Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Computación

“Aplicación Web para la administración de recursos materiales y académicos de una escuela primaria mediante una metodología de desarrollo de software basada en el Proceso Unificado”

Tesis para obtener el grado de:

Licenciado en Ciencias de la Computación

Presenta:

Andrei Sánchez Castillejos

Director y Asesor de Tesis:

M. C. Miguel Rodríguez Hernández

Septiembre 2020

Contenido

Agradecimientos 4
Prefacio 5
Introducción 6
Capitulo I. Marco teórico 7
1.1 Descripción general 7
1.2 Proceso unificado de desarrollo de software 7
1.3 Diseño de base de datos 11
1.3.1 Modelo entidad relación 11
1.3.1.1 Entidad 11
1.3.1.2 Atributos 12
1.3.1.3 Relación 12
1.3.1.4 Identificadores 12
1.3.2 Normalización 13
1.3.2.1 Dependencia funcional 14
1.3.2.2 Formas normales 15
1.4. Plataformas de implementación 18
1.4.1 Windows 18
1.4.2 IIS (Internet Information Service) 18
1.4.3 SQLServer 19
1.4.4 ASP.NET Core 20
Capitulo II. Análisis y diseño del sistema 21
2.1 Requisitos funcionales 21
2.2 Requisitos no funcionales 22
2.3 Análisis del sistema 23
2.3.1 Diagrama de casos de uso 23
2.3.2 Especificaciones de casos de uso 24
2.4 Diseño de base de datos 30
Capitulo III. Implementación y pruebas 32
3.1 Inicio de sesión 32
3.2 Recuperación de contraseña 33
3.3 Cambiar contraseña 34

3.4 Cerrar sesión 35
3.5 Capturar periodo escolar 36
3.6 Capturar materias 37
3.7 Capturar usuario 38
3.8 Capturar alumno 39
3.9 Capturar tutor 40
3.10 Asignación de curso 41
3.11 Inscribir alumno a curso 42
3.12 Consultar estadísticas 43
3.13 Capturar calificaciones 44
3.14 Capturar libro 45
3.15 Registrar código de libro 46
3.16 Prestar libro 48
3.17 Devolver libro 49
3.18 Pruebas de caja negra 51
3.19 Pruebas de caja blanca 52
3.20 Resultados de aplicar las pruebas 53
Conclusiones 54
Bibliografía 55

Agradecimientos

A Dios

Por la oportunidad de vivir y por todas sus bendiciones otorgadas.

A mi mamá

Por todo su apoyo y esfuerzo incondicional para brindarme la oportunidad de cumplir mis metas, por ser mi ejemplo a seguir, por sus enseñanzas, buenos valores y sacarnos siempre adelante sin importar las circunstancias. Te amo mamá.

A mi papá

Por todas sus enseñanzas, porque me cuida desde el cielo y por qué sé que siempre fui un orgullo para él.

A mis hermanos

Por todo su apoyo y sacrificios realizados para verme cumplir mis metas, por toda la confianza depositada en mí y salir adelante juntos. Por estar siempre a mi lado en todo momento sin importar la situación.

A mi cuñado

Por ser un gran ejemplo en mi vida, por todas sus enseñanzas, apoyo, paciencia y lo más importante siempre creer en mí.

A mi asesor

Por el gran apoyo brindado desde el primer día que lo conocí como profesor, por compartir sus conocimientos, paciencia y tiempo para lograr este trabajo.

A mis sinodales

Por el apoyo brindado, conocimientos y tiempo para revisar y aportar su experiencia en este trabajo.

A mis profesores

Por compartir su conocimiento a lo largo de mi formación académica, por cada granito de arena que cada uno de ustedes aporto para lograr que este trabajo fuera posible.

Prefacio

El siguiente trabajo se encuentra enfocado al desarrollo de una aplicación web orientada a la administración de recursos materiales y personal académico de una escuela primaria.

En esta aplicación se propone llevar el control interno administrativo de una escuela primaria, proponiendo de una forma sencilla, rápida, ecológica (no papel) y eficaz de la administración de los recursos.

Para el desarrollo de este sistema se analizaron las funciones administrativas susceptibles para ser automatizas.

Este proyecto se sustenta en una metodología de desarrollo de software basada en el proceso unificado.

Mediante la teoría de base de datos relacionales se crean los modelos de entidad-relación para la persistencia de datos.

La naturaleza de la herramienta propuesta será utilizando las tecnologías de desarrollo de software actuales, cuidando el aspecto de inversión económica en medida de lo posible para su elaboración, implementación y manejo del software.

Se utilizara Windows como Sistema operativo, IIS (Internet Information Service) como servidor web, SQLServer como gestor de base de datos, ASP.NET Core como framework basado en el lenguaje de programación C# para desarrollo de las reglas de negocio y HTML para las interfaces de usuario.

Introducción

Una aplicación Web es un sistema de software el cual se accede desde un dispositivo con un navegador web y una conexión a internet, estas aplicaciones constituyen una clase especial de aplicaciones de software que se construyen con ciertas tecnologías y estándares.

La Ingeniería Web tiene como objetivo la aplicación y desarrollo de enfoques de la Ingeniería de Software a las aplicaciones Web, este tipo de aplicaciones ganan un lugar especial dentro de la ingeniería de software debido a sus características.

El contenido de una aplicación web que se presenta al usuario hace necesaria una estrecha colaboración de un equipo que involucra diferentes especialistas, como son ingenieros de software, líderes técnicos, programadores, especialistas en contenido y diseñadores gráficos.

Además de las aplicaciones tradicionales centradas en documentos, existen muchas otras. Por citar sólo algunas, podemos mencionar las aplicaciones interactivas, transaccionales, sociales y colaborativas.

Algunos de los requerimientos de calidad a considerar es la consistencia, confiabilidad, actualización, relevancia entre otros, por tal motivo estos requerimientos se vuelven más complejos.

El contexto social se vuelve extremadamente complejo, ya que el perfil de usuarios es casi imposible de determinar. Hay también diferencias educativas y culturales que hacen que los parámetros de usabilidad se vuelvan relativos. El aspecto de “retención” de un usuario es relevante debido a la facilidad con la que puede dejar de usar un sitio si este no le gusta, es difícil de usar o no cumple con sus expectativas.

El contexto técnico es también complejo. La multiplicidad de plataformas desde las cuales se puede acceder una aplicación Web cambia tanto que es difícil para un usuario normal tener un panorama general. Esto hace que el tradicional “mantra” que se suele invocar cuando se habla de requerimientos (hay que concentrarse en “qué” y no en “cómo”) se vuelve impracticable en el caso de aplicaciones Web por la especial interacción que hay entre los requerimientos y la arquitectura.

Finalmente, la localización geográfica del usuario requiere de las aplicaciones capacidad para adaptarse a diversos contextos locales (por ejemplo, diferencias de hora, idioma, cultura, etc.).

Capitulo I. Marco teórico

1.1 Descripción general

Se asiste a una escuela primaria para analizar el planteamiento del problema que presentan con la administración de recursos materiales y académicos, posterior a la información recabada se debe proponer una solución tecnológica que resuelva sus necesidades más importantes.

En la solución se propone una aplicación web para la gestión administrativa de una escuela primaria, la cual se describen a continuación los módulos más importantes.

- Administración de periodo escolar.
- Administración de Materias.
- Administración de perfil Profesor.
- Administración de Alumnos.
- Administración de tutor.
- Administración de biblioteca.
- Administración de perfil administrador.
- Consulta de estadísticas.

1.2 Proceso unificado de desarrollo de software

El Proceso Unificado es un proceso de software genérico que puede ser utilizado para una gran cantidad de tipos de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de competencia y diferentes tamaños de proyectos.

Provee un enfoque disciplinado en la asignación de tareas y responsabilidades dentro de una organización de desarrollo. Su meta es asegurar la producción de software de muy alta calidad que satisfaga las necesidades de los usuarios finales, dentro de un calendario y presupuesto predecible.

El Proceso Unificado tiene dos dimensiones (Figura 1.1):

- Un eje horizontal que representa el tiempo y muestra los aspectos del ciclo de vida del proceso a lo largo de su desenvolvimiento
- Un eje vertical que representa las disciplinas, las cuales agrupan actividades de una manera lógica de acuerdo a su naturaleza.

La primera dimensión representa el aspecto dinámico del proceso conforme se va desarrollando, se expresa en términos de fases, iteraciones e hitos (milestones).

La segunda dimensión representa el aspecto estático del proceso: cómo es descrito en términos de componentes del proceso, disciplinas, actividades, flujos de trabajo, artefactos y roles.

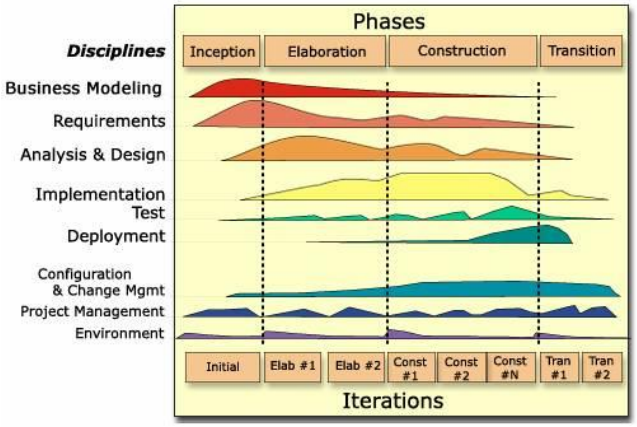


Figura 1.1 Diagrama ilustrativo de proceso unificado [13].

El Proceso Unificado se encuentra basado en componentes, esto quiere decir que el sistema en construcción es desarrollado en componentes de software que se interconectan mediante interfaces bien definidas.

El Proceso Unificado utiliza el Lenguaje de Modelado Unificado (UML) en la preparación de los planos del sistema. UML es una parte integral del Proceso Unificado, ya que se desarrollaron a la par [13].

Lo que hace único el Proceso Unificado son sus tres conceptos claves, dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental [12].

El Proceso Unificado es dirigido por casos de uso

Un sistema de software es construido con la finalidad de servir a sus usuarios. Por tal motivo para que la construcción de un sistema se exitoso se debe conocer a fondo la problemática y la lógica de negocio, esto se traduce en que quieren y que necesitan los usuarios prospectos.

El termino usuario no solo se refiere a los humanos, sino también a otros sistemas, ya que este término representa algo o alguien que interactúa con el sistema por desarrollar.

Un caso de uso es una pieza en la funcionalidad del sistema que le da al usuario un resultado de valor. Los casos de uso capturan los requerimientos funcionales. Todos los casos de uso juntos describe la funcionalidad completa del sistema.

Este modelo reemplaza la especificación funcional tradicional de un sistema, ya que una especificación funcional tradicional se centra en responder una pregunta ¿Qué se supone que el sistema debe hacer?, mientras que la estrategia de casos de uso aumenta tres palabras a la pregunta que se centra en cada usuario ¿Qué se supone que el sistema debe hacer por cada usuario?, estas tres palabras extra, juegan un papel importante, ya que nos obliga a pensar en término del valor a los usuarios y no solamente en término de las funciones.

Sin embargo, los casos de uso no son una herramienta solo para especificar los requerimientos de un sistema, sino también dirigen su diseño, implementación y pruebas.

Aunque los casos de uso dirigen el proceso, no son elegidos de manera aislada. Son desarrollados a la par con la arquitectura del sistema, por lo tanto los casos de uso dirigen la arquitectura del sistema y la arquitectura del sistema influencia la elección de los casos de uso. En consecuencia la arquitectura del sistema y los casos de uso maduran conforme avanza el ciclo de vida [11].

El Proceso Unificado está centrado en la arquitectura

El papel del arquitecto de software es similar al papel que el arquitecto de construcción desempeña. El edificio se mira desde diferentes puntos de vista como es la estructura, servicios, plomería, electricidad, etc. esto permite al constructor ver una radiografía completa antes de empezar la construcción. Lo mismo sucede en la arquitectura de software, ya que es descrita como diferentes vistas del sistema que se está construyendo.

La arquitectura de software es la vista del diseño completo del sistema a construir, dejando los detalles de lado y enfocándose en las características más importantes. También se encuentra influenciado por otros factores como es el S.O en el que se ejecutara, los componentes a ser reutilizados, desempeño, seguridad, confiabilidad entre otros.

Los casos de uso se relacionan con la arquitectura debido a que uno solo de los dos no es suficiente, ya que función corresponde a los casos de uso y forma a la arquitectura. Por lo tanto ambos deben evolucionar en paralelo [11].

El Proceso Unificado es Iterativo e Incremental

El desarrollo de un software comercial es una tarea compleja de realizar, la cual puede continuar por meses o años. Por tal motivo la mejor decisión en su construcción es dividir el trabajo en mini-proyectos. Cada mini-proyecto representa una iteración que se refiere a los pasos en el flujo de trabajo, la cual finaliza en un incremento que se refiere al crecimiento o avance del proyecto. Las iteraciones deben estar controladas, seleccionadas y ejecutadas de forma planeada.

La implementación de una iteración se basa en dos factores. Primero, la iteración trata con un grupo de casos de uso que en conjunto extienden la usabilidad del producto. Segundo, la iteración trata con los riesgos más importantes. Las iteraciones sucesivas construyen los artefactos del desarrollo a partir del estado en el que fueron dejados en la iteración anterior.

En cada iteración se debe identificar y especificar los casos de uso relevantes, la arquitectura se utiliza como guía para la creación del diseño. El diseño se implementa en componentes y se verifica que cada uno de estos satisface cada caso de uso. Para continuar con la siguiente iteración, esta debe cumplir sus metas. Si la iteración no cumple con sus metas, los desarrolladores deben revisar sus decisiones previas y probar un nuevo enfoque [11].

1.3 Diseño de base de datos

El diseño de una base de datos es un proceso complejo que abarca decisiones a muy distintos niveles. La complejidad se controla mejor si se descompone el problema en subproblemas y se resuelve cada uno de estos subproblemas independientemente, utilizando técnicas específicas [2].

1.3.1 Modelo entidad relación

El modelo entidad-relación ER es un modelo de datos que permite representar cualquier abstracción, percepción y conocimiento en un sistema de información formado por un conjunto de objetos denominados entidades y relaciones, incorporando una representación visual conocida como diagrama entidad-relación [3].

1.3.1.1 Entidad

La entidad es cualquier clase de objeto o conjunto de elementos presentes o no, en un contexto determinado dado por el sistema de información o las funciones y procesos que se definen en un plan de automatización. Dicho de otra forma, las entidades las constituyen las tablas de la base de datos que permiten el almacenamiento de los ejemplares o registros del sistema, quedando recogidos bajo la denominación o título de la tabla o entidad. Por ejemplo, la entidad usuarios guarda los datos personales de los usuarios de la biblioteca, la entidad catalogo registra todos los libros catalogados, la entidad circulación todos los libros prestados y devueltos y así sucesivamente con todos los casos [3].

1.3.1.2 Atributos

Son las características, rasgos y propiedades de una entidad, que toman como valor una instancia particular. Es decir, los atributos de una tabla son en realidad sus campos descriptivos, el predicado que permite definir lo que decimos de un determinado sujeto. Por ejemplo de una entidad o tabla catálogo, se pueden determinar los atributos título, subtítulo, título paralelo, otras formas del título, autor principal, otras menciones de responsabilidad, edición, mención de edición, editorial, lugar de publicación, fecha de publicación, etc. [3].

1.3.1.3 Relación

Vínculo que permite definir una dependencia entre los conjuntos de dos o más entidades. Esto es la relación entre la información contenida en los registros de varias tablas. Por ejemplo, los usuarios suelen clasificarse según una lista de tipos de usuarios, ya sean profesores, alumnos o investigadores. De esta forma es posible emitir la relación entre el usuario Jorge Martínez como alumno y Enrique Valtierra como profesor. Las relaciones son definidas de forma natural en un diagrama relacional para expresar un modelo cognitivo que dará lugar posteriormente a las interrelaciones de las entidades [3].

1.3.1.4 Identificadores

Son aquellos que permiten diferenciar a una instancia de la entidad de otra distinta. Por ejemplo, el atributo identificativo que distingue a un alumno de otro es su número de ID.”

“Un identificador de una entidad es un atributo o conjunto de atributos que determina de modo único cada ocurrencia de esa entidad.”

Distinguen a una instancia de la entidad a otra distinta, como también se puede llamar atributos que diferencia la manera única de las ocurrencias, un tipo de entidad puede tener más de un identificador [2].

Identificadores primarios

Son aquellos atributos que permiten identificar una entidad de manera única puede ser simple o compuesta, son referidos como identificadores únicos o claves primarias (PK) de una entidad como se describe en la Figura 1.2 Modelo lógico E-R (entidad relación) [2].

Identificadores foráneos

La clave o claves foráneas de una tabla son referencias a registros de otra tabla, formándose entre ambas tablas una relación. Una registro de la tabla que tiene la clave foránea, llamémoslo registro hijo, apunta a un solo registro de la tabla a la que hace referencia, llamémoslo registro padre. Por tanto, una clave foránea apuntará siempre a la clave primaria de otra tabla.

De hecho el nombre ya nos indica que es una clave externa, es decir, el valor que contiene un registro en el campo, o campos, que ejercen de clave foránea, deberá contenerlo algún registro (uno solo) en el campo, o campos, que ejercen de clave primaria en la tabla a la que hace referencia dicha clave foránea como de describe en la figura a continuación [4].

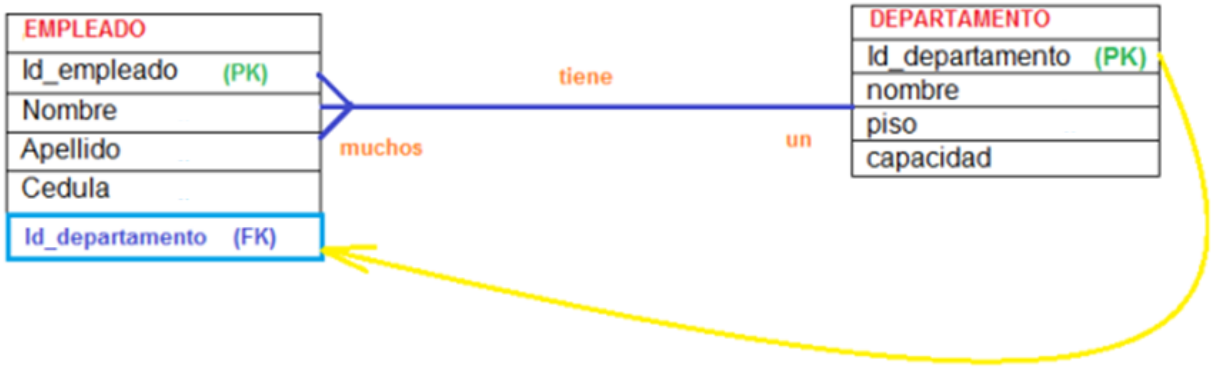


Figura 1.2 Modelo lógico E-R (entidad relación) [2].

1.3.2 Normalización

La normalización de bases de datos es un proceso que consiste en designar y aplicar una serie de reglas a las relaciones obtenidas tras el paso del modelo entidad-relación al

modelo relacional. con objeto de minimizar la redundancia de datos, facilitando su gestión posterior.

Las bases de datos relacionales se normalizan para:

- Minimizar la redundancia de los datos.
- Disminuir problemas de actualización de los datos en las tablas.
- Proteger la integridad de datos.

En el modelo relacional es frecuente llamar tabla a una relación; para que una tabla sea considerada como una relación tiene que cumplir con algunas restricciones:

- Cada tabla debe tener su nombre único.
- No puede haber dos filas iguales. No se permiten los duplicados.
- Todos los datos en una columna deben ser del mismo tipo [5].

1.3.2.1 Dependencia funcional

Una dependencia funcional es una relación entre uno o más atributos. Por ejemplo, si se conoce el valor de DNI (Documento Nacional de Identidad-España) tiene una conexión con Apellido o Nombre.

Las dependencias funcionales del sistema se escriben utilizando una flecha, de la siguiente manera:

FechaDeNacimiento -> Edad

De la normalización (lógica) a la implementación (física o real) puede ser sugerible tener estas dependencias funcionales para lograr la eficiencia en las tablas como se muestra en la figura 1.3 Dependencia funcional [5].

Figura 1.3 Dependencia funcional [5].

1.3.2.2 Formas normales

Para evitar la redundancia de datos y obtener un buen desempeño en una base datos, es importante conocer las formas de normalización [6].

Primera forma normal (1FN)

La primera regla de normalización se expresa generalmente en forma de dos indicaciones separadas.

1. Todos los atributos, valores almacenados en las columnas, deben ser indivisibles.
2. No deben existir grupos de valores repetidos.

El valor de una columna debe ser una entidad atómica, indivisible, excluyendo así las dificultades que podría conllevar el tratamiento de un dato formado de varias partes.

Supongamos que tienes en una tabla una columna Dirección para almacenar la dirección completa, dato que se compondría del nombre de la calle, el número exterior, el número interior (puerta), el código postal, el estado y la capital como se muestra en la figura 1.4 Atributo divisible en varias partes [6].

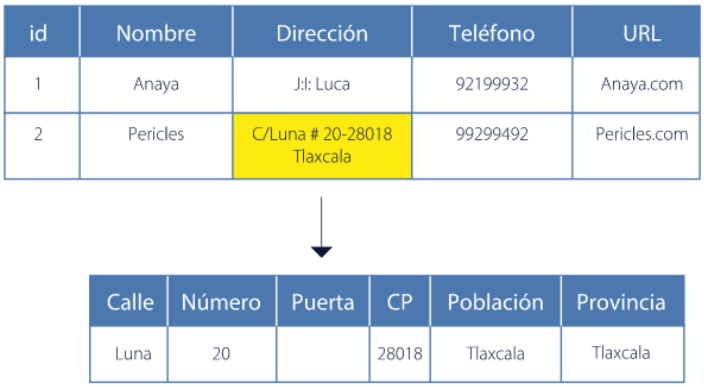


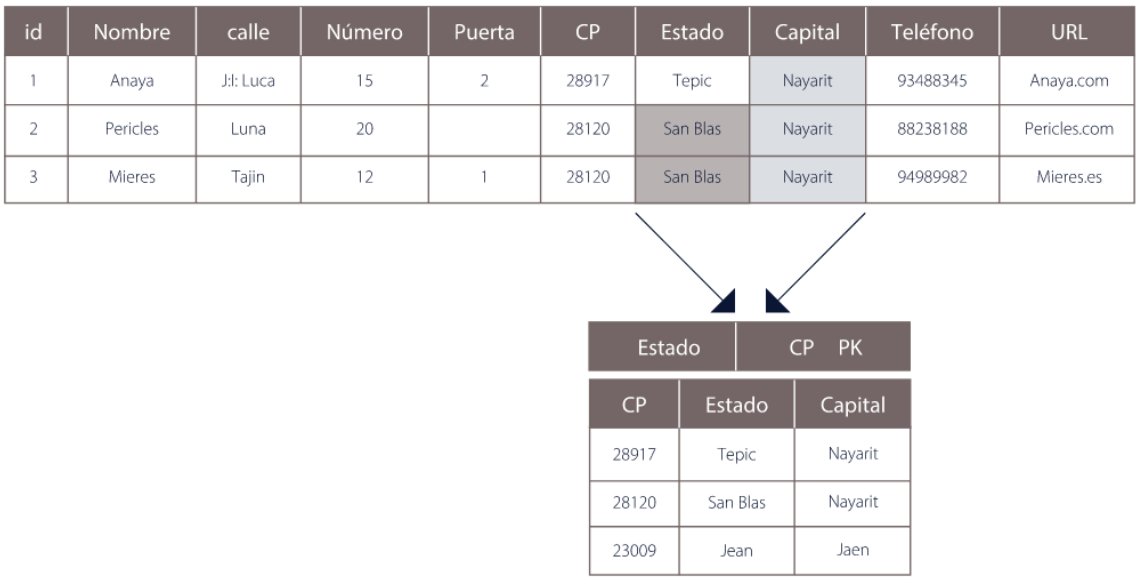
Figura 1.4 Atributo divisible en varias partes [6].

La estructura de esta tabla complica la recuperación de información, debido a que al buscar datos filtrados por calle, población o provincia se vuelve difícil de resolver, por el simple hecho que la dirección es una cadena de caracteres de estructura libe.

Por lo tanto, cada una de las columnas debe contener un valor simple e indivisible el cual debe facilitar las operaciones entes mencionadas.

Debemos evitar que los datos de la población y provincia se repitan en cada fila. Para evitar esto se recomienda realizar una tabla independiente como se muestra en la figura 1.5 Tabla independiente.

Con esto nuestro diseño de base datos se encuentra listo para pasar a la segunda forma normal [6].



id	Nombre	calle	Número	Puerta	CP	Estado	Capital	Teléfono	URL
1	Anaya	J:l: Luca	15	2	28917	Tepic	Nayarit	93488345	Anaya.com
2	Pericles	Luna	20		28120	San Blas	Nayarit	88238188	Pericles.com
3	Mieres	Tajin	12	1	28120	San Blas	Nayarit	94989982	Mieres.es

Estado	CP PK

CP	Estado	Capital
28917	Tepic	Nayarit
28120	San Blas	Nayarit
23009	Jean	Jaen

Figura 1.5 Tabla independiente [6].

Segunda forma normal (2FN)

La segunda forma normal añade la no existencia de dependencias funcionales parciales. Lo que significa que todos los valores de las columnas de una fila dependen de la clave primaria.

Las tablas que están ajustadas a la primera forma normal, y además disponen de una clave primaria formada por una única columna con un valor indivisible, cumplen ya con la segunda forma normal. Ésta afecta exclusivamente a las tablas en las que la clave primaria está formada por los valores de dos o más columnas, debiendo asegurarse, en este caso, que todas las demás columnas son accesibles a través de la clave completa y nunca mediante una parte de esa clave [6].

Tercera forma normal (3FN)

Esta forma normal indica que las columnas que no son parte de la clave primaria deben depender solo de la clave. En conclusión no deben existir dependencias transitivas entre columnas de una tabla [6].

1.4. Plataformas de implementación

1.4.1 Windows

Windows es una palabra del idioma inglés que significa “ventana”. Su utilización en español, en cambio, está casi exclusivamente vinculada a un sistema informático desarrollado por la empresa Microsoft y comercializado desde 1985.

Microsoft Windows es un sistema operativo, es decir, un conjunto de programas que posibilita la administración de los recursos de una computadora. Este tipo de sistemas empieza a trabajar cuando se enciende el equipo para gestionar el hardware a partir desde los niveles más básicos.

Microsoft domina el mercado de los sistemas operativos con comodidad, ya que Windows está instalado en más del 90% de las computadoras con acceso a Internet en todo el mundo.

Entre sus principales aplicaciones (que pueden ser desinstaladas por los usuarios o reemplazadas por otras similares sin que el sistema operativo deje de funcionar), se encuentran el navegador Internet Explorer, el reproductor multimedia Windows Media, el editor de imágenes Paint y el procesador de texto WordPad.

La principal novedad que aportó Windows desde sus orígenes fue su atractivo visual y su facilidad de uso. De hecho, su nombre (“ventanas”) proviene de la forma en que el sistema presenta al usuario los recursos de su computadora, lo que facilita las tareas cotidianas [7].

1.4.2 IIS (Internet Information Service)

Inicialmente lanzado como un grupo de servicios orientado a internet para Windows NT 3.51 en 1995, recién al año siguiente vio la luz Active Server Pages (ASP) para Windows NT 4.0 con IIS 3.0.

Por muchos años IIS fue un CD-ROM separado del Sistema Operativo denominado Option Pack, recién con Windows 2000 se incorporó al instalador del sistema como un paquete más instalable variando las limitaciones dependiendo de la versión de Windows.

Desde 2016 es un servidor que ha venido en crecimiento, las estadísticas de Netcraft lo ubican como el servidor más utilizado como servidor web con cerca de 688 millones de instalaciones.

A diferencia de Apache y Nginx que dividen la carga de trabajo en diversos subprocesos, dependiendo de la configuración incluso un subprocesos por cada request, IIS opta por el modelo de proceso único, es decir que un solo proceso maneja todas las peticiones.

También cabe decir que para determinadas operaciones especiales se delegan a subprocesos, es decir que el proceso principal hace uso de subproceso sin embargo sólo delega un parte de la responsabilidad porque es siempre el proceso principal quien recibe y responde la petición.

Una de las ventajas más importantes de IIS es el manejo de las peticiones y como estas puedes ser atendidas de manera concurrente sobre todo por la alta necesidad que tiene las aplicaciones de realizar peticiones al servidor para enviar y recibir datos, así mismo como importantes mejoras en la entrada y salida asíncrona de datos que permiten aumentar la performance de la aplicación [8].

1.4.3 SQLServer

Microsoft SQL Server es un sistema de gestión de base de datos relacional, desarrollado por la empresa Microsoft.

El lenguaje de desarrollo utilizado es Transact-SQL (TSQL), una implementación del estándar ANSI del lenguaje SQL, utilizado para manipular y recuperar datos (DML), crear tablas y definir relaciones entre ellas (DDL).

SQL Server ha estado tradicionalmente disponible solo para sistemas operativos Windows de Microsoft, pero desde 2016 está disponible para GNU/Linux y a partir de 2017 para Docker.

Características

- Soporte de transacciones.
- Soporta procedimientos almacenados.
- Incluye también un entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.

- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y los terminales o clientes de la red sólo acceden a la información.
- Además permite administrar información de otros servidores de datos.

Este sistema incluye una versión reducida, llamada MSDE con el mismo motor de base de datos pero orientado a proyectos más pequeños, que en sus versiones 2005 y 2008 pasa a ser el SQL Express Edition, que es una edición que se distribuye en forma gratuita [9].

1.4.4 ASP.NET Core

.NET Core es una plataforma de desarrollo de código abierto para uso general. Puede crear aplicaciones de .NET Core para Windows, macOS y Linux para procesadores x64, x86, ARM32 y ARM64 mediante varios lenguajes de programación. Se proporcionan marcos y API para la nube, IoT, la Interfaz de usuario de cliente y el aprendizaje automático [16].

Es multiplataforma debido a su ejecución en varios sistemas operativos. Su código abierto es bajo licencia de MIT y Apache 2. Su posible implementación de programación asíncrona lo hacen moderno [10].

El alto rendimiento, su mismo comportamiento al ser ejecutado en diferentes S.O, la compatibilidad con docker y la variedad de lenguajes como C#, Visual Basic y F#, lo hacen una excelente opción para el desarrollo de aplicaciones [14].

Entity Framework Core

Entity Framework (EF) Core es una versión ligera, extensible, de código abierto y multiplataforma de la popular tecnología de acceso a datos Entity Framework.

EF Core puede servir como asignador relacional de objetos (O/RM), lo que permite a los desarrolladores de .NET trabajar con una base de datos mediante objetos .NET y eliminar la mayoría del código de acceso a los datos que normalmente deben escribir.

EF Core es compatible con muchos motores de base de datos [15].

Capitulo II. Análisis y diseño del sistema

Considerando el proceso unificado (UP) como metodología para el desarrollo de este sistema, procedemos a realizar el análisis y diseño del sistema.

2.1 Requisitos funcionales

El sistema debe tener un control de acceso el cual debe permitir el ingreso a los usuarios por medio de email y contraseña, permitiendo únicamente la manipulación a los módulos asignados de acuerdo a su rol asignado.

El sistema debe permitir realizar consultas, modificaciones, altas y bajas de administrador, alumno, maestro y tutor. En el caso de las bajas únicamente se cambiara el estatus a inactivo para que estos no aparezcan en la gestión diaria, conservando los registros para futuras posibles consultas.

El sistema debe permitir generar nuevos registros en cada catálogo con la finalidad de mantener actualizada la información para cada ciclo escolar.

Se debe contar con un tablero de control, el cual estará conformado de diferentes graficas con la finalidad de observar el comportamiento y obtener estadísticas de acuerdo al aprovechamiento de cada ciclo escolar.

Debe contar con un módulo específicamente para la administración de la biblioteca de la escuela.

2.2 Requisitos no funcionales

Seguridad: La seguridad será a nivel de usuario, aplicando algún algoritmo de encriptación a la contraseña para posteriormente guardarla en la base de datos, para asegurar la confidencialidad.

Disponibilidad: Es suficiente contar con una disponibilidad aceptable debido que el sistema únicamente podrá ser accedido desde la intranet.

Accesibilidad: Únicamente está permitida para los administradores y maestros.

Escalabilidad: En diseño contempla el crecimiento de nuevas funcionalidades.

Portabilidad: El sistema será diseñado para la WEB.

Operatividad: Para que el sistema funcione basta con tener una PC que soporte un navegador WEB.

Interfaz: La interacción del usuario será por medio de una aplicación WEB, a través de menús para acceder a la funcionalidad de cada uno de los módulos, formularios para la captura de información y botones para activar cada uno de los eventos.

2.3 Análisis del sistema

2.3.1 Diagrama de casos de uso

En la figura 2.1 se describe el diagrama general de casos de uso de la aplicación/sistema SchoolManage donde se cuenta con tres actores que son Administrador, Profesor y Bibliotecario.

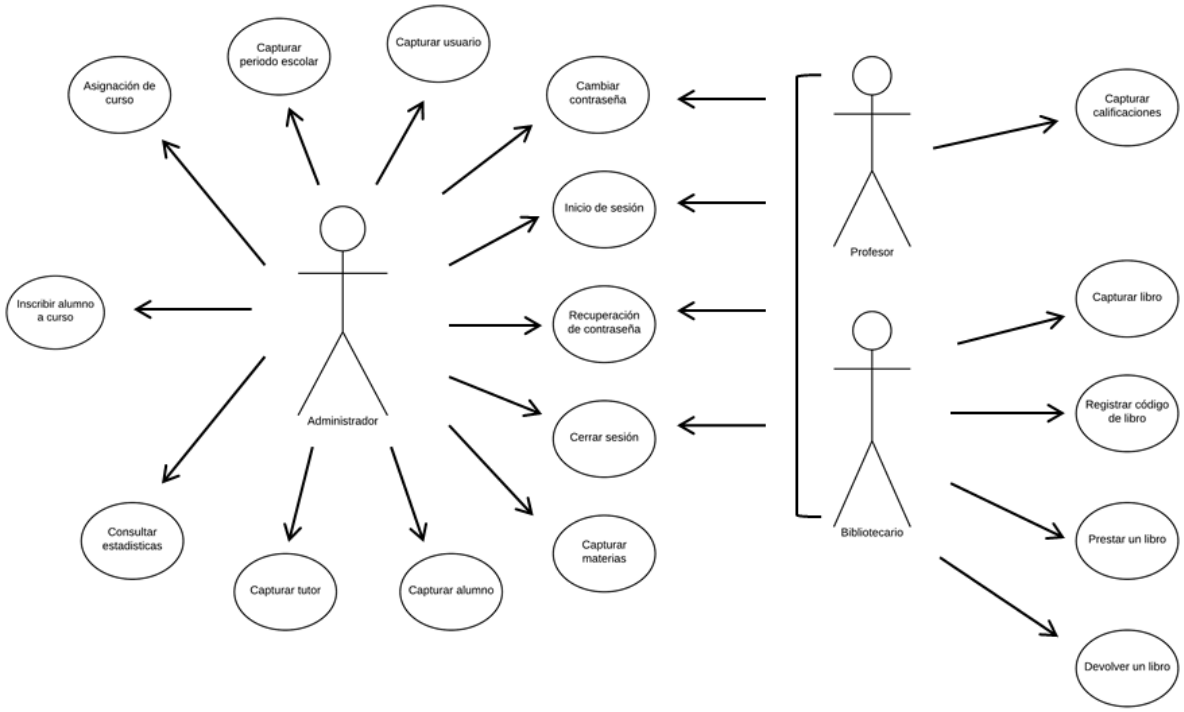


Figura 2.1 Diagrama general de casos de uso.
Fuente creación propia.

2.3.2 Especificaciones de casos de uso

Caso de Uso: Inicio de sesión
Actor: Administrador, Profesor, Bibliotecario
1. El usuario ingresa su email y contraseña en el formulario
2. El sistema valida que las credenciales sean correctas
3. El sistema obtiene su rol y muestra el menú correspondiente

Caso de Uso: Capturar usuario
Actor: Administrador
1. El usuario solicita al administrador una cuenta para poder tener acceso al sistema
2. El administrador llena el formulario y envía la petición
3. El sistema recibe la petición y notifica vía correo electrónico la confirmación de registro y sus credenciales de acceso asignadas
4. El usuario final realiza el cambio de contraseña a la brevedad para asegurar su confidencialidad

Caso de Uso: Capturar periodo escolar
Actor: Administrador
1. Al finalizar un ciclo escolar el usuario captura uno nuevo
2. El sistema obtiene la información básica del periodo escolar, realiza el alta y desactiva el anterior

Caso de Uso: Capturar materias
Actores: Administrador
1. El usuario llena el formulario con el nombre de la materia y el grado al que pertenece
2. El sistema recibe la petición y guarda la información recibida

Caso de Uso: Asignación de curso
Actor: Administrador
1. El profesor solicita la asignación del curso a impartir en el sistema
2. El administrador llena un formulario con el nombre de profesor, grado, grupo y periodo escolar
3. El sistema recibe la información, asigna el nuevo curso al profesor y realiza la baja del curso anterior en caso que exista

Caso de Uso: Capturar alumno
Actor: Administrador
1. El tutor solicita la inscripción de un alumno a la escuela
2. El administrador captura la información del alumno en el formulario y envía la solicitud al sistema
3. El sistema recibe la información y da de alta el alumno

Caso de Uso: Capturar tutor
Actor: Administrador
1. El tutor solicita la inscripción de un alumno
2. El administrador solicita información al tutor para capturar el formulario correspondiente
3. El sistema recibe la información, valida que no se encuentre registrado y procede a realizar el alta en el sistema

Caso de Uso: Inscribir alumno a curso
Actor: Administrador
1. El tutor solicita la inscripción del alumno a un curso
2. El administrador llena un formulario seleccionado el alumno y el curso que el corresponde.
3. El sistema recibe la información, asigna el curso al alumno y realiza la baja del curso anterior en caso que exista

Caso de Uso: Capturar calificaciones
Actor: Profesor
1. El profesor selecciona el bimestre y materia a calificar, posteriormente captura a cada alumno la calificación que le corresponde
2. El sistema recibe la información, valida que la calificación se encuentre entre un rango del 0 y 10 y procede a realizar la asignación de la calificación
3. Se repite al paso 1 hasta que haya calificado todas las materias

Caso de Uso: Capturar libro
Actor: Bibliotecario
1. El usuario recibe un libro nuevo y captura su información
2. El sistema recibe la información y realiza el alta

Caso de Uso: Registrar código de libro
Actor: Bibliotecario
1. El usuario busca el título de un libro
2. Da clic en el botón registrar código y captura el código único del libro físico en el formulario
3. Si existen más libros del mismo título se repite el paso 2
4. Se envía la solicitud al sistema
5. El sistema recibe la información y agrega los códigos al stock del libro seleccionado

Caso de Uso: Prestar un libro
Actor: Bibliotecario
1. El usuario solicita un libro
2. El bibliotecario busca el libro en el sistema por medio del código único, selecciona el nombre del usuario que lo solicita y envía los datos al sistema
3. El sistema procesa los datos recibidos y marca el libro como prestado
4. Se repite el paso 2 hasta que el alumno ya no solicite libros

Caso de Uso: Devolver un libro
Actor: Bibliotecario
1. El usuario desea devolver un libro
2. El bibliotecario busca el libro a ser devuelto en el sistema por medio del código único y selecciona el botón devolver
3. El sistema recibe la petición y marca el libro como disponible
4. Se repite el paso 2 hasta que el usuario no cuete con libros a devolver

Caso de Uso: Consultar estadísticas
Actor: Administrador
1. El administrador desea conocer el aprovechamiento de un ciclo escolar
2. Selecciona el ciclo del cual desea conocer la estadística
3. El sistema recibe la petición, analiza la información solicitada y muestras las graficas correspondientes

Caso de Uso: Cambiar contraseña
Actor: Administrador, Profesor, Bibliotecario
1. El usuario desea cambiar su contraseña actual
2. El usuario llena los campos de su contraseña actual, nueva contraseña y confirmación de nueva contraseña
3. El sistema valida los datos y procesa la solicitud

Caso de Uso: Recuperar contraseña
Actor: Administrador, Profesor, Bibliotecario
1. El usuario no recuerda su contraseña para acceder al sistema
2. Selecciona el link de olvide contraseña desde la pantalla de inicio de sesión, captura su correo electrónico registrado y envía la solicitud
3. El sistema recibe la petición , valida que el correo se encuentre registrado en el sistema y envía un link al correo electrónico para recuperar el acceso
4. El usuario revisa su correo y selecciona el link que lo redirecciona a un formulario para ingresar y confirmar su nueva contraseña
5. El sistema recibe y valida la información para posteriormente aplicar el cambio de contraseña

Caso de Uso: Cerrar sesión
Actor: Administrador, Profesor, Bibliotecario
1. El usuario desea cerrar su sesión en el sistema
2. El sistema recibe y ejecuta la petición para denegar las futuras solicitudes redireccionando a la página de inicio de sesión

Diagrama de secuencia “capturar alumno”

En la figura 2.2 se muestra el diagrama de secuencia donde el administrador crea un alumno en el sistema.

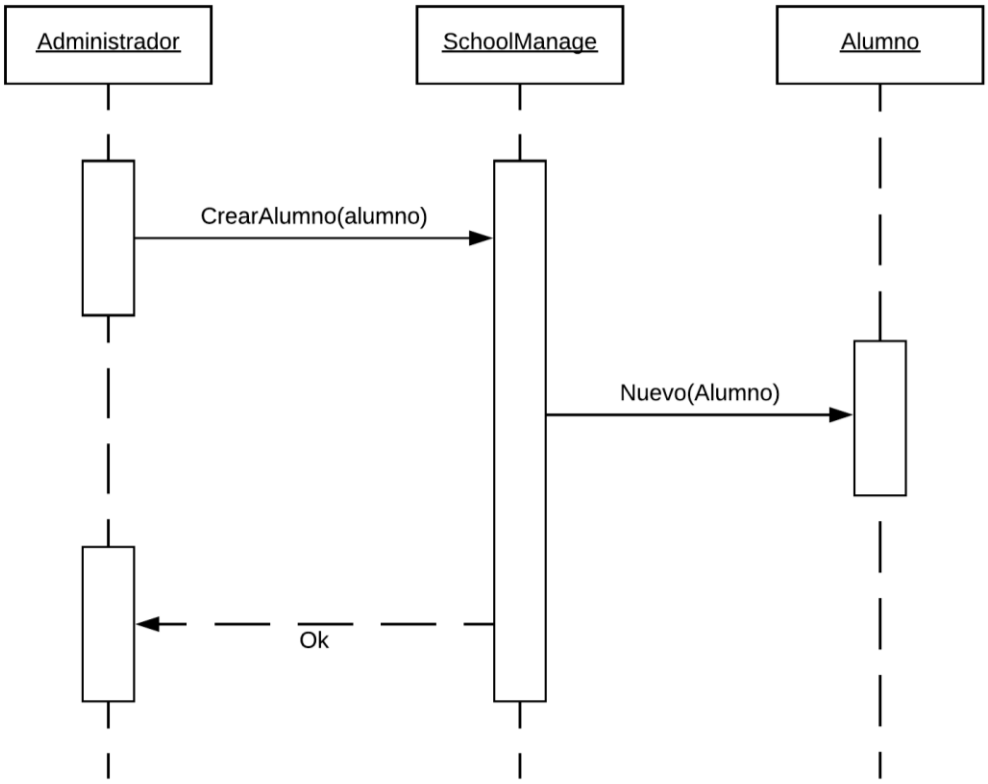


Figura 2.2 Capturar alumno.
Fuente creación propia.

Diagrama de secuencia “asignación de curso”

En la figura 2.3 se muestra el diagrama de secuencia donde el administrador realiza la asignación de un curso al profesor en el sistema.

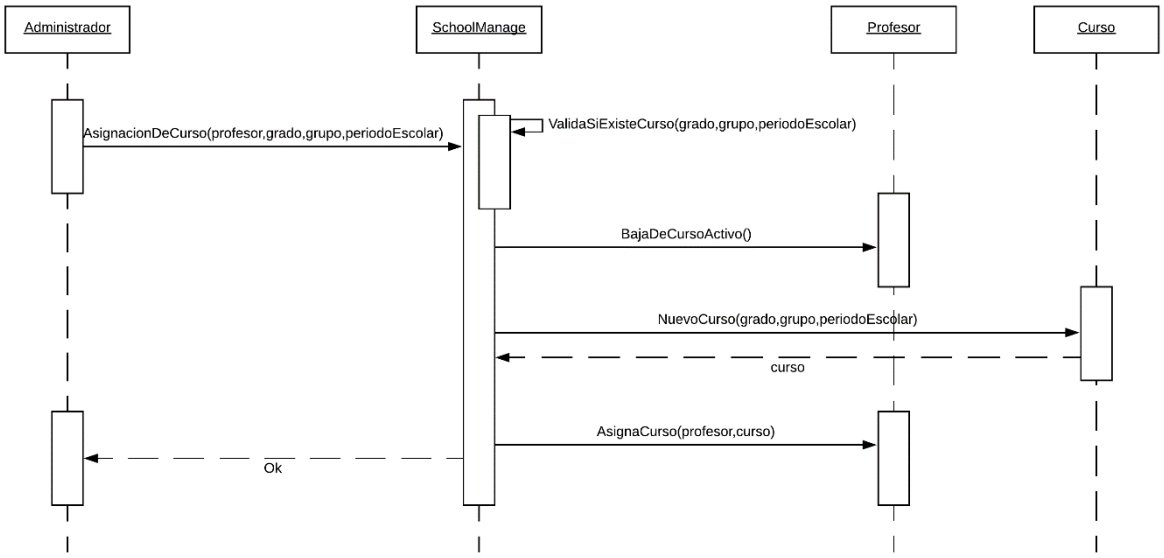


Figura 2.3 Asignación de curso.
Fuente creación propia.

2.4 Diseño de base de datos

Para este diseño se realizó un análisis basado en el planteamiento del problema, aplicando las reglas de normalización.

Se crea una entidad llamada Person conformada con las columnas de los datos que comparten las entidades User, Studen y Tutor, relacionándolos de uno a muchos.

La entidad Address se separa en calle, numero interior, numero exterior, colonia, código postal y municipio con la finalidad de tener valores en las columnas indivisibles, facilitando la recuperación de información.

Se crean tablas independientes (catálogos) en el caso de género, grado, estado, periodo, etc. para evitar que la información en una tabla repita los datos, esto también facilita la búsqueda por medio de alguna de estas entidades, ya que de lo contrario realizar una búsqueda en una secuencia de caracteres con estructura libre sería muy complejo.

Después de analizar los requerimientos del sistema, las entidades resultantes fueron las que se muestran en el diagrama de bases de datos de este trabajo. La base de datos fue normalizada hasta su tercera forma normal.

El siguiente diagrama de la figura 2.4, representa el modelo entidad – relación de la base de datos para el sistema “SchoolManage”.

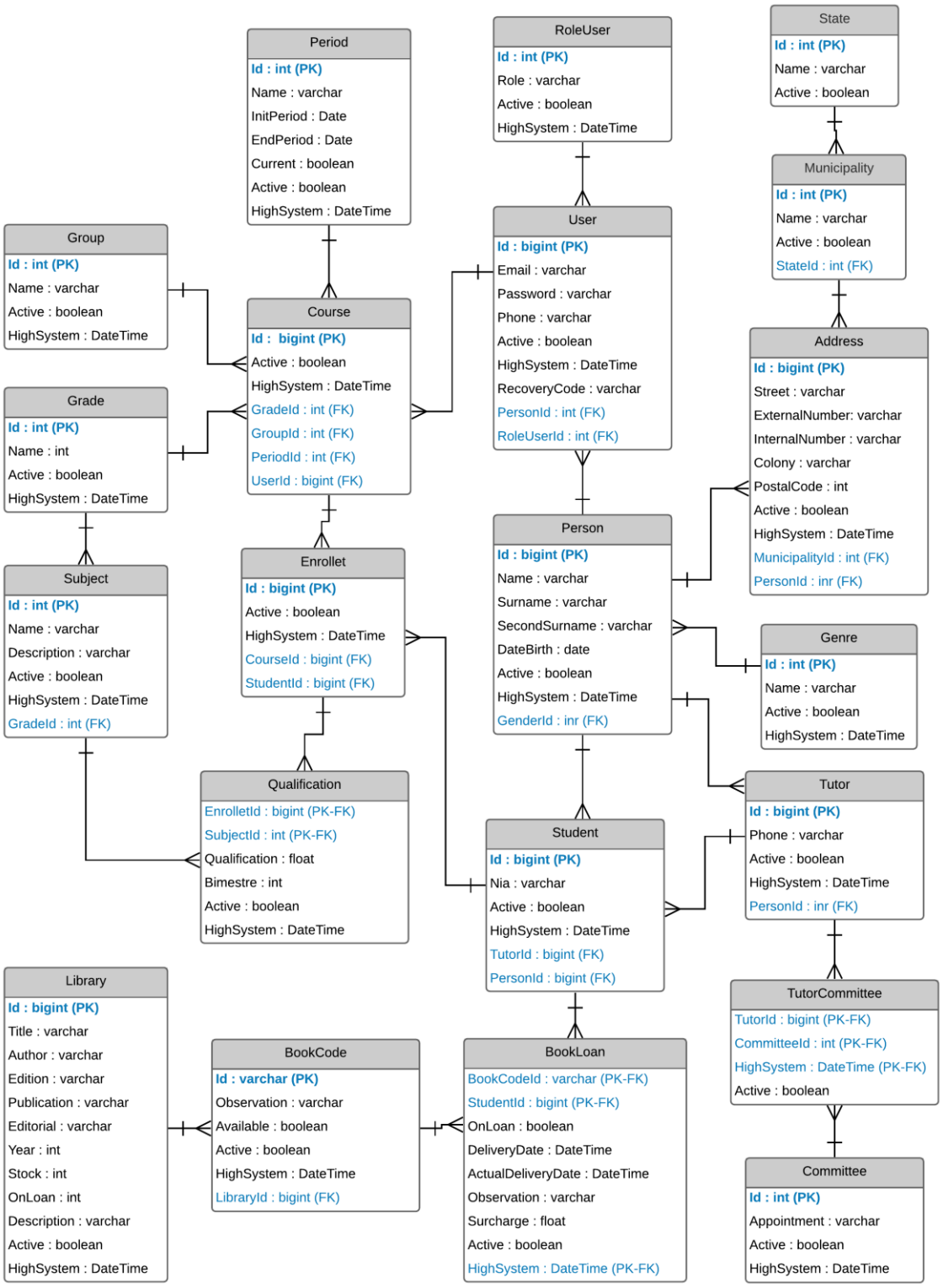


Figura 2.4 Diagrama de base de datos.
Fuente creación propia.

Capitulo III. Implementación y pruebas

3.1 Inicio de sesión

La figura 3.1 representa la pantalla de inicio de sesión del sistema que permite el ingreso al sistema llenando un formulario con los datos correspondientes. Una vez que la plataforma ha brindado acceso al usuario, lo siguiente que hace es validar cada uno de los permisos existentes para conocer si el usuario tiene acceso a ciertas secciones de la plataforma o a información específica de ciertos módulos.

Una de las precondiciones que se deben cumplir es que el usuario debe estar previamente dado de alta en el sistema. El usuario ingresa su correo electrónico de la misma forma que fue registrado en la pantalla 3.7 capturar usuario, el cual se describe más adelante. De igual forma ingresa su contraseña en el cuadro de texto correspondiente y presiona la tecla <ENTER> o selecciona el botón enviar. Si la información del correo y contraseña son correcto se permite el acceso al sistema y muestra la pantalla principal con las operaciones relevantes del mismo. En caso de que el correo o la contraseña sean incorrectos el sistema mostrará una ventana con un mensaje de error para indicar al usuario que intente nuevamente ingresar al sistema.

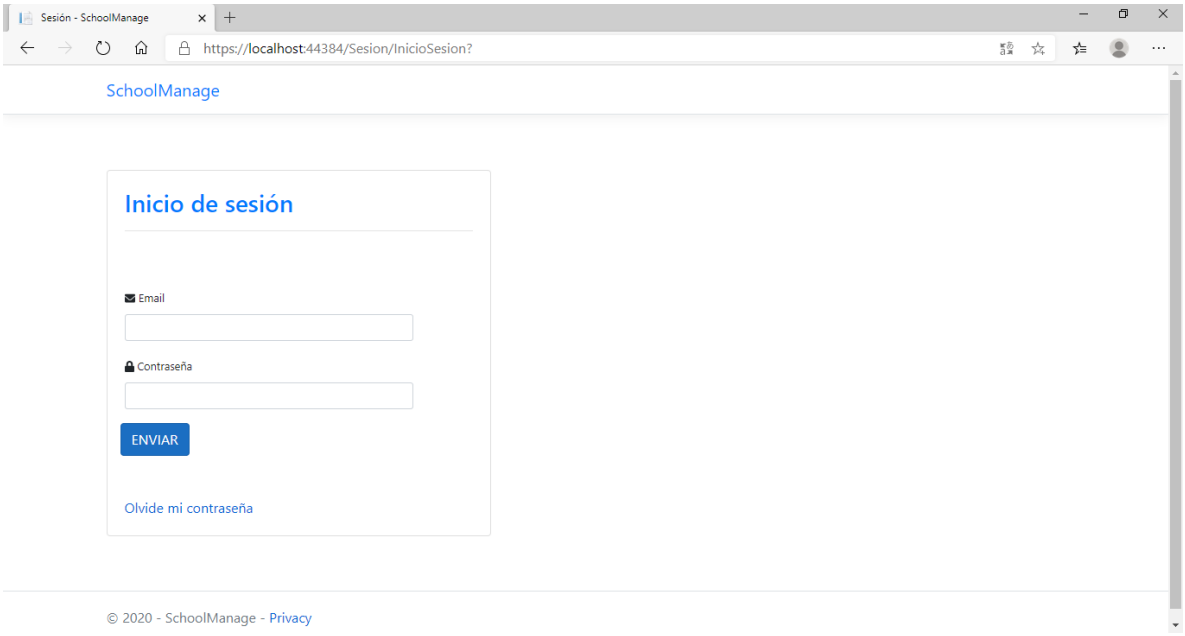


Figura 3.1 Pantalla de inicio de Sesión.
Fuente creación propia.

3.2 Recuperación de contraseña

La figura 3.2 representa la pantalla de recuperación de contraseña para el acceso al sistema, para llegar a esta pantalla debes seleccionar la opción “olvide mi contraseña” desde la pantalla de Inicio de sesión, posteriormente se debe llenar el campo email y dar clic en el botón enviar, el sistema genera un link y lo envía mediante un correo electrónico a la cuenta de email que ingreso, al seleccionar el link desde el correo electrónico recibido redireccionará a la pantalla para ingresar una nueva contraseña.

Una de las precondiciones que se debe cumplir es que el correo que ingreso debe estar previamente registrado, de lo contrario el sistema muestra el error “Email no se encuentra registrado”. Una poscondición es que una vez recuperada la contraseña el acceso al link expira.

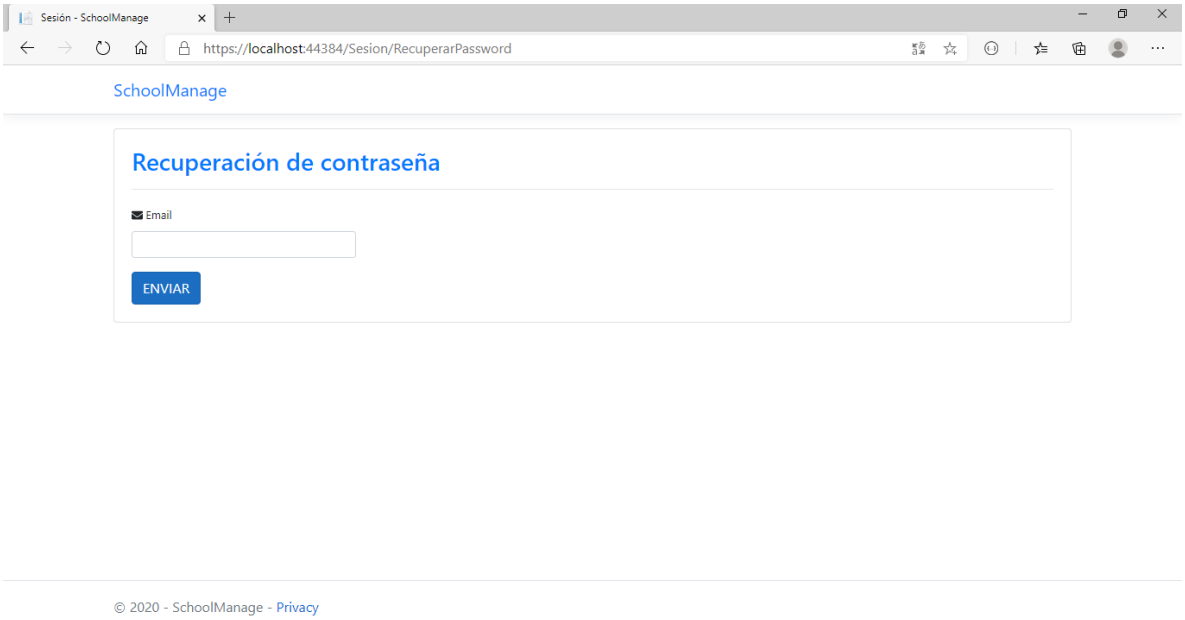


Figura 3.2 Pantalla de recuperación de contraseña.
Fuente creación propia.

3.3 Cambiar contraseña

La figura 3.3 representa la pantalla para realizar cambio de contraseña, esta opción permite poder cambiar la contraseña a través de un formulario llenando los datos solicitados, una vez llenos los campos procedes a dar un clic en el botón enviar, si la contraseña anterior es incorrecta o la nueva contraseña no coincide con la confirmación de contraseña el sistema muestra el error correspondiente, de lo contrario procede a procesar la solicitud finalizando la sesión actual para ingresar con la contraseña nueva. Una precondición para poder acceder a esta pantalla debes haber iniciado sesión.

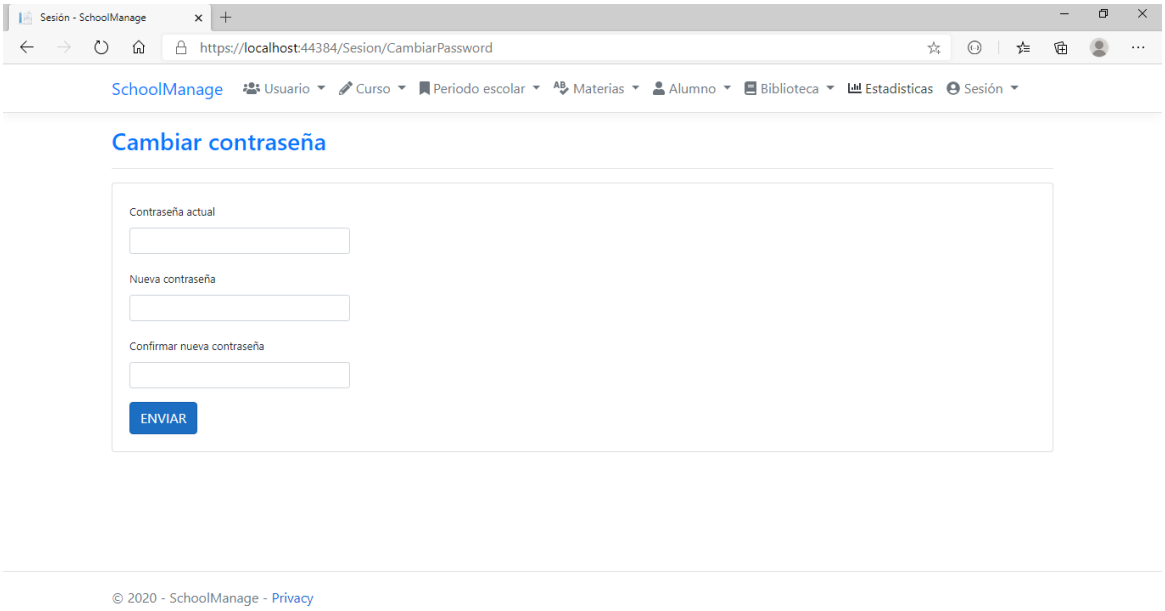


Fig. 3.3 Pantalla cambiar contraseña.
Fuente creación propia.

3.4 Cerrar sesión

La figura 3.4 representa la pantalla para cerrar sesión, al final del menú de navegación del sistema se encuentra un botón llamado sesión, al dar clic muestra un submenú con la opción cerrar sección como se muestra en la figura, procedemos a dar clic, el sistema recibe la petición y finaliza la sesión denegando las futuras solicitudes redireccionando a la página inicio de sesión.

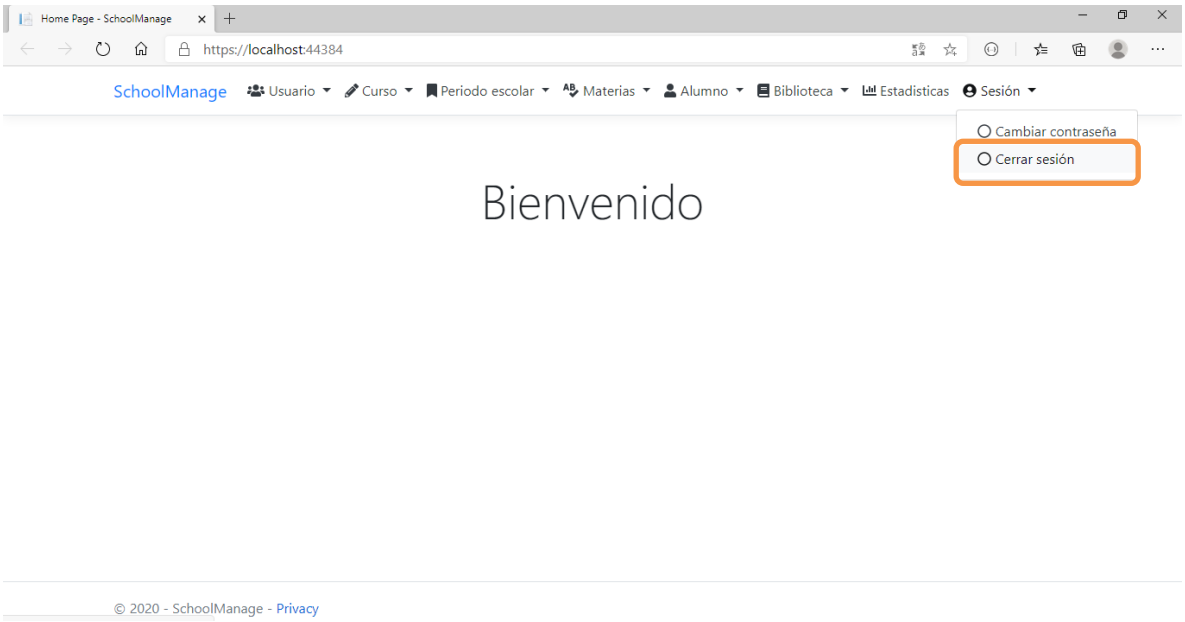


Figura 3.4 Pantalla cerrar sesión.
Fuente creación propia.

3.5 Capturar periodo escolar

La figura 3.5 representa la pantalla para capturar periodo escolar, esto se logra llenando un formulario con el nombre del periodo, fecha de inicio y fecha de fin, una vez llenos los campos solicitados das clic en el botón enviar, el sistema procesa la solicitud realizando la baja del último periodo activo en caso que exista. Una de las condiciones en el llenado del formulario es que las fechas no pueden ser iguales o la fecha inicial no debe ser mayor a la fecha final, en caso que esto ocurra el sistema lo notificara a través de un mensaje de error.

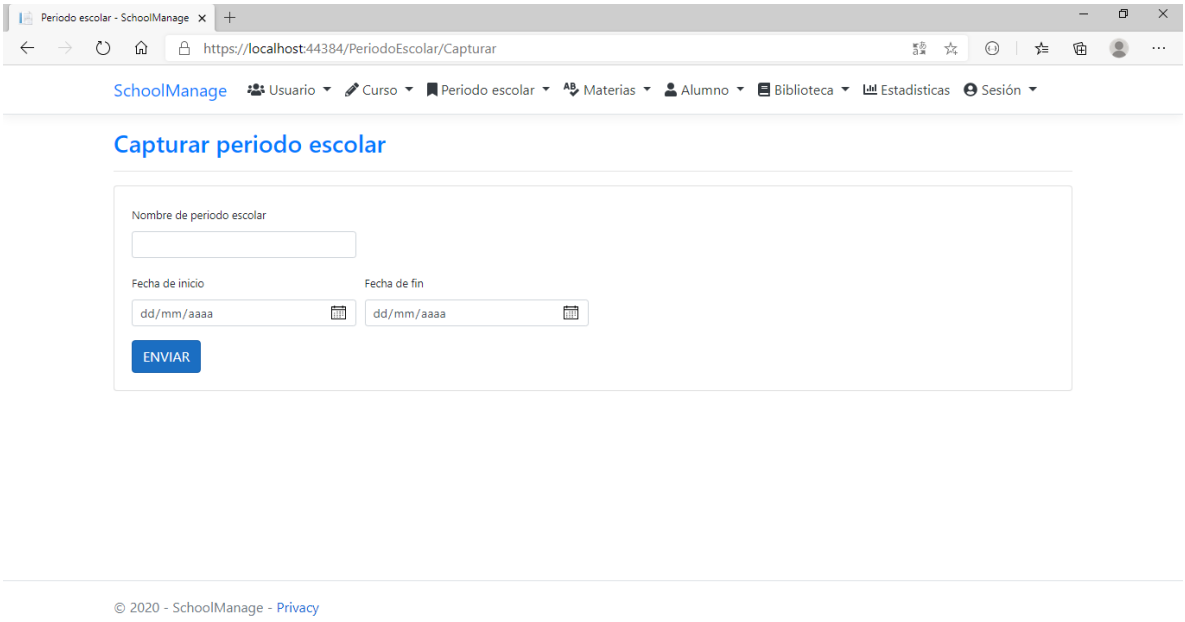


Figura 3.5 Pantalla capturar periodo escolar.
Fuente creación propia.

3.6 Capturar materias

La figura 3.6 representa la pantalla para capturar materias, esto se logra a través del llenado de un formulario con el nombre de la materia y seleccionas en una lista previamente cargada el grado al que pertenece, posteriormente se da clic en el botón enviar y el sistema procede a realizar la petición. Una precondición para que la petición se ejecute correctamente es que todos los campos son requeridos, de lo contario se notificara a través de un mensaje de error.

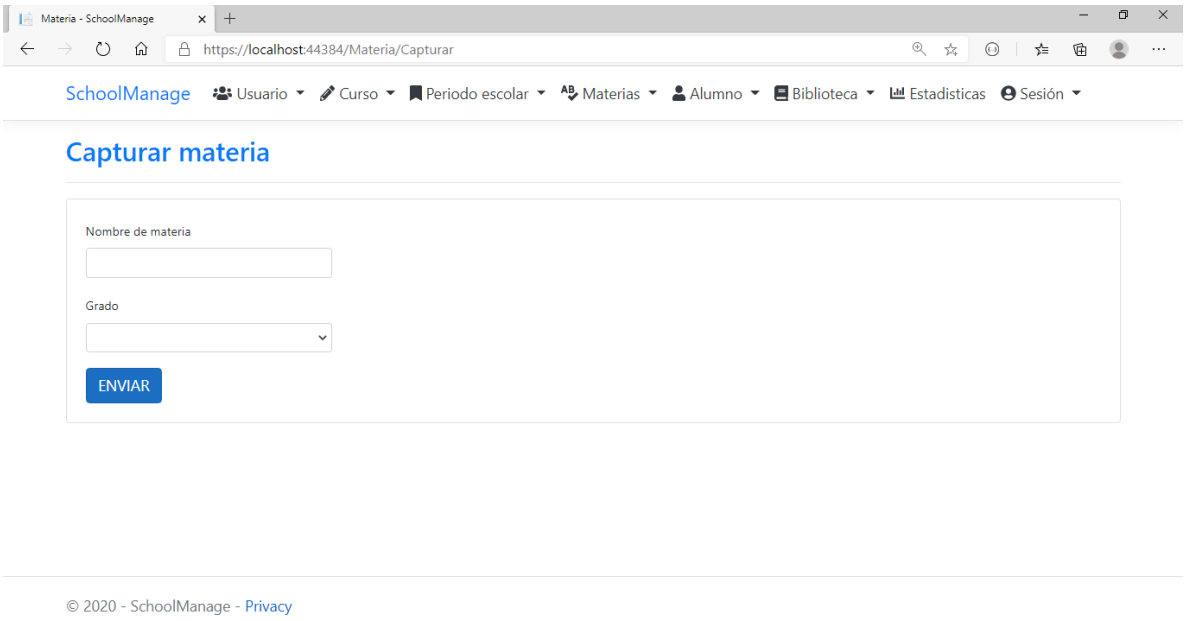


Figura 3.6 Pantalla capturar materias.
Fuente creación propia.

3.7 Capturar usuario

La figura 3.7 representa la pantalla para capturar usuario, esto se realiza a través del llenado de un formulario donde solicita sus datos generales, dirección y otros, en esta pantalla se define el rol de cada usuario a dar de alta (administrador, profesor o bibliotecario), una vez lleno el formulario das clic en enviar, el sistema procesa la solicitud y envía un correo electrónico con la confirmación de registro y las credenciales correspondientes para el acceso al sistema, se sugiere al usuario cambiar la contraseña a la brevedad para asegurar su confidencialidad. En caso de dar clic en el botón enviar y todos los campos no se encuentren llenos se notifica con la etiqueta “Requerido” en el campo faltante.

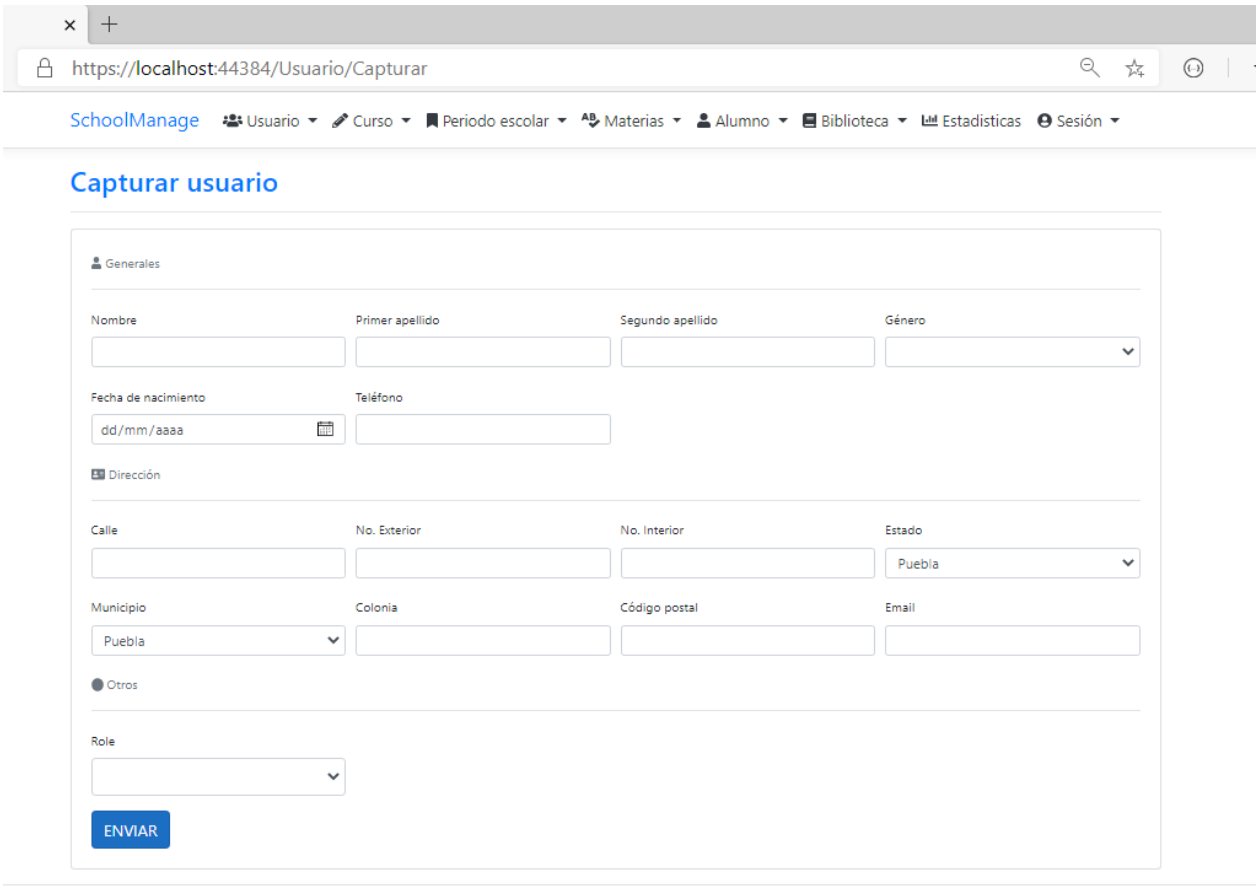


Figura 3.7 Pantalla capturar materias.
Fuente creación propia.

3.8 Capturar alumno

La figura 3.8 representa la pantalla para capturar alumno, esta captura se realiza mediante el llenado de un formulario, posteriormente das clic en el botón enviar y el sistema procede a procesar la solicitud, si alguno de los campos requeridos no son llenados, se notificara con un mensaje de campo requerido. Una precondición es que para poder asignar un tutor este deberá estar previamente registrado en el sistema como se describe más adelante.

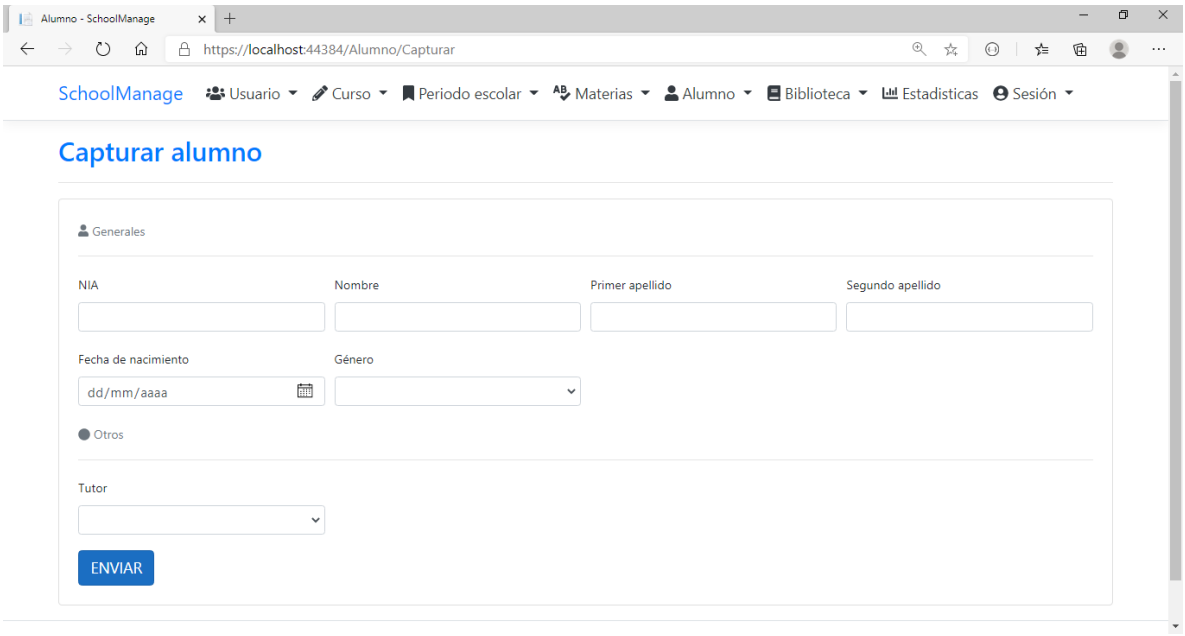


Figura 3.8 Pantalla capturar alumno.
Fuente creación propia.

3.9 Capturar tutor

La figura 3.9 representa la pantalla para capturar tutor, esto se logra a través del llenado de un formulario compuesto por datos generales, dirección y otros, una vez que el formulario este lleno se procede a dar clic en enviar. En caso de dar clic en el botón enviar y falta llenar algún campo, el sistema notificara mediante el mensaje “Requerido” en el campo o campos que falten por llenar.

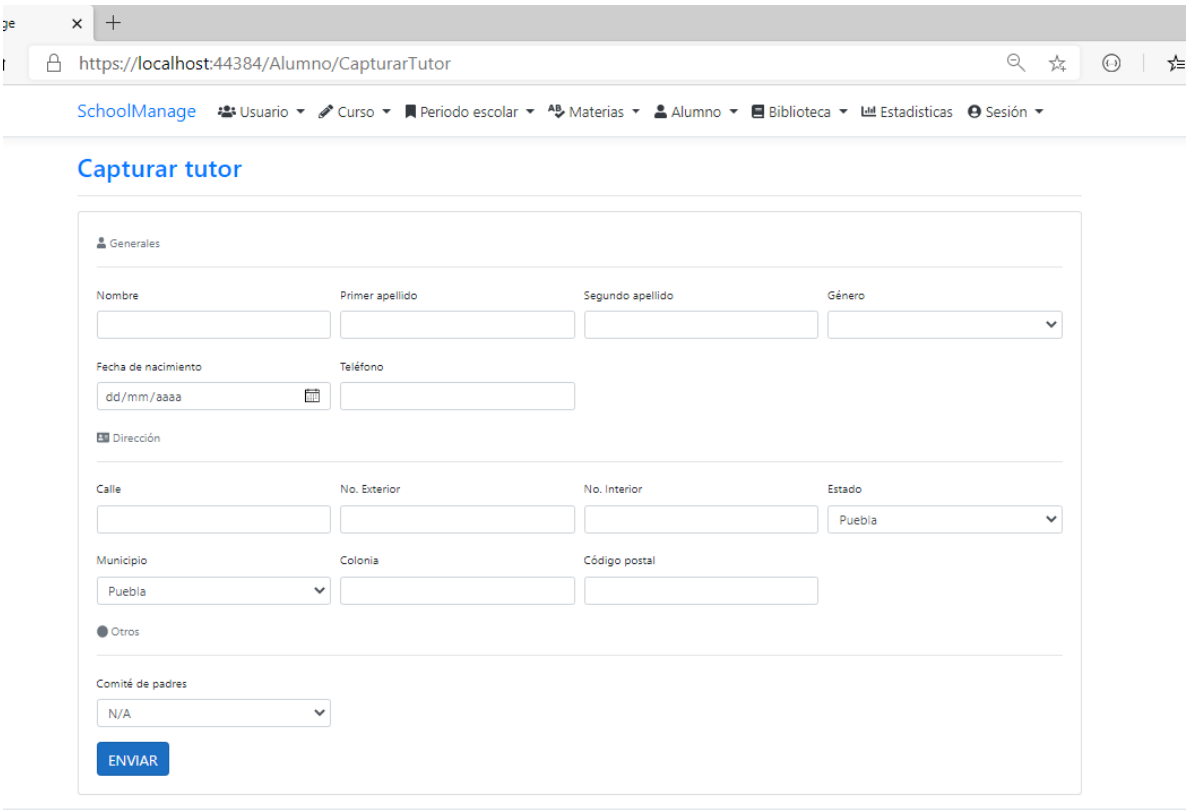


Figura 3.9 Pantalla capturar tutor.
Fuente creación propia.

3.10 Asignación de curso

La figura 3.10 representa la pantalla para asignación de curso, la asignación de un curso a un profesor es mediante el llenado de un formulario, seleccionando un profesor, periodo escolar, grado y grupo, posteriormente das clic en enviar, el sistema recibe la información, realiza la baja del curso anterior en caso de existir y le asigna el nuevo curso. Una precondición es que la información de cada una de las listas antes descritas debe estar previamente cargada.

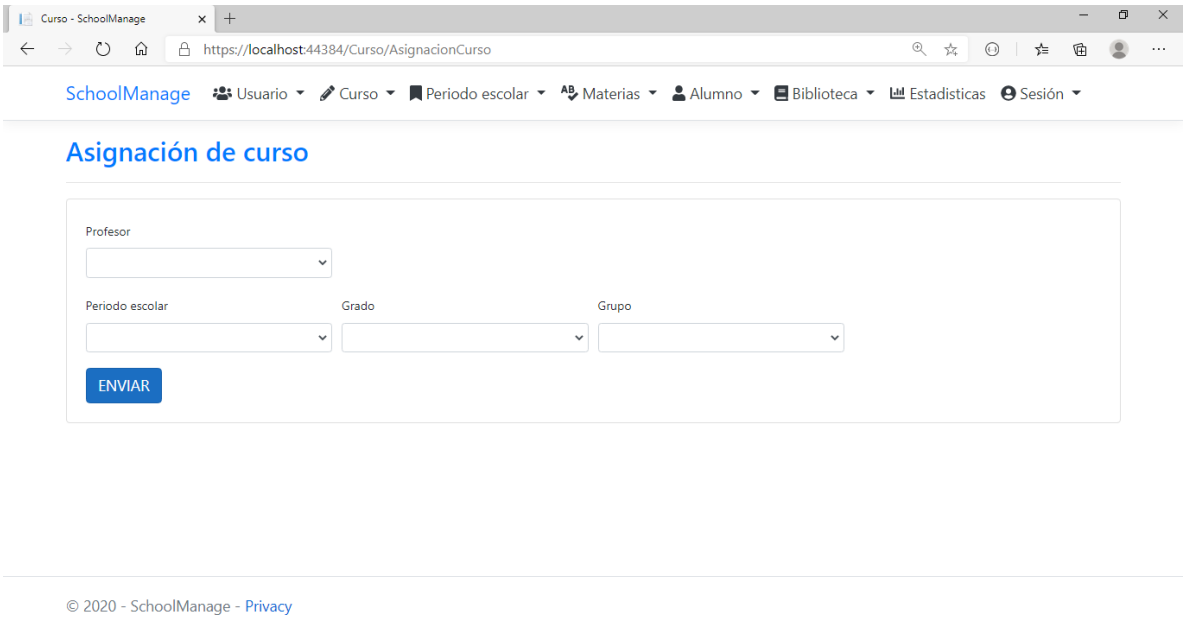


Figura 3.10 Pantalla asignación de curso.
Fuente creación propia.

3.11 Inscribir alumno a curso

La figura 3.11 representa la pantalla para inscribir alumno a curso, la inscripción se realiza seleccionado un alumno y el curso por medio de un formulario, posteriormente procedes a dar clic en el botón enviar, el sistema recibe la información, asigna el curso al alumno y realiza la baja del curso anterior en caso que exista. Las precondiciones de inscripción son que el alumno y el curso estén previamente registrados y/o asignados como se muestra en los apartados 3.8 Capturar alumno y 3.10 Asignación de curso.

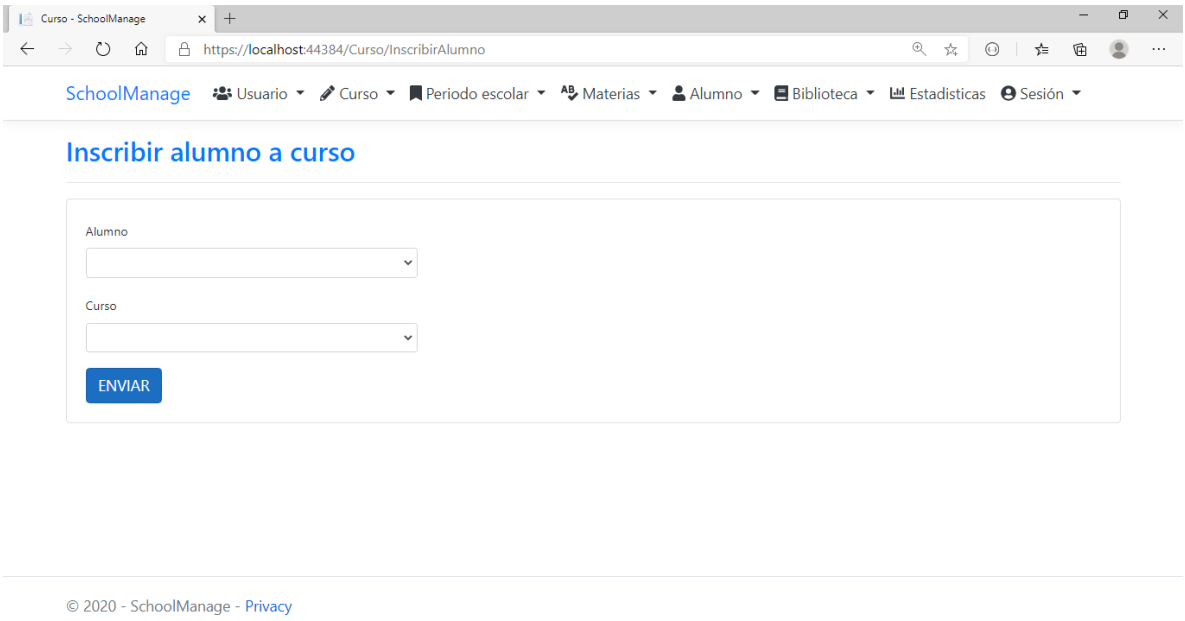


Figura 3.11 Pantalla inscribir alumno a curso.
Fuente creación propia.

3.12 Consultar estadísticas

La figura 3.12 representa la pantalla consultar estadísticas, en esta pantalla se observan cuatro graficas inscritos, aprobados/reprobados, aprovechamiento y promedio.

Estadística inscritos representa el total de alumnos inscritos en cada grado.

Estadística aprobados/reprobados representa el total de alumnos reprobados en color rojo y el total de alumnos aprobados en color azul.

Estadística aprovechamiento divide el total de alumnos en tres grupos graficando en color naranja los que obtuvieron un promedio de 6, en color amarillo los que obtuvieron un promedio entre 7 y 9 y en color verde los alumnos que obtuvieron 10.

Estadística promedio desglosa el total de alumnos por cada promedio de 5 a 10.

Las estadísticas mostradas se crean a partir de un periodo escolar seleccionado, esto permite observar el comportamiento de cada periodo escolar por separado, apoyando a tomar posibles decisiones de aprovechamiento en los próximos periodos escolares.

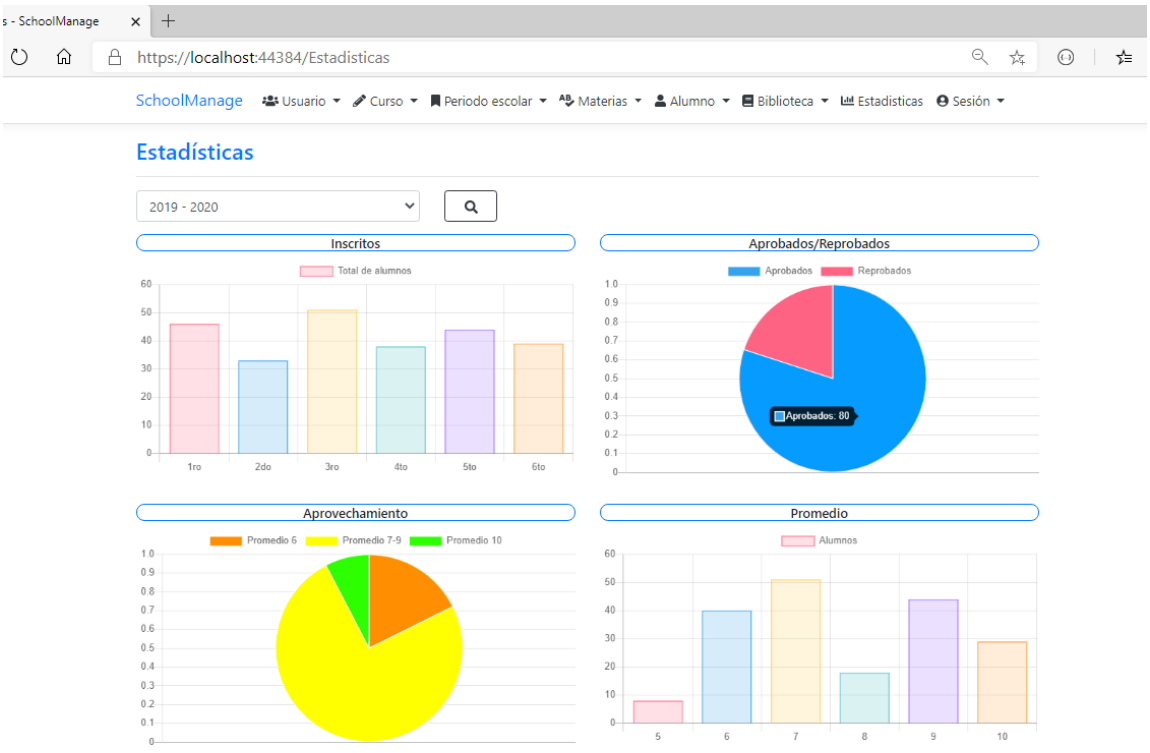


Fig. 3.12 Pantalla consultar estadísticas.
Fuente creación propia.

3.13 Capturar calificaciones

La figura 3.13 representa la pantalla capturar calificaciones. Cada periodo escolar cuenta con 5 bimestres asignados a cada curso, por lo tanto este módulo permite dar de alta la calificación de cada uno de los alumnos, para lograr esto se debe seleccionar la materia y el bimestre a calificar, posteriormente dar un clic en el botón con el icono de una lupa, el sistema recibe la petición y muestra en pantalla una tabla con dos columnas filtrando a los alumnos inscritos al curso del profesor logueado. La columna uno muestra el nombre completo del alumno, la columna dos muestra un campo abierto para ingresar la calificación correspondiente a la metería y bimestre antes seleccionado, posteriormente se debe llenar cada uno de los campos con la calificación correspondiente y das clic en el botón enviar, si el sistema procesa la solicitud de forma exitosa muestra un mensaje en pantalla con la siguiente leyenda “Las calificaciones se asignaron con éxito” en caso contrario la leyenda será “Error al asignar calificaciones. Intente de nuevo por favor”.

Una precondición es que el profesor debe contar con un curso asignado.

Una poscondición es que al calificar el bimestre de un a materia y seleccionas nuevamente esta materia, el bimestre calificado ya no debe aparecer en la lista de bimestres y si ya calificaste los 5 bimestres de una materia, esta debe desaparecer de la lista de materias mostrada.

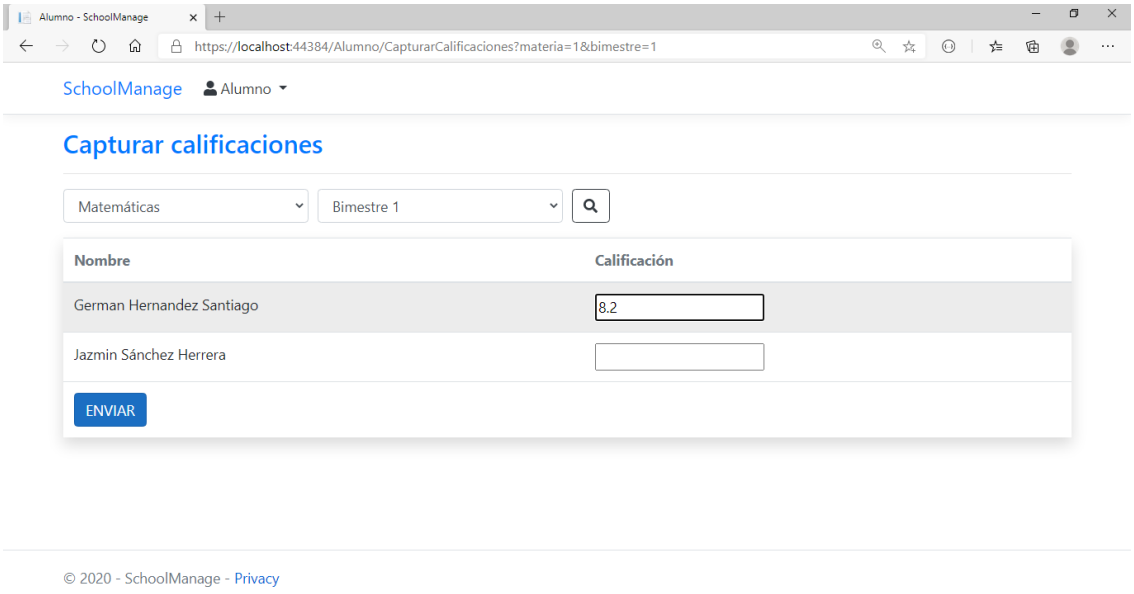


Figura 3.13. Pantalla capturar calificaciones.
Fuente creación propia.

3.14 Capturar libro

La figura 3.14 representa la pantalla capturar libro, esta pantalla permite dar de alta un libro a través de un formulario que solicita el título, autor, edición, publicación, editorial, año y descripción, una vez que los campos estén llenos das un clic en el botón enviar, el sistema recibe la información y guarda los datos. En caso que alguno de los datos no esté lleno muestra un mensaje indicando que la información es requerida.

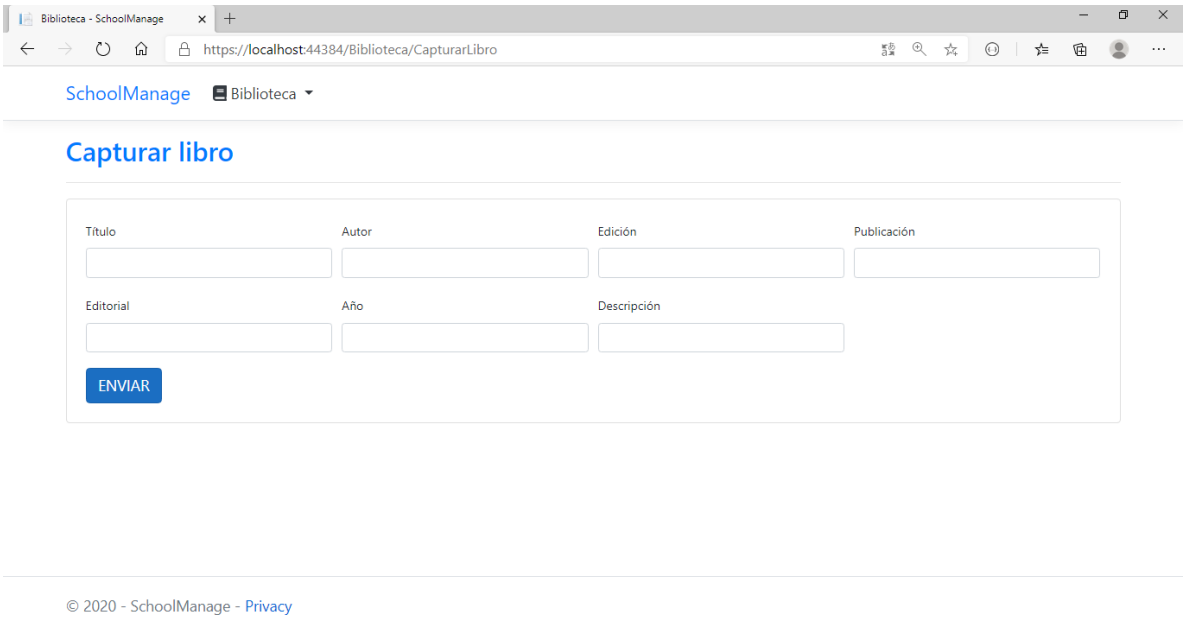


Figura 3.14 Pantalla capturar libro.
Fuente creación propia.

3.15 Registrar código de libro

La figura 3.15 representa la pantalla registrar código de libro, en esta pantalla el primer paso para registrar uno o varios códigos de un mismo libro es seleccionar el título del libro.

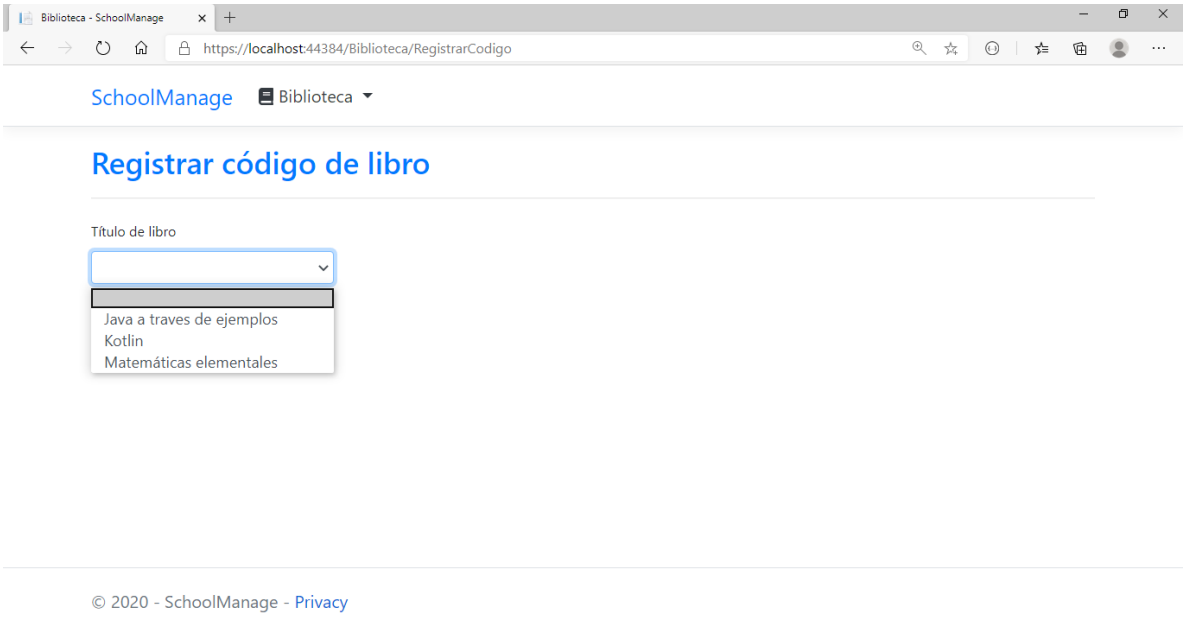


Figura 3.15 Pantalla registrar código de libro.
Fuente creación propia.

Una vez seleccionado el libro, el sistema detecta automáticamente la selección y solicita el primer código a asignar, como se muestra en la figura 3.16 Pantalla título de libro seleccionado.

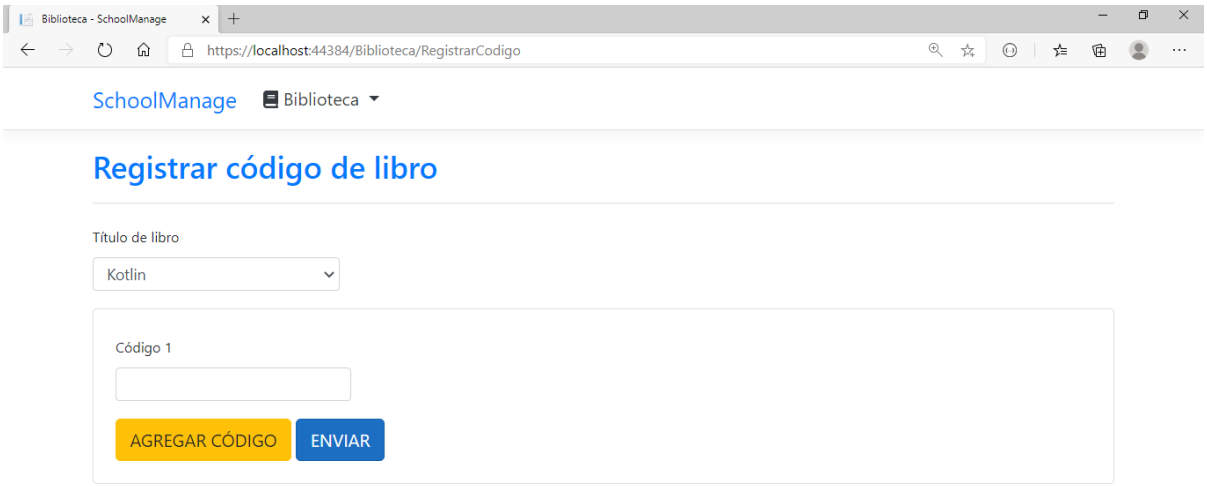


Figura 3.16 Pantalla título de libro seleccionado.
Fuente creación propia.

Si deseas agregar más códigos al mismo título del libro das clic en el botón “AGREGRA CÓDIGO”, por cada clic en este botón el sistema agrega otro campo para insertar la información como se muestra en la figura 3.17 Pantalla agregar código. Si ya no cuentas con códigos del mismo título del libro das clic en el botón enviar, el sistema recibe la petición y valida que los códigos recibidos no se encuentren antes registrados, en caso contrario muestra un mensaje indicando los códigos que no se pudieron guardar por dicho motivo.

Una precondición es que el libro se encuentre registrado para que pueda ser mostrado en la lista de selección título de libro.

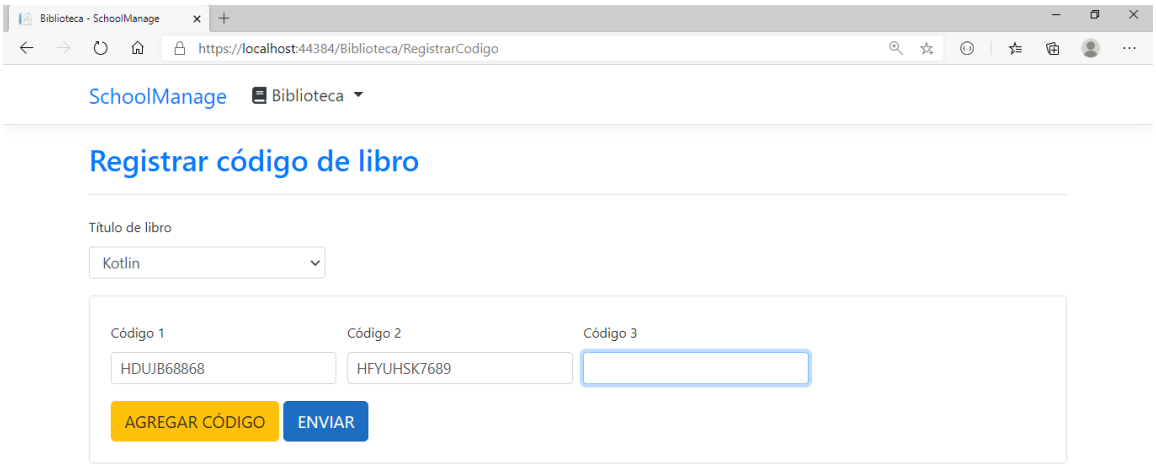


Figura 3.17 Pantalla agregar código.
Fuente creación propia.

3.16 Prestar libro

La figura 3.18 representa la pantalla prestar libro, en esta pantalla se captura el código del libro a prestar y das clic en el botón con el icono de búsqueda. El sistema recibe la petición y realiza la búsqueda del libro.

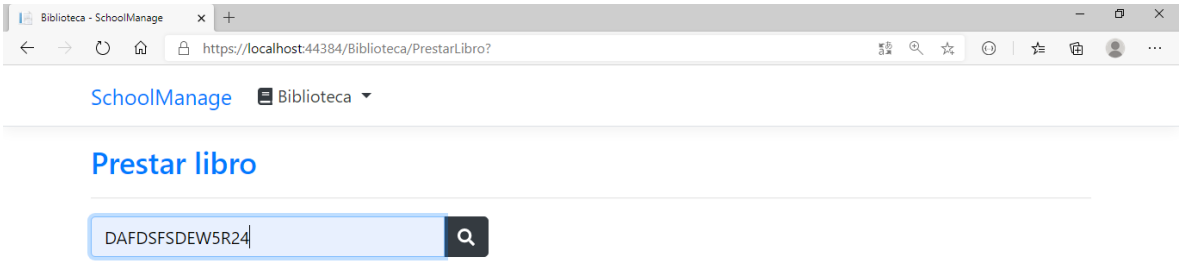


Figura 3.18 Pantalla prestar libro.
Fuente creación propia.

Si el código no existe muestra el siguiente mensaje “El código no se encuentra registrado”.

Si el código existe, muestra el nombre del libro y un campo con una lista de alumnos para seleccionar quien lo solicita, como se muestra en la figura 3.19 Pantalla seleccionar alumno, por ultimo das clic en el botón enviar, el sistema recibe la petición y cambia el estatus del libro a prestado.

Una precondición es que el código del libro y el alumno deben estar previamente registrados.

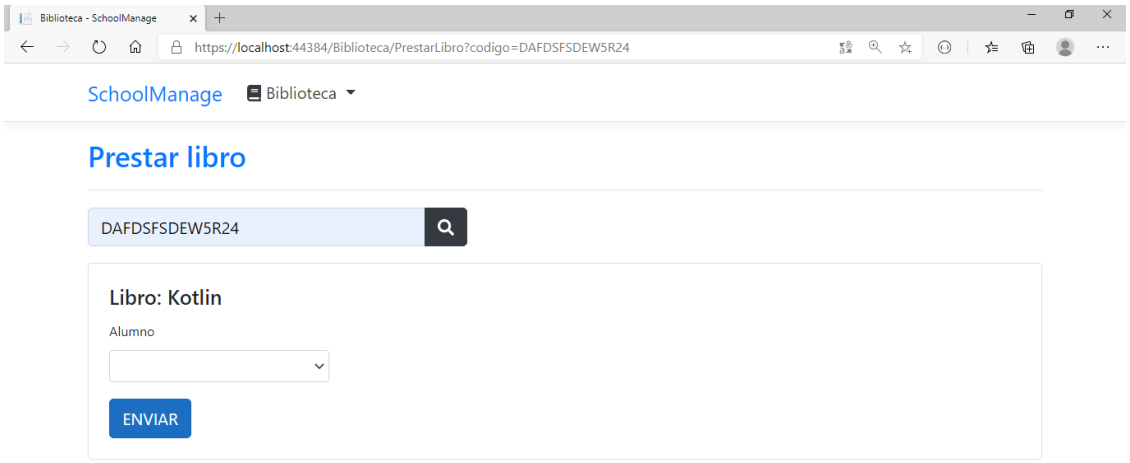


Figura 3.19 Pantalla seleccionar alumno.
Fuente creación propia.

3.17 Devolver libro

La figura 3.20 representa la pantalla devolver libro, en esta pantalla se captura el código del libro a devolver y das clic en el botón con el icono de búsqueda. El sistema recibe la petición y realiza la búsqueda.

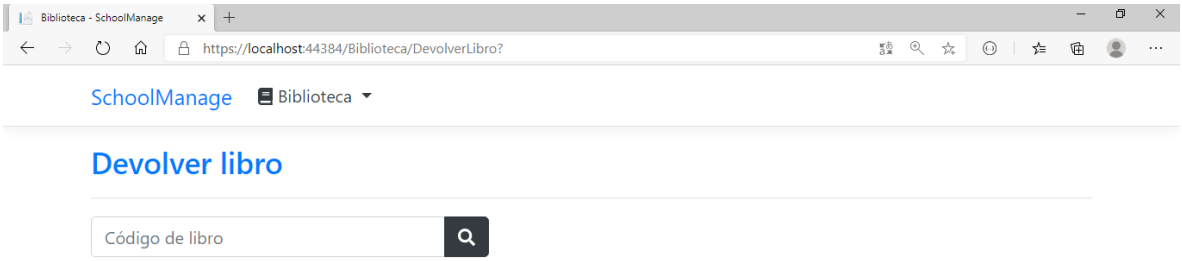


Figura 3.20 Pantalla devolver libro

Si el código no existe muestra el siguiente mensaje “El código no se encuentra registrado”.

Si el código existe muestra el nombre del libro y el nombre del alumno que lo solicito como se muestra en la figura 3.21 Pantalla con información de préstamo, por último das

clic en el botón enviar, el sistema recibe la petición y cambia el estatus del libro a disponible.

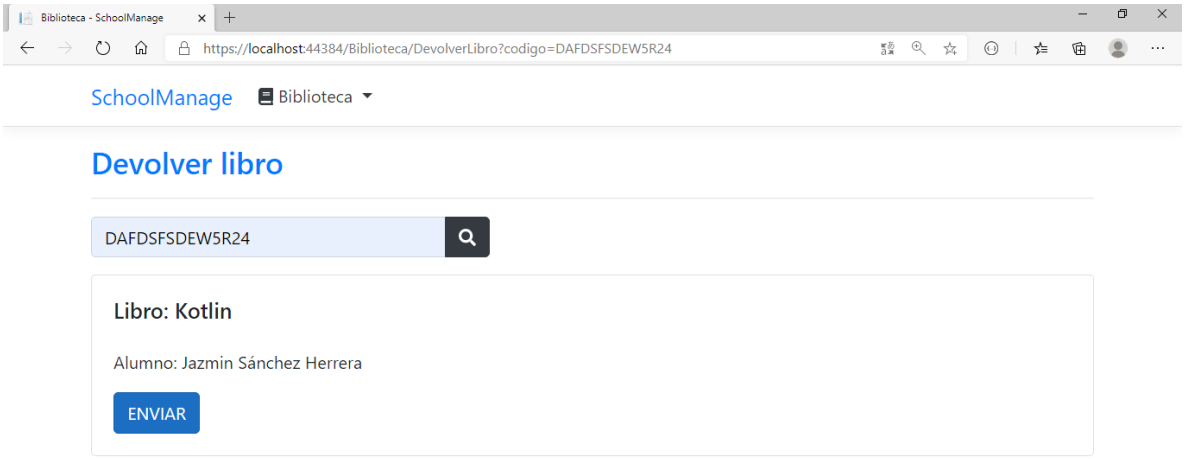


Figura 3.21 Pantalla con información de préstamo.
Fuente creación propia.

3.18 Pruebas de caja negra

Las pruebas de caja negra se definen como una técnica en la cual solo se prueba la funcionalidad de una aplicación ignorando la parte interna. Esto quiere decir que se obvia la estructura del código, la arquitectura, los detalles relacionados con la implementación de los diferentes módulos, paquetes o rutas en la que se compone el código. Este tipo de prueba se basa por completo en los requisitos de la aplicación y en sus especificaciones técnicas.

Las pruebas de caja negra solo se centran en las entradas y salidas de la aplicación, sin preocuparse por el contenido interno. Lo que pase por dentro es indiferente, solo importa que si se realiza cierta acción, la salida sea la indicada según los requerimientos.

Este tipo de prueba se aplica a cualquier tecnología que se desee. Por ejemplo, se puede realizar pruebas de caja negra en Software de escritorio, en aplicaciones web o incluso en aplicaciones móviles. Cualquier software se podría probar simplemente enfocándose en las entradas y salidas sin conocer las implementaciones del código interno. Por tal motivo, esta prueba se convierte en una de las más realizadas a nivel profesional.

Para preparar los casos de pruebas debe existir un análisis previo de los requisitos y especificaciones del software.

Se debe contar con las entradas de datos válidos, también llamado escenario de prueba positiva, para verificar si el software las procesa correctamente. También se debe contar con las entradas de datos no válidas (llamado escenario de prueba negativa) para comprobar si el software que se está probando es capaz de detectarlas y reaccionar ante estas entradas.

Basándose en las entradas, se debe determinar para cada una de estas las salidas esperadas correspondientes.

Una vez que se tienen las entradas y su correspondiente salida, se ejecutan los casos de prueba, comprobando la salida que ha emitido el software con la salida esperada de los casos de prueba.

Si la salida del software coincide con la salida esperada, el software hace lo que tiene que hacer para esa entrada. Pero si la salida del software no coincide con la salida esperada, hemos encontrado un defecto en el software. Lo que conllevará su posterior reparación [17].

3.19 Pruebas de caja blanca

La prueba de caja blanca se basa en el diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivarlos. Mediante la prueba de la caja blanca el ingeniero del software puede obtener casos de prueba que:

- Garanticen que se ejerciten por lo menos una vez todos los caminos independientes de cada módulo, programa o método.
- Ejerciten todas las decisiones lógicas en las vertientes verdadera y falsa.
- Ejecuten todos los bucles en sus límites operacionales.
- Ejerciten las estructuras internas de datos para asegurar su validez.

Es por ello que se considera a la prueba de Caja Blanca como uno de los tipos de pruebas más importantes que se le aplican a los software, logrando como resultado que disminuya en un gran porciento el número de errores existentes en los sistemas y por ende una mayor calidad y confiabilidad.

La prueba de caja blanca es sobre el código fuente de la aplicación, y en consecuencia, sobre los diferentes algoritmos y estructuras de datos utilizados.

Básicamente, el que prueba selecciona distintos valores de entrada para examinar cada uno de los posibles flujos de ejecución del programa y cerciorarse de que se devuelven los valores de salida adecuados. Al estar basadas en una implementación concreta, si ésta se modifica, por regla general las pruebas también deberán rediseñarse.

Aunque las pruebas de caja blanca son aplicables a varios niveles (unidad, integración y sistema), habitualmente se aplican a las unidades de software. Su cometido es comprobar los flujos de ejecución dentro de cada unidad (función, clase, módulo, etc.) pero también pueden probar los flujos entre unidades durante la integración, e incluso entre subsistemas, durante las pruebas de sistema.

Para realizar este tipo de pruebas, tendremos que tener habilidades en programación y conocimiento funcional para poder entender el objeto de determinadas clases y métodos [17].

3.20 Resultados de aplicar las pruebas

Las pruebas de caja negra y caja blanca aplicadas en el proyecto Aplicación Web para la administración de recursos materiales y académicos de una escuela primaria mediante una metodología de desarrollo de software basada en el Proceso Unificado fueron de gran ayuda para detectar errores en la lógica de negocio y en su comportamiento a eventos inesperados con el funcionamiento del sistema.

En las pruebas de caja negra y caja blanca se prepararon datos de entrada y salida, para cada uno de los casos de uso existentes.

En cada ejecución de pruebas de caja negra se comparó el dato de salida con el resultado del sistema esperado. También se ejecutó cada caso de uso con datos que pudieran generar un mal funcionamiento del sistema, esperando un resultado de salida erróneo o peor aún que el sistema se cayera, y efectivamente se encontraron vulnerabilidades en algunos casos de uso debido a la no recuperación a fallos, que fueron generados por no contemplar algunas excepciones. Se aplicaron las correcciones necesarias y se volvieron a ejecutar las pruebas obteniendo un buen resultado en estas.

En las pruebas de caja blanca se ejecutó paso a paso cada caso de uso, siguiendo y analizando la ejecución de cada clase, esto facilito la detección y corrección de errores a nivel de lógica de negocio, datos o excepciones no controladas.

Las pruebas de software jugaron un papel importante en el desarrollo de este proyecto, ya que si estas pruebas no existieran o peor aún, no se hubieran realizado, sería difícil poder detectar errores antes de llevar el sistema a un ambiente de producción. Gracias a esto se logró una gran reducción de errores.

Conclusiones

El proyecto descrito proporcionó la visión general de la problemática que se presenta en la construcción de sistemas de información, analizando el problema a resolver detectando las necesidades más importantes y demandantes. Este análisis permitió proponer una solución a través de una aplicación web para la administración de recursos materiales y académicos de una escuela primaria.

Al utilizar la metodología de proceso unificado de desarrollo de software provee las características de desarrollo de calidad, los diagramas y especificaciones de casos de uso contribuyeron a entender el problema para la construcción del mismo, creando un software escalable y de fácil mantenimiento.

Una de las posibles mejoras a futuro es añadir nuevas gráficas para el análisis de datos que ayuden a la toma de decisiones.

Se realizó el análisis y diseño de base de datos y reforzó los conocimientos ya obtenidos, sumando experiencia en el tema y evitando la redundancia de datos.

La codificación de este proyecto me ayudo a poner en práctica las convenciones de código basados en la guía de programación de C# como son nomenclatura, diseño, comentarios y lenguaje. Entity Framework Core permitió el acceso a datos mediante un modelo compuesto por clases de entidad y un objeto de contexto para la conexión con la base de datos.

Se lograron los objetivos de crear una aplicación web para la administración de recursos materiales y académicos de una escuela primaria, cumpliendo con lo propuesto en el diagrama de casos de uso de manera exitosa, obteniendo el resultado previsto en la construcción de la aplicación web, cumpliendo con la solicitud del usuario para obtener un mayor control que ayuda a facilitar la administración de recursos materiales y académicos.

Bibliografía

1. https://www.ecured.cu (2020). Metodologías de desarrollo de software https://www.ecured.cu/Metodologias_de_desarrollo_de_Software [Accedido 15 Marzo. 2020]
2. https://irfeyal.wordpress.com (2020). Diseño de base de datos https://irfeyal.wordpress.com/bases-de-datos/modelamiento-de-bdd/ [Accedido 15 Marzo. 2020]
3. http://ccdoc-basesdedatos.blogspot.com (2020). Modelo entidad-relación ER http://ccdoc-basesdedatos.blogspot.com/2013/02/modelo-entidad-relacion-er.html [Accedido 15 Marzo. 2020]
4. http://deletesql.com (2020). Clave foránea http://deletesql.com/viewtopic.php?f=5&t=20 [Accedido 15 Marzo. 2020]
5. https://es.wikipedia.org (2020). Normalización de bases de datos https://es.wikipedia.org/wiki/Normalizaci%C3%B3n_de_bases_de_datos [Accedido 15 Marzo. 2020]
6. https://programas.cuaed.unam.mx (2020). Normalización https://programas.cuaed.unam.mx/repositorio/moodle/pluginfile.php/872/mod_resource/content/1/contenido/index.html [Accedido 15 Marzo. 2020]
7. https://definicion.de (2020). Windows https://definicion.de/windows/ [Accedido 15 Marzo. 2020]
8. https://blog.infranetworking.com (2020).IIS https://blog.infranetworking.com/servidor-iis/ [Accedido 15 Marzo. 2020]
9. https://es.wikipedia.org (2020). SQLServer https://es.wikipedia.org/wiki/Microsoft_SQL_Server [Accedido 24 Marzo. 2020]
10. docs.microsoft.com (2020). Información general sobre .NET Core. [online] Disponible en: https://docs.microsoft.com/es-es/dotnet/core/about [Accedido 03 Abril. 2020]
11. James Rumbaugh, Ivar Jacobson, Grady Booch (2020). El Proceso Unificado de Desarrollo de Software.
12. Rosa Imelda García Chi (2017). Desarrollo de software con Ingeniería Web: Guía práctica y técnicas
13. Cristian Calvo Lores (2016). Diseño de Software con Modelado UML: 2ª Edición.
14. Mircea Teodor Oprea. Full-stack ASP.NET Core: Build full-stack web applications with ASP.NET Core 3.1
15. Brian L Gorman (2020) Practical Entity Framework: Database Access for Enterprise Applications
16. https://docs.microsoft.com (2020). Introducción a .NET Core. [Online] Disponible en: https://docs.microsoft.com/es-es/dotnet/core/introduction. [Accedido 26 Julio. 2020]
17. Roger S. Pressman (2006). Ingeniería del software: un enfoque práctico.