingresar sus datos | | | |
| 4 | Dar clic en el botón Aceptar | 5 | Se muestra una pantalla confirmando que se ha realizado con éxito el registro. | E1,E2 |

Flujo de eventos alternativos

| Id | Nombre | Acción |
|---|---|---|
| E1 | La conexión con la base de datos se interrumpió | Se manda un mensaje de error el cual indique el error |
| E2 | Los campos están vacíos o faltó alguno | Se manda un mensaje de error mencionando error al llenar los campos |

Modificar Perfil


Figura 3.5 Caso de Uso Modificar Perfil

Descripción: Mediante esta opción se pueden actualizar los datos del usuario y únicamente interviene el usuario (véase figura 3.5).

Precondiciones:

- Se cuente con internet.
- El usuario debe haber ingresado a la página web.
- El usuario debe hacer iniciado sesión.

Flujo de eventos normales

| Paso | Acción Usuario | Paso | Acción Sistema | Excepción |
|---|---|---|---|---|
| 1 | Da clic en la opción Mi Perfil | 2 | La pantalla muestra un formulario con los datos del usuario previamente almacenado | |
| 3 | Cambia los datos que quiere modificar | | | |
| 4 | Dar clic en el botón Aceptar | 5 | Se muestra una pantalla diciendo que la actualización se ha realizado con éxito. | E1,E2 |

Flujo de eventos alternativos

| Id | Nombre | Acción |
|---|---|---|
| E1 | La conexión con la base de datos se interrumpió | Se manda un mensaje de error el cual indique el error |
| E2 | Los campos están vacíos o faltó alguno | Se manda un mensaje de error mencionando error al llenar los campos |

Recomendar


Figura 3.6 Caso de Uso Recomendar

Descripción: En esta opción un usuario puede realizar una recomendación de un libro (véase figura 3.6).

Precondiciones:

- Se cuente con internet.
- El usuario debe haber ingresado a la página web.
- El usuario debe haber iniciado sesión para realizar la recomendación.
- El libro a recomendar debe estar registrado.
- Hacer una búsqueda del libro a recomendar.

Flujo de eventos normales

| Paso | Acción Usuario | Paso | Acción Sistema | Excepción |
|---|---|---|---|---|
| 1 | Da clic en la opción Recomienda este Libro | 2 | La pantalla muestra un formulario a llenar. | |
| 3 | Ingresar los datos que se pide en pantalla. | | | |
| 4 | Dar clic en el botón Aceptar | 5 | Se muestra una pantalla diciendo que la recomendación se agregó exitosamente. | E1,E2 |

Flujo de eventos alternativos

| Id | Nombre | Acción |
|---|---|---|
| E1 | La conexión con la base de datos se interrumpió | Se manda un mensaje de error el cual indique el error |
| E2 | Los campos están vacíos o faltó alguno | Se manda un mensaje de error mencionando error al llenar los campos |

Iniciar Sesión

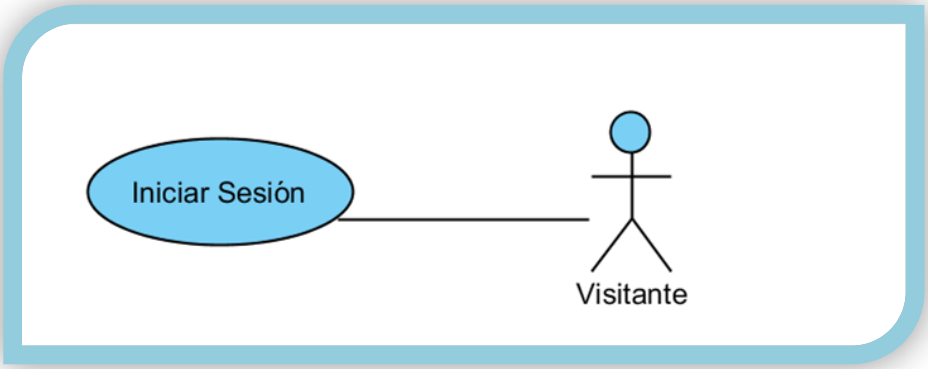


Figura 3.7 Caso de Uso Iniciar Sesión

Descripción: Mediante esta opción un visitante puede pasar de ser visitante a ser un usuario de la página web (véase figura 3.7).

Precondiciones:

- Se cuente con internet.
- El usuario debe haber ingresado a la página web.

Flujo de eventos normales

| Paso | Acción Usuario | Paso | Acción Sistema | Excepción |
|---|---|---|---|---|
| 1 | Da clic en la opción Iniciar Sesión. | 2 | La pantalla muestra un formulario a llenar con los datos requeridos para iniciar sesión. | |
| 3 | Ingresar sus datos | | | |
| 4 | Dar clic en el botón Aceptar | 5 | Se muestra una pantalla confirmando que se ha realizado con éxito el inicio de sesión. | E1,E2 |

Flujo de eventos alternativos

| Id | Nombre | Acción |
|---|---|---|
| E1 | La conexión con la base de datos se interrumpió | Se manda un mensaje de error el cual indique el error |
| E2 | Los campos están vacíos o faltó alguno | Se manda un mensaje de error mencionando error al llenar los campos |

### 3.6.3 Diagrama de Clases

Un diagrama de clases es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. En la figura 3.8 se muestra las clases principales del sistema.

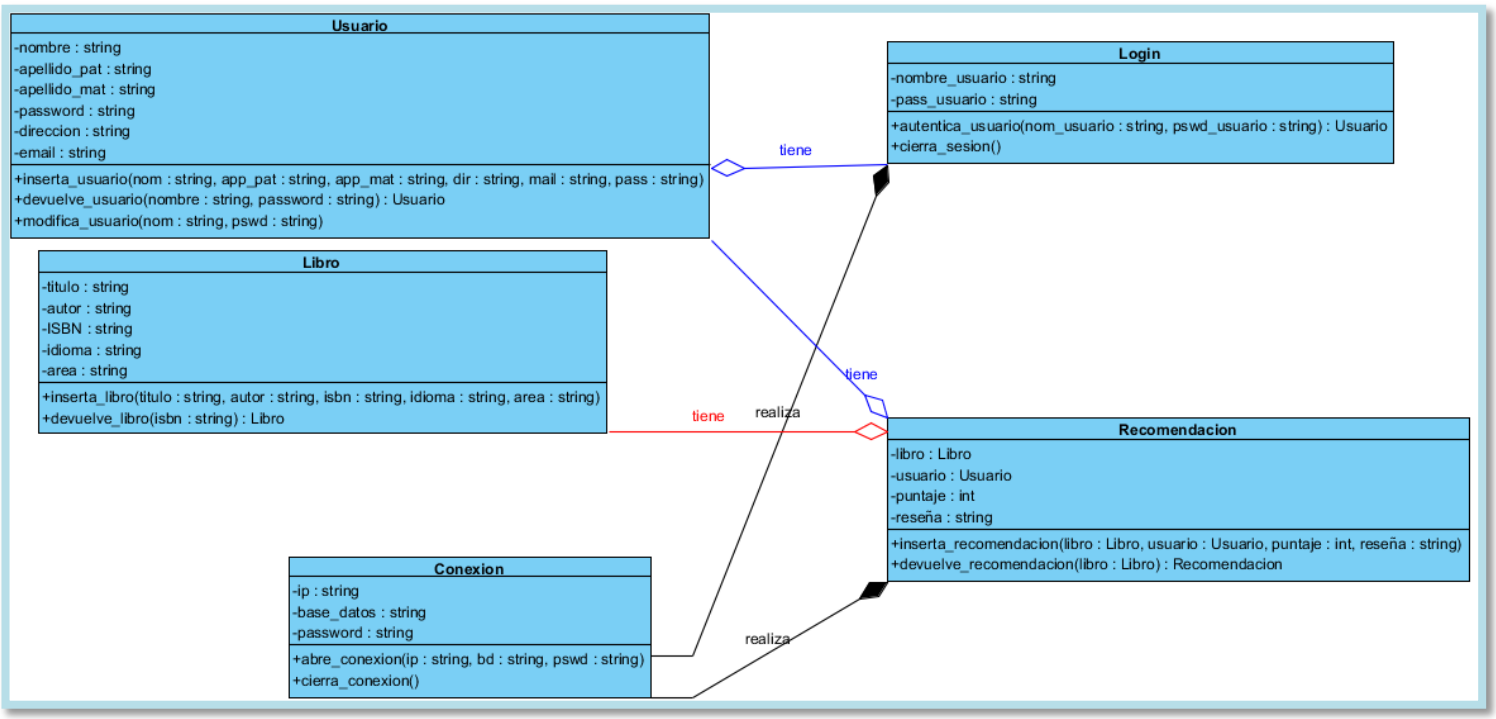


Figura 3.8 Diagrama de Clases

## 3.7 Diseño del Sistema

El Diseño de sistemas es la fase en la que se define la arquitectura de hardware y software, componentes, módulos y datos de un sistema de cómputo para satisfacer ciertos requerimientos.

### 3.7.1 Diagrama de Estados

El diagrama de estado muestra el conjunto de estados por los cuales pasa un usuario en una aplicación en respuesta a eventos. En el diagrama 3.9 se muestra los estados posibles del sistema.

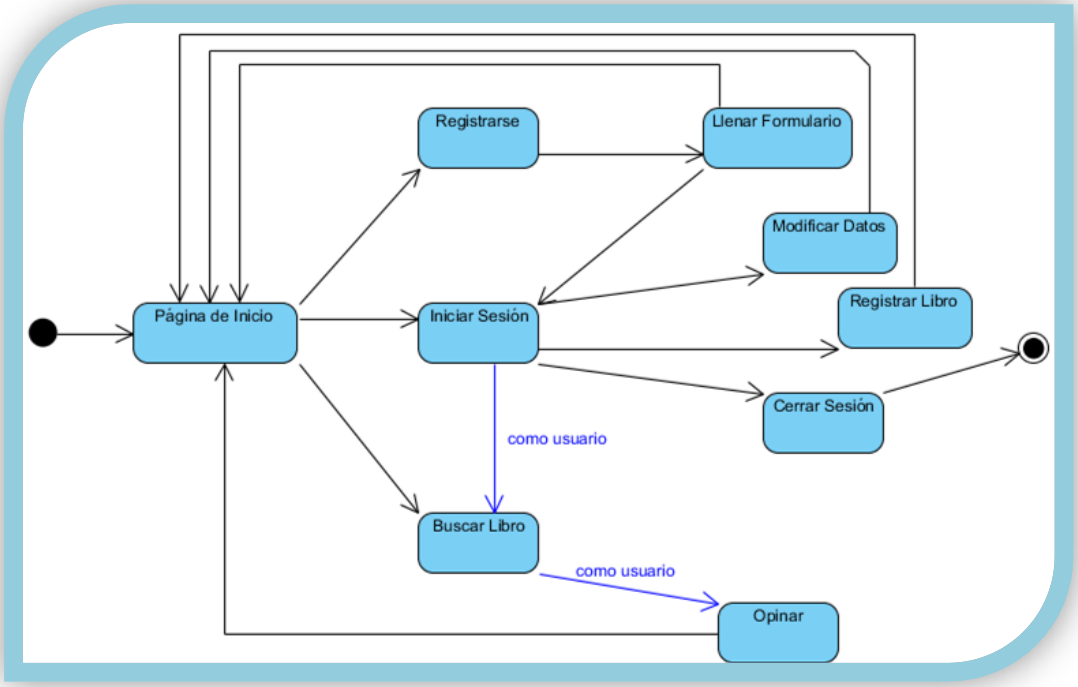


Figura 3.9 Diagrama de Estados

### 3.7.2 Diagramas de Secuencia

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo. A continuación se describen casos en los que se efectúa interacción entre objetos del sistema.

Iniciar Sesión – En esta acción el visitante proporcionando un correo electrónico y una contraseña realiza el proceso que se muestra en la figura 3.10 para iniciar sesión.

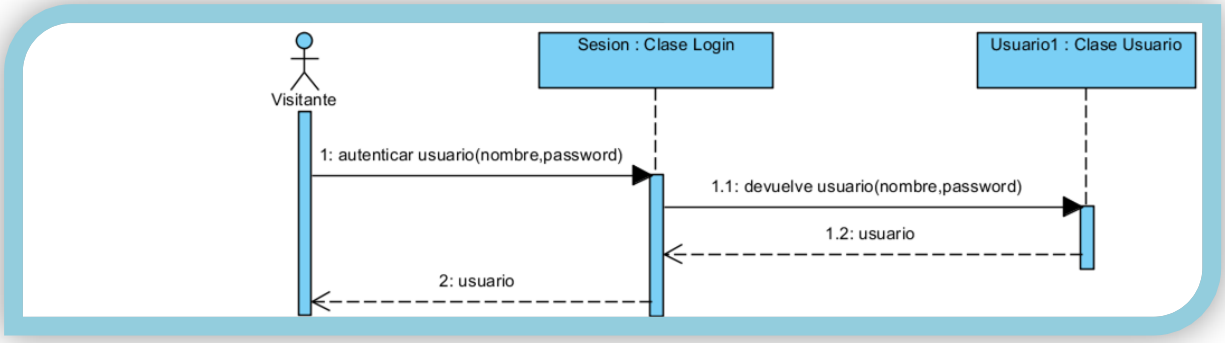


Figura 3.10 5Diagrama de Secuencia Iniciar Sesión

Recomendar - Para realizar una recomendación el sistema sigue el flujo representado en el diagrama 3.

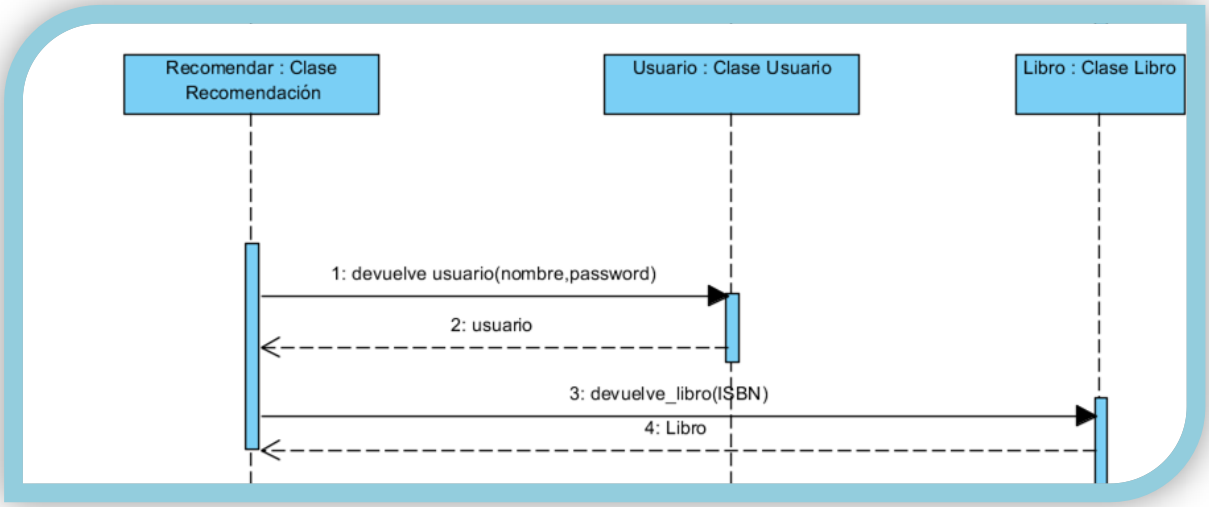


Figura 3.11 Diagrama de Secuencia Recomendar

### 3.7.3 Glosario de Términos

Dentro del sistema se manejan términos que a continuación se describen.

- Visitante.- Es aquel usuario de la página que ingresa por primera vez y no se ha registrado en el sistema o no ha iniciado sesión.
- Usuario.- Es aquel usuario que ya esta registrado y ha iniciado sesión en el sistema y puede realizar las operaciones a las que está facultado.
- Recomendación.- Es un texto breve que el usuario redacta a manera de dar su punto de vista acerca de un libro, en esta recomendación se tendrá un atributo que puntuará al libro.

# CAPÍTULO 4: DISEÑO DE LA BASE DE DATOS

## 4.1. Modelo conceptual de la base de datos

Existen distintos tipos de modelos conceptuales:

- Basados en registros
    - Jerárquico: datos en registros, relacionados con apuntadores y organizados como colecciones de árboles.
    - Redes: datos en registros relacionados por apuntadores y organizados en gráficas arbitrarias.
    - Relacional: datos en tablas relacionados por el contenido de ciertas columnas.
- Basados en objetos
    - Orientado a objetos: datos como instancias de objetos (incluyendo sus métodos).
    - Entidad-relación: datos organizados en conjuntos interrelacionados de objetos (entidades) con atributos asociados.

### 4.1.1. Entidades, Atributos y Relaciones.

A continuación se describe de manera general las entidades, sus atributos y las relaciones entre ellas que se han identificado para conformar la base de datos.

Entidades
- **Usuario**
- **Libro**
- **Autor**

Atributos
- **Usuario**- email, password, nombre, apellido paterno, apellido materno
- **Libro**- título, ISBN, idioma, área, editorial, fecha de publicación, numero de edición y autor(es)
- **Autor**- nombre, apellido paterno y apellido materno

Relaciones

- Los **Usuarios** recomiendan **Libros**
- Los **Usuarios** opinan acerca de **Libros**
- Los **Libros** tienen **Autores**

### 4.1.2. Modelo Entidad – Relación.

Un modelo entidad- relación es una herramienta para el modelado de datos que permite representar las entidades relevantes de un sistema de información así como sus interrelaciones y propiedades. En el diagrama 4.1 se representa cada entidad del sistema así como sus propiedades y relaciones entre ellas.

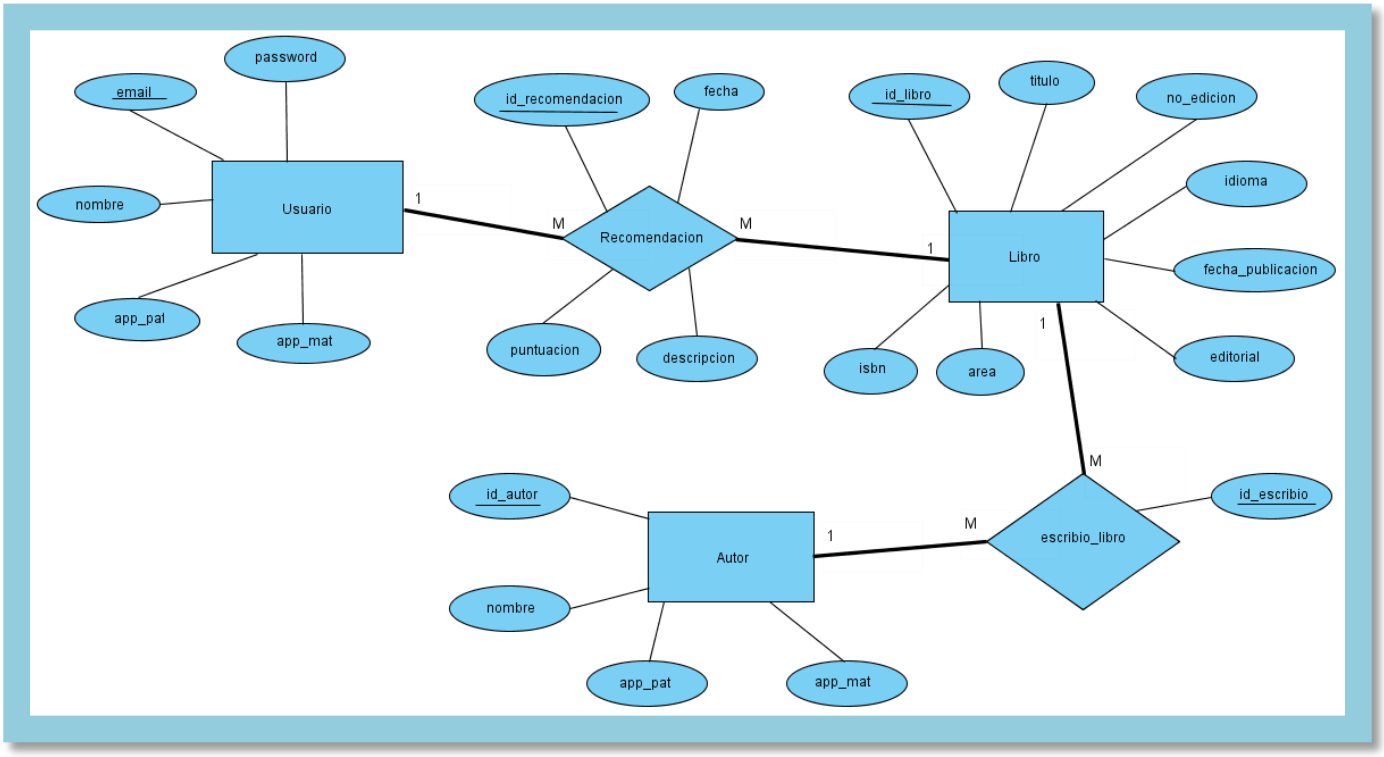


Figura 4.1 6Modelo Entidad Relación

## 4.2. Diseño Lógico.

El objetivo del diseño lógico es convertir los esquemas conceptuales de la Base de Datos en un esquema lógico global que se ajuste al modelo de Sistema de Gestión de Base de Datos sobre el que se va a implementar el sistema que en este caso es MySQL.

### 4.2.1. Esquema Relacional de la Base de Datos.

El modelo relacional define que todos los datos son almacenados en relaciones, una relación representa una tabla que no es más que un conjunto de filas, cada fila es un conjunto de campos y cada campo representa un valor que interpretado describe el mundo real (véase figura 4.2).

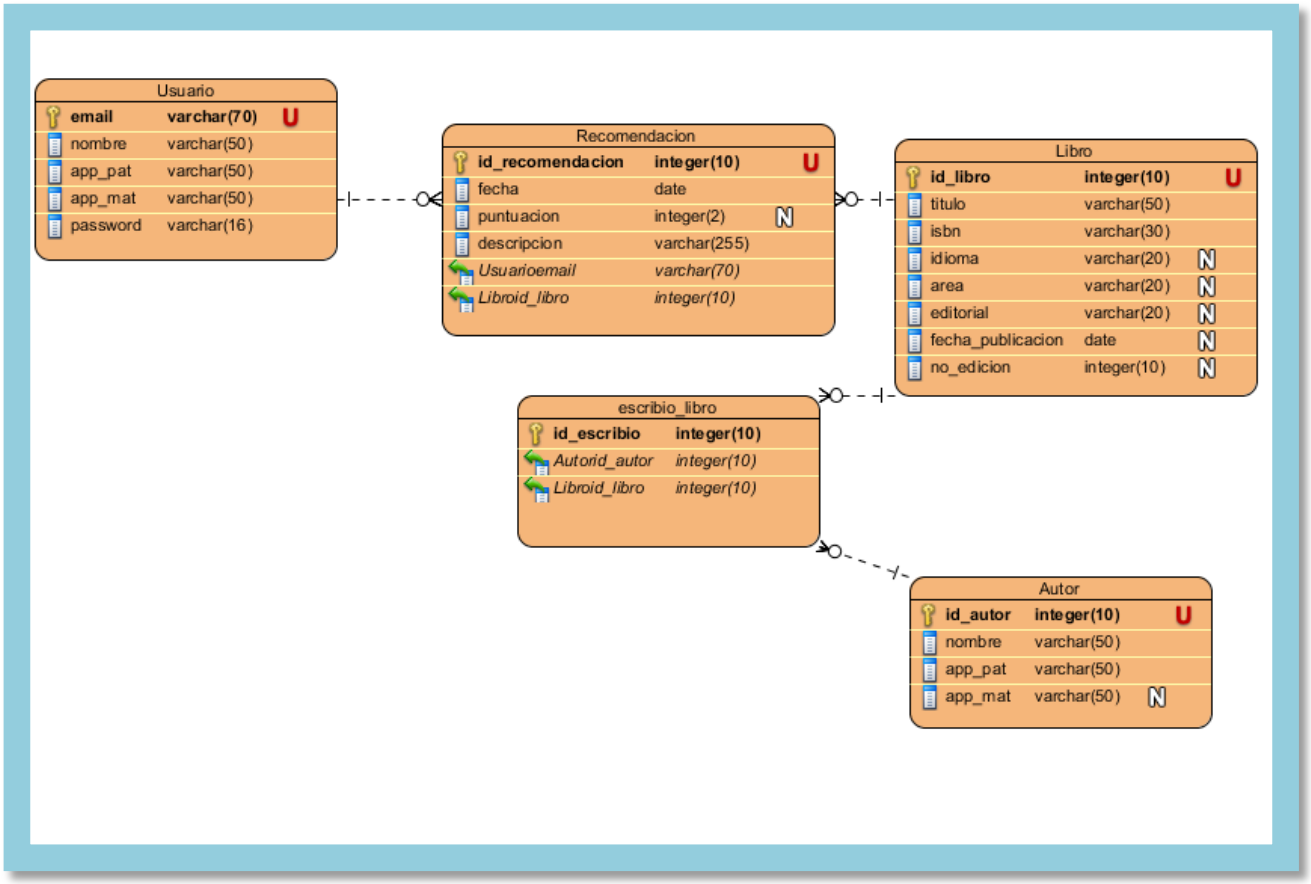


Figura 4.2 7Modelo Relacional

### 4.3. Normalización de la Base de Datos.

Se dice que la base de datos se encuentra Normalizada debido a que cumple con las 3 formas normales que son:

- 1FN- Todos y cada uno de los datos almacenados en la base de datos son atómicos (debido a que no pueden haber filas idénticas), cada columna de su respectiva tabla tiene un nombre único, todos los datos de cada columna son del mismo tipo.
- 2FN- Cumple con la 1FN, todos y cada uno de los atributos de cada tabla dependen de la llave primaria de la tabla.
- 3FN- Cumple con 2FN, no existen dependencias transitivas en atributos que no son llaves.
- 4FN- Cumple con 3FN, para las relaciones de muchos a muchos se generaron tablas para evitar redundancia en dependencias multivaluadas.

### 4.4. Instalación y configuración de los Servidores Web y de Bases de Datos.

Para implementar y configurar la base de datos diseñada, se utilizará el software WAMP que es el acrónimo usado para describir un sistema de infraestructura de internet que usa las siguientes herramientas:

- Windows, como sistema operativo;
- Apache, como servidor web;
- MySQL, como gestor de bases de datos;
- PHP como lenguaje de programación.

### 4.5. Creación de tablas utilizando el lenguaje SQL.

A continuación se muestra el Script para generar las tablas de la base de datos en MySQL.

```
-- Estructura de tabla para la tabla `autor`
--
CREATE TABLE IF NOT EXISTS `autor` (
  `id_autor` int(10) NOT NULL AUTO_INCREMENT,
  `nombre` varchar(50) NOT NULL,
  `app-pat` varchar(50) NOT NULL,
  `app_mat` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`id_autor`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
AUTO_INCREMENT=1 ;
-- --------------------------------------------------------
 Estructura de tabla para la tabla `escrito_libros`
--
CREATE TABLE IF NOT EXISTS `escrito_libros` (
  `id_escrito_libros` int(10) NOT NULL
AUTO_INCREMENT,
  `id_libro` int(10) NOT NULL,
  `id_autor` int(10) NOT NULL,
PRIMARY KEY (`id_escrito_libros`),
KEY `id_libro` (`id_libro`),
KEY `id_autor` (`id_autor`),
  KEY `id_autor_2` (`id_autor`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
AUTO_INCREMENT=1 ;
-- --------------------------------------------------------
-- Estructura de tabla para la tabla `libro`
--
CREATE TABLE IF NOT EXISTS `libro` (
  `id_libro` int(10) NOT NULL AUTO_INCREMENT,
  `titulo` varchar(50) NOT NULL,
  `isbn` varchar(30) NOT NULL,
  `idioma` varchar(20) DEFAULT NULL,
  `area` varchar(20) DEFAULT NULL,
  `editorial` varchar(20) DEFAULT NULL,
  `fecha_publicaccion` date DEFAULT NULL,
  `no_edicion` int(2) DEFAULT NULL,
  PRIMARY KEY (`id_libro`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
AUTO_INCREMENT=1 ;
-- --------------------------------------------------------
-- Estructura de tabla para la tabla `recomendacion`
--
CREATE TABLE IF NOT EXISTS
`recomendacion` (
  `id_recomendacion` int(10) NOT NULL
AUTO_INCREMENT,
  `fecha` datetime NOT NULL,
  `puntuacion` int(2) DEFAULT NULL,
  `descripcion` varchar(255) NOT NULL,
  `email` varchar(70) NOT NULL,
  `id_libro` int(10) NOT NULL,
  PRIMARY KEY (`id_recomendacion`),
  KEY `email` (`email`),
  KEY `id_libro` (`id_libro`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
AUTO_INCREMENT=1 ;
-- --------------------------------------------------------
--
-- Estructura de tabla para la tabla `usuario`
--
CREATE TABLE IF NOT EXISTS `usuario` (
  `email` varchar(70) NOT NULL,
  `password` varchar(16) NOT NULL,
  `nombre` varchar(50) NOT NULL,
  `app_pat` varchar(50) NOT NULL,
  `app_mat` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`email`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Restricciones para tablas volcadas
--


--
-- Filtros para la tabla `escrito_libros`
--
ALTER TABLE `escrito_libros`
  ADD CONSTRAINT `escrito_libros_ibfk_1` FOREIGN KEY
(`id_libro`) REFERENCES `libro` (`id_libro`) ON DELETE
NO ACTION ON UPDATE NO ACTION,
ADD CONSTRAINT `escrito_libros_ibfk_2` FOREIGN KEY
(`id_autor`) REFERENCES `autor` (`id_autor`) ON DELETE
NO ACTION ON UPDATE NO ACTION;

--
-- Filtros para la tabla `recomendacion`
--
ALTER TABLE `recomendacion`
  ADD CONSTRAINT `recomendacion_ibfk_1` FOREIGN
KEY (`email`) REFERENCES `usuario` (`email`) ON
DELETE NO ACTION ON UPDATE NO ACTION,
  ADD CONSTRAINT `recomendacion_ibfk_2` FOREIGN
KEY (`id_libro`) REFERENCES `libro` (`id_libro`) ON
DELETE NO ACTION ON UPDATE NO ACTION;
```

# CAPÍTULO 5: IMPLEMENTACIÓN Y PRUEBAS DEL SISTEMA

## 5.1 Implementación

A continuación se describirá brevemente los pasos que se siguieron para implementar el sistema.

### 5.1.1 Implementación de la interfaz

Para realizar el marco de la página web se recurrió a una plantilla web gratuita disponible en la dirección http://www.templatemo.com/ (véase figura 5.1) y se le realizaron los cambios pertinentes para cubrir con los requerimientos antes mencionados en el Capitulo 3 Análisis y Diseño.


Figura 5.1 8Pantalla de la plantilla web original

El marco quedó con opciones como Inicio, Iniciar Sesión, Libros y Acerca de, además de links a redes Sociales. Por lo que se decidió titular a la página web como "YaLoLei.com" para atraer a la mayor cantidad de Usuarios.


Figura 5.2 Pantalla del esqueleto de YaLoLei.com

La página web cuenta con cuatro pestañas principales: Inicio, Iniciar Sesión, Mi Perfil, Libros y Acerca de.

En la página Inicio se presenta una descripción breve del propósito del sistema (véase figura 5.2).

En la página Iniciar Sesión (véase figura 5.3) se presenta los campos para iniciar sesión una vez que ya se ha registrado, además un link a la página de registro.

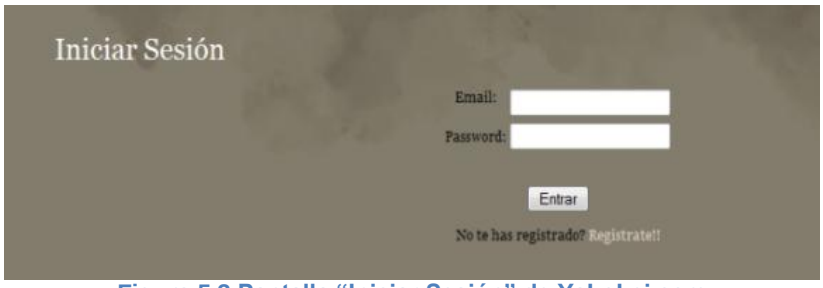


Figura 5.3 Pantalla “Iniciar Sesión” de YaLoLei.com

La página Registro muestra un formulario a llenar con un “Captcha” para validar a la persona (véase figura 5.4).


Figura 5.4 Pantalla "Registro" de YaLoLei.com

Una vez que se ha iniciado sesión, esta liga cambia por "Mi perfil", el cual muestra los datos del usuario (véase figura 5.5).


Figura 5.5 Pantalla "Mi Perfil" de YaLoLei.com

La pestaña Libros le proporciona al usuario diversas pantallas, tales como Buscar Libro (figura 5.6), donde hay diversas formas de realizar una búsqueda del sistema ya que se pueden realizar búsquedas por palabras clave, ver todos los libros, ver todos en orden alfabético ya sea por título o por autor.

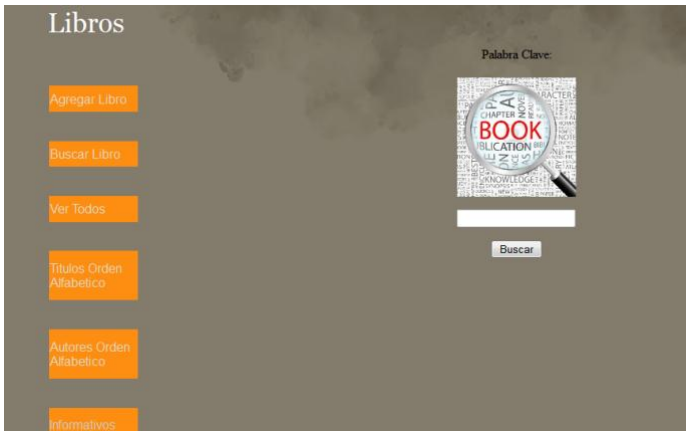


Figura 5.6 Pantalla "Libros-Buscar" de YaLoLei.com

En la figura 5.7 se puede apreciar el resultado de la búsqueda alfabética por autor, donde el resultado es dividido en páginas donde se despliegan 5 libros por página.


Figura 5.7 Pantalla "Libros-VerTodos" de YaLoLei.com

La funcionalidad de Agregar Libro (figura 5.8), despliega un formulario a llenar con los campos necesarios para registrar un libro.

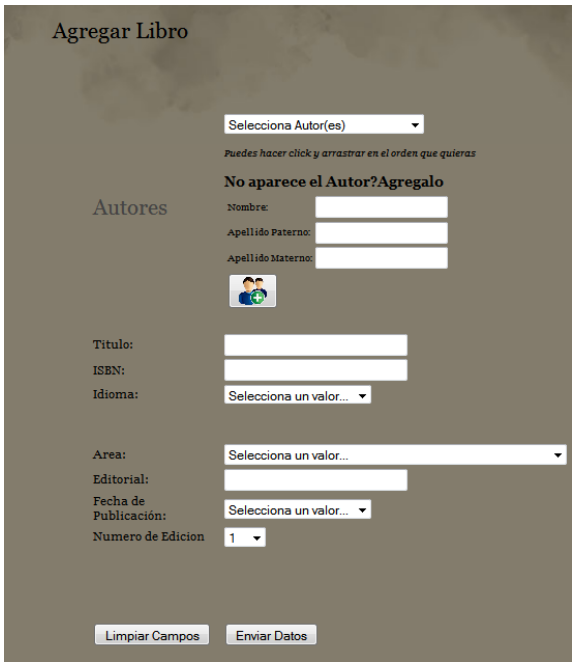


Figura 5.8 Pantalla "Libros-Agregar Libro" de YaLoLei.com

Una vez que se encuentra el libro que se busca, basta con hacer click sobre él en la lista y aparecerá el resultado de todas las recomendaciones que hay acerca de ese libro (véase figura 5.9).

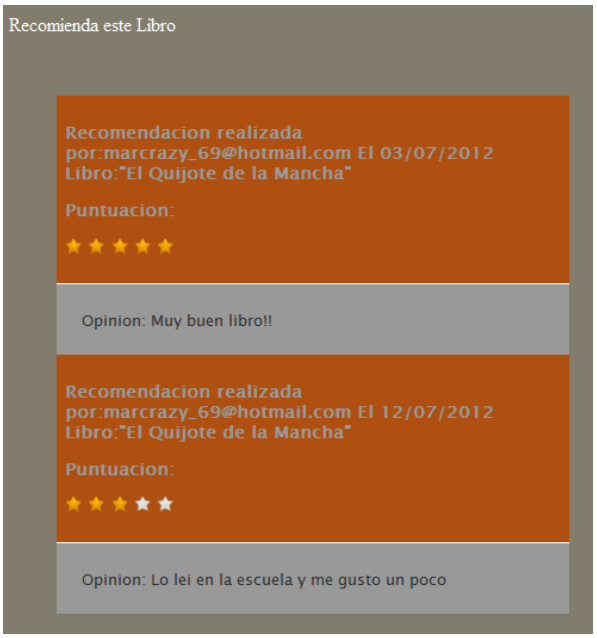


Figura 5.9 Pantalla "Libros-Recomendaciones" de YaLoLei.com

Y si deseamos agregar una recomendación, hacemos click sobre la liga a “Agregar Recomendacion” y aparece la siguiente pantalla (véase figura 5.10).


Figura 5.10 Pantalla “Realizar Recomendación” de YaLoLei.com

La pestaña Acerca de provee al usuario una descripción y justificación del trabajo realizado en este sistema (véase figura 5.11).


Figura 5.11 Pantalla “Acerca de” de YaLoLei.com

### 5.1.2 Implementación de las Funcionalidades

El sistema cumple las Funcionalidades especificadas en los requerimientos mencionados en el apartado 3.6.3, que después fueron representados en el Diagrama de Casos de Uso (véase figura 3.1) y también fueron implementadas las clases definidas en el Diagrama de Clases (figura 3.9) ya que se utilizó el paradigma de Programación Orientada a Objetos, dichas clases fueron de mucha utilidad para cumplir con las funcionalidades del sistema.

- Iniciar Sesión

  Para realizar el inicio de sesión, se implementó un Procedimiento Almacenado que recibe de parámetro el correo del usuario registrado (figura 5.12) y devuelve el correo y contraseña encriptada si existe o no devuelve nada si el correo no existe en la base de datos.

```
call login(
  "marcrazy_69@hotmail.com"
)
```

Figura 5.12 Ejecución del procedimiento "login"

- Buscar Libro

  Para cumplir con este requerimiento se implementaron diferentes modos de búsqueda de libros, ya sea por título, autor, palabra clave, categoría y ver todos. Estas búsquedas se realizan por medio de procedimientos almacenados los cuales reciben diferentes parámetros para buscar en filas de la tabla "Libro" y cumplir con el tipo de búsqueda requerida.

  En la figura 5.13 se aprecia los datos que devuelve el procedimiento "muestra_libros_autor" que realiza una búsqueda por apellido o nombre del autor.

```
call muestra_libros_autor(
  "Tolkien"
)
```

| id_libro | titulo | isbn | idioma | area | editorial | fecha_publicacion | no_edicion |
|---|---|---|---|---|---|---|---|
| 44 | The Hobbit | 9780618260300 | Ingles | LITERARIOS/Narrativa de ciencia ficcion | Houghton Mifflin Har | 2002 | 4 |

Figura 9Ejecución del procedimiento "muestra_libro_autor"

- Registrar Libro

Esta funcionalidad se realiza obteniendo los campos requeridos en la interfaz y mandándolos como parámetros a un procedimiento que se encarga de inserta un nuevo registro en la tabla “Libro”.

En la figura 5.14 se aprecia una inserción.

```
call inserta_libro(
  "CREPUSCULO", "9788420469287", "Español", "LITERARIOS/Narrativa de ciencia ficcion", "Alfaguara", 2006, 1
)
```

Figura 5.14 Ejecución del procedimiento “inserta_libro”

- Registrar Usuario

  Este requerimiento se realiza llamando a un procedimiento con los parámetros necesarios para la inserción en la tabla “Usuario”.

  Se puede apreciar en la figura como se ha insertado un nuevo usuario en la tabla.

```
call inserta_usuario(
  "luna_sol@gmail.com", "miContraseña", "Maria", "Perez", "Perez"
)
```

Figura 5.15 Ejecución del procedimiento “inserta_usuario”

- Modificar Perfil

  Una vez que se ha iniciado sesión, se llena el formulario con los datos actuales en la interfaz, el usuario cambiará los que sea necesario y se realiza una actualización de los datos en el registro de la tabla “Usuario”. Se realiza mediante un procedimiento que recibe todos los datos del usuario como parámetros y hace el “Update” en la tabla.

  En la figura 5.16 se aprecia los datos del usuario antes y después de haber sido modificados.

| | | | | |
|---|---|---|---|---|
| luna_sol@gmail.com | miContraseña | Maria | Perez | Perez |

```
call modifica_usuario(
  "luna_sol@gmail.com", "miContraseña2", "Maria Maria", "Perez", "Sanchez"
)
```

| | | | | |
|---|---|---|---|---|
| luna_sol@gmail.com | miContraseña2 | Maria Maria | Perez | Sanchez |

Figura 5.16 Ejecución del procedimiento "modifica_usuario"

- Recomendar

  Esta funcionalidad permite que el usuario pueda agregar una puntuación y opinión acerca de un libro en específico. Para realizar esto es necesario datos del libro a recomendar, el usuario que realiza la recomendación y la opinión acerca del libro. Todo lo anterior se envía como parámetros a un procedimiento que realiza la inserción en la tabla "Recomendación" (véase figura 5.17).

```
call inserta_recomendacion( 5, "El mejor libro de la saga", "luna_sol@gmail.com", 50 )
```

Figura 5.17 Ejecución del procedimiento "inserta_recomendacion"

## 5.2 Integrando Seguridad en el Sistema

Se integraron aspectos de seguridad en el sistema, principalmente seguridad en la base de datos y seguridad en la interfaz.

### 5.2.1 Seguridad en la Base de Datos

En la base de datos se consideraron aspectos como cambio de contraseña para el usuario administrador "root" (véase figura 5.18).

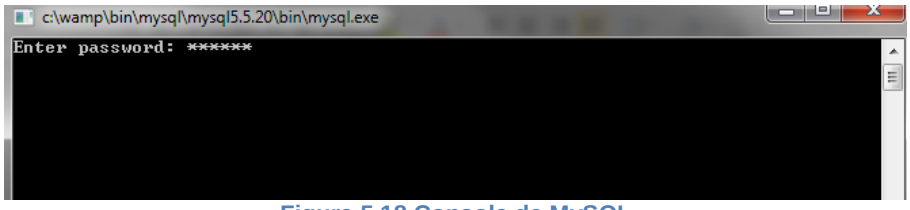


Figura 5.18 Consola de MySQL

Creación de usuarios con permisos limitados para realizar consultas e inserciones en la base de datos. En la figura 5.19 se aprecian los usuarios y privilegios en la base de datos.

| | | | | | |
|---|---|---|---|---|---|
| ☐ | martin3 | localhost | Sí | ALL PRIVILEGES | Sí |
| ☐ | root | 127.0.0.1 | No | ALL PRIVILEGES | Sí |
| ☐ | root | ::1 | No | ALL PRIVILEGES | Sí |
| ☐ | root | localhost | Sí | ALL PRIVILEGES | Sí |
| ☐ | usuario | localhost | Sí | SELECT, INSERT, UPDATE, DELETE, EXECUTE | No |

Figura 5.19 Tabla de usuarios y privilegios en el motor de base de datos

Encriptación de contraseñas de usuarios mediante el algoritmo DES para evitar que los administradores o usuarios de la base de datos vean contraseñas ajenas.

En la figura 5.20 se aprecian la tabla usuarios de la base de datos, donde la segunda columna muestra las contraseñas encriptadas.

| | | | | |
|---|---|---|---|---|
| marcrazy_69@hotmail.com | €÷²¦ÝyÙÛ | Martin | Gonzalez | Escamilla |
| net.cafe.zacatlan@gmail.com | €ïnqxÜVhö[OÐýF | meee | goo | esss |

Figura 5.20 Tabla de usuarios registrados en la base de datos del sistema.

### 5.2.2 Seguridad en la Interfaz

En la interfaz se implementó validación de los campos de formularios, se verifica que no estén vacíos y que tengan un formato correcto (véase figura 5.21).

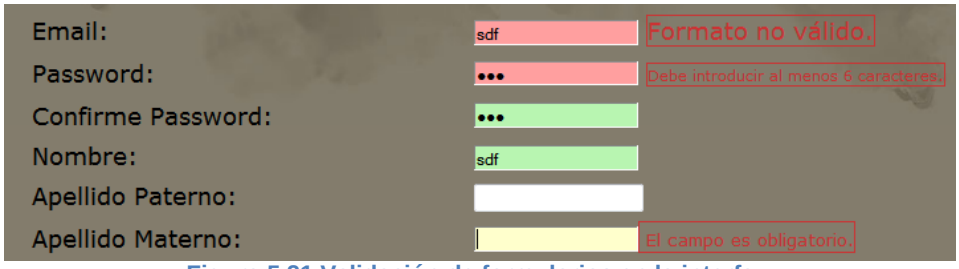


Figura 5.21 Validación de formularios en la interfaz.

Se implementaron “Captchas” para validar que es una persona quien hace una inserción (véase figura 5.22).

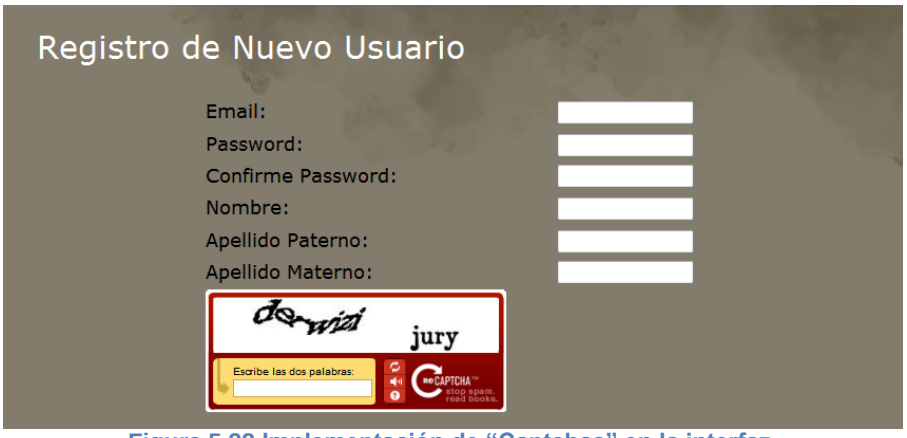


Figura 5.22 Implementación de "Captchas" en la interfaz.

Se encripta la contraseña para compararla con la base de datos y para que esté segura en la dirección URL.

## 5.3 Pruebas del sistema

Para verificar el correcto funcionamiento del sistema se decidió evaluar los dos puntos más importantes la interfaz y la funcionalidad. Para realizar pruebas de funcionalidad se optó por emplear el método de "Caja Negra" el cual se encarga de comprobar que la función que se realiza correctamente [20].

Para realizar pruebas en la interfaz se empleo el método de "Pruebas de Usabilidad" que consiste presentarle la aplicación a un usuario final, realizarle un cuestionario acerca de su experiencia utilizando el sistema y en base a ello realizar las adecuaciones necesarias [21].

### 5.3.1 Pruebas de Caja Negra

Mediante pruebas de caja negra se evaluaron cada una de las funciones especificadas en el diagrama de casos de uso (véase figura 3.1).

- Iniciar Sesión

Se realizó la prueba de funcionalidad de caja negra desde la interfaz, se probo que funcionara correctamente, al introducir tus datos y te direccionará al inicio con una sesion iniciada (véase figura 5.23).

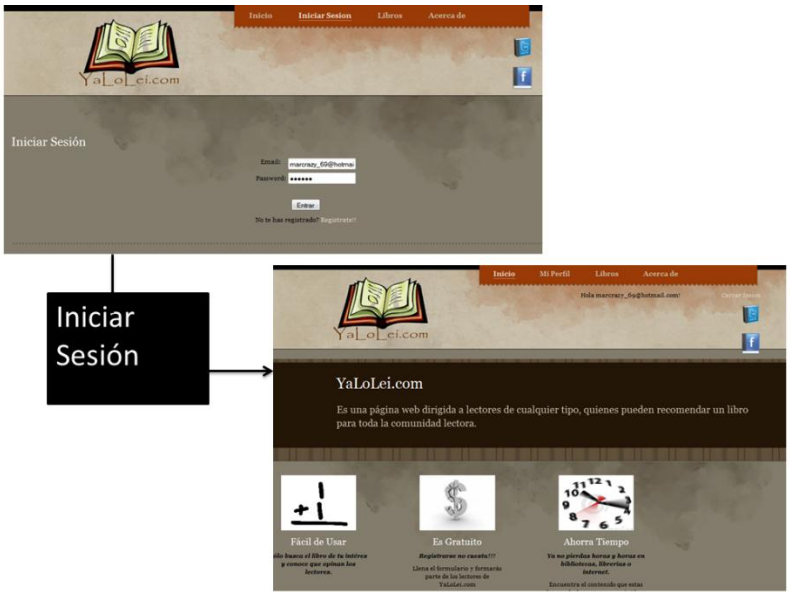


Figura 5.23 Prueba de Caja Negra en la función "Iniciar Sesión".

- Buscar Libro

Se realizaron pruebas con los diferentes modos de búsqueda de libros, en la figura de abajo se aprecia la búsqueda con palabra clave se hace una búsqueda y despliega los libros que cuenten con esta palabra (véase figura 5.24).


Figura 5.24 Prueba de Caja Negra en la función "Buscar Libro".

- Registrar Libro

Se realizaron pruebas de la inserción del libro, el cual al llenar todo el formulario correctamente y dar click en "Guardar" muestra un mensaje que el libro se ha guardado correctamente, de lo contrario, muestra otro indicando que hubo un error en el proceso (véase figura 5.25).


Figura 5.25 Prueba de Caja Negra en la función "Registrar Libro".

- Registrar Usuario

Se verifico que el registro de usuario funcionara correctamente y para ello se debe llenar todo el formulario correctamente y escribir la palabra del "Captcha" y si todo es correcto al presionar "Registrarse" aparecerá un mensaje que indica que se registró correctamente (véase figura 5.26).


Figura 5.26 Prueba de Caja Negra en la función "Registrar Usuario".

- Modificar Perfil

Se realizaron pruebas para comprobar que se actualizaban los datos del usuario y desplegaran en pantalla correctamente, una vez que se modificó algún campo y se dio click en Guardar aparece un mensaje indicando que los datos han sido actualizados (véase figura 5.27).


Figura 5.27 Prueba de Caja Negra en la función "Modificar Perfil".

- Recomendar

Se probó que las recomendaciones se guardaran correctamente con los datos del libro y del usuario, para mantener la integridad en la base de datos. Una vez que se realiza un recomendación se muestra un mensaje indicando que se guardo correctamente (véase figura 5.28).


Figura 5.28 Prueba de Caja Negra en la función "Recomendar".

### 5.3.2 Pruebas de Usabilidad

Para realizar esta prueba se presentó el sistema a tres personas y se le realizó el siguiente cuestionario

**Persona 1**

Nombre: **Aldo González Escamilla**
Edad: **20**
Sexo: **Hombre**
Estudios: **Lic. Comercio Internacional**
Conocimientos de computación: **Medio**
Preguntas Realizadas al usuario:
¿Qué te pareció el sistema? **Es una página que tiene un propósito necesario y que no existe actualmente.**
¿Te pareció difícil usarlo? **No.**
¿Tienes alguna sugerencia para mejorarlo? **Cambiar los colores y si se pudiera ligar a redes sociales para que sepan que opino de un libro estaría bien**.

**Persona 2**

Nombre: **Ana Karen González Velázquez**
Edad: **20**
Sexo: **Mujer**
Estudios: **Preparatoria**
Conocimientos de computación: **Básico**
Preguntas Realizadas al usuario:
¿Qué te pareció el sistema? **Es una página bonita.**
¿Te pareció difícil usarlo? **No**
¿Tienes alguna sugerencia para mejorarlo? **Todo me pareció muy bien.**

**Persona 3**

Nombre: **Selene María Andraca Velázquez**
Edad: **22**
Sexo: **Mujer**
Estudios: **Ing. En Ciencias de la Computación.**
Conocimientos de computación: **Avanzado**
Preguntas Realizadas al usuario:
¿Qué te pareció el sistema? **Es un sistema pequeño pero completo, con validaciones y una interfaz amigable.**
¿Te pareció difícil usarlo? **No.**
¿Tienes alguna sugerencia para mejorarlo? **Si, podrías agregar imágenes pequeñas de los libros para que se identifique más.**

# CAPÍTULO 6: CONCLUSIONES Y TRABAJOS A FUTURO

## 6.1. Aportaciones

Como aportaciones de esta Tesis, se podrían citar las siguientes:

- Se generó un sistema que permite hacer recomendaciones referentes a libros de distintas disciplinas y editoriales para así considerar su consulta, uso o compra de aquellos recomendados que cumplan con las necesidades del lector.
- El sistema es una aplicación Web, la cual es accesible el público en general y solo se necesita contar con una computadora y conexión a internet. Ei sistema cuenta con una interfaz amigable, ya que la página va dirigida a personas de cualquier edad o profesión.
- El sistema cubre con la necesidad que presentamos la mayoría de los estudiantes así como profesores, para buscar y encontrar un libro adecuado a la información requerida.

## 6.2. Trabajo a Futuro

Algunas funcionalidades que se pueden agregar a futuro para mejorar el sistema y que no se consideraron son:

- Confirmación de Registro vía Email. Enviar un correo electrónico al usuario para confirmar sus datos y finalizar el registro es una funcionalidad implementada en muchos sistemas y sirve para comprobar la identidad y la veracidad de los datos registrados por parte del usuario.
- Conexión a Librerías y Bibliotecas en Línea. Una vez que consultaste lo que opinan y recomiendan los usuarios del libro de tu interés, se podría contar con la opción de tener una liga hacia una biblioteca en línea para consultar el libro o una librería en línea para comprarlo.
- Foro para Intercambio literario. Contar con un foro donde los usuarios convivan, se comuniquen e intercambien libros entre ellos.

## 6.3. Conclusiones Finales

En esencia, la Tesis se encaminó a proponer el análisis, diseño e implementación de un sistema para recomendación de libros.

En la etapa de análisis se identificaron los requerimientos principales del sistema, los cuales son muy importantes para el planteamiento y desarrollo de un sistema.

En la fase de diseño se desarrollaron diagramas para representar y descomponer el sistema en pequeños módulos o funcionalidades lo cual facilita enormemente el entendimiento e implementación del sistema.

Posteriormente se inició con la implementación del sistema, la cuál se facilitó demasiado debido a que en base a los diagramas desde los de Caso de Uso para representar funcionalidades hasta el de Entidad Relación para crear la base de datos, se creó lo necesario sin tantas regresiones ni modificaciones en lo diseñado inicialmente.

# BIBLIOGRAFÍA


[1] Ian Sommerville, “Ingeniería de Software.”, Addison-Wesley, 2005.

[2] Craig Larman. “UML y Patrones, Introducción al Análisis y Diseño Orientado a Objetos”, PHH, 2007.

[3] Booch G. “Software Architecture and the UML”, Addison-Wesley, 1998.

[4] Greiff W. R. “Paradigma vs Metodología; El Caso de la POO”, Alfa-Omega, 1994.

[5] Boehm B, “A Spiral Model of Software Development and Enhancement”, IEEE Computer, 1988.

[6] Michael Haug et al,”Developing a Software Project Life Cycle Process”, Springer-Verlag, 2001.

[7] Ivar Jacobson, Grady Booch, “El Proceso Unificado de Desarrollo de Software”, Addison-Wesley, 1999.

[8] Watts S. Humphrey, “Personal Software Process”, Addison-Wesley, 1997.

[9] Nicolas Marin, “Introducción a las Bases de Datos”,: Thomson, 2005.

[10] Sergio Ezequiel, “Base de datos y sus aplicaciones en SQL”, M.P. Ediciones, 2004.

[11] Elisa Bertino, “Intelligent Database Systems”, ACM Press, 2001.

[12] Abraham Silberschatz, “Fundamentos De Bases De Datos”, McGRAW-HILL, 2002.

[13] María Elena Rodríguez et al, “Diseño Y Administración De Bases De Datos”, Aula Politècnica, 2006.

[14] Ivar Jacobson, “Object-Oriented Software Engineering; A Use Case Driven Aproach”, Addison-Wesley, 1994.

[15] Mohammed J. Kabir, “La Biblia De Servidor Apache”, Anaya Multimedia, 2001.

[16] John Griffin, “Creación de sitios Web con Xml y SQL 2000”, Pearson, 2002.

[17] Christophe Aubry, “PHP/MySQL con Dreamweaver CS4”, ENI, 2010.

[18] Olivier Hertuel, “PHP 5.3”, Ediciones ENI, 2011.

[19] Michael Kruckenberg, “PRO MySQL”, Springer, 2006.

[20] Bolaños Alonso, “Pruebas De Software Y Junit”, Pearson España, 2007.

[21] Mario G. Piattini Velthuis, “Medición Y Estimación Del Software: Técnicas Y Métodos Para Mejorar La Calidad Y La Productividad”,RA-MA, 2008.