

## **Proyecto Final - Sistema de Gestión de Matricula y Mensualidad del Estudiante**

U N I V E R S I D A D  
C I E N T Í F I C A  
D E L S U R



**Fidel Lace**

**José Pardo**

**Luigui Salazar**

Curso de Análisis y Diseño OO (IN 300)

Profesor: Eric Coronel Castillo

Universidad Científica del Sur

2017

## Contenido

1. Introduccion .....	3
2. Presentación de la Empresa .....	3
3. Desarrollo del Proyecto .....	5
4. Conclusiones y Recomendaciones .....	28
5. Referencias Bibliograficas .....	29

## **1. Introducción**

Actualmente los colegios hacen uso de herramientas de aplicación general u hojas de cálculo con el fin de informatizar los datos y así obtener mejoras en la gestión administrativa. Sin embargo, esto no logra reducir o eliminar los problemas tales como tareas repetitivas y datos no integrados que conllevan a errores y re-procesos. Por consiguiente existen aún pérdida de tiempo e ineficiencia en el trabajo.

Es necesario entonces que los colegios tomen conciencia y que opten por un aplicativo informático que verdaderamente brinde soluciones a estos problemas. En el mercado actual existen sistemas administradores para colegios, pero se aprecia que en varios de estos su uso solo se da en un solo computador, que no existen restricciones en cuanto al acceso, o que las funcionalidades no satisfacen las necesidades requeridas.

## **2. Presentación de la Empresa**

El Colegio El alma del saber es un centro educativo particular fundado por la Institución Teresiana en 1950. Se encuentra ubicado en la ciudad de Lima. En estos años su opción educativa se fundamente en los valores del evangelio y se adhiere a las orientaciones de la Iglesia universal local y el sistema educativo vigente. El Colegio tiene un estilo pedagógico propio inspirado en la propuesta educativa de San Pedro Poveda, sacerdote mártir fundador de la Institución Teresiana cuyos principios básicos son:

1. La tarea educativa como un proceso de humanización.
2. La educación es entendida como medio para el cambio social que exige una preparación académica y formación en valores.

3. Vivir una opción de fe personal y comunitaria desde una espiritualidad Cristiana, laical.
4. La formación y capacitación permanente del personal docente como agentes de cambio.

### **Misión**

Busca formar personas autónomas, críticas, creativas, reflexivas, abiertas al diálogo y a lo diferente, comprometidas y solidarias. Estilo inspirado en la propuesta pedagógica de San Pedro Poveda, basada en los valores del evangelio.

Una metodología activa y participativa que dinamiza procesos educativos personalizados de cambio y la construcción colectiva de saberes y propuestas que aportan al cambio social.

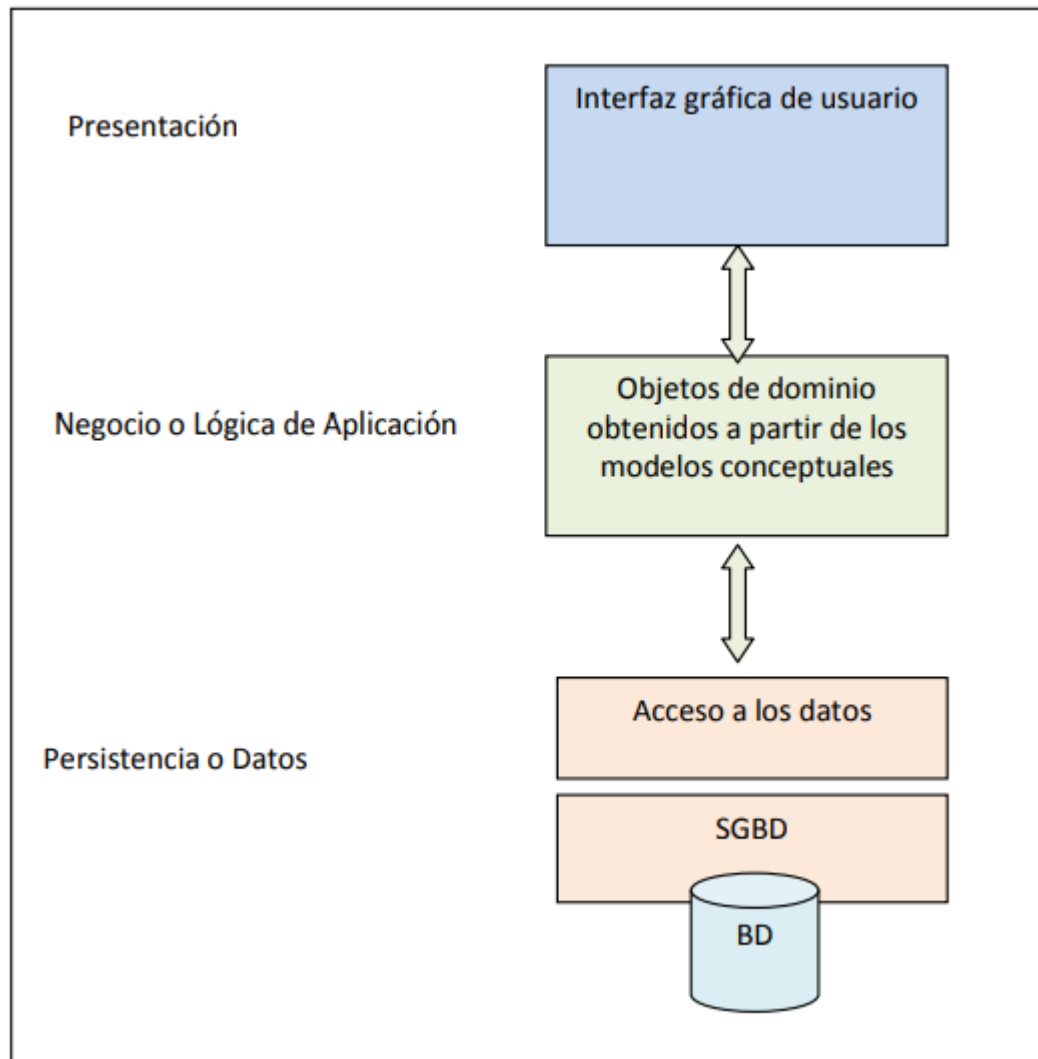
### **Visión**

Su finalidad es la promoción humana y la transformación social, desde la formación de ciudadanos ético democráticos. Educación como un proceso de “humanización” que coloca a la persona en el centro de su pedagogía y organización. Los miembros de la comunidad educativa identificados como “colaboradores”, dan testimonio de su compromiso con la Institución.

### 3. Desarrollo del Proyecto

Análisis, diseño, desarrollo e implementación de un Sistema de Información para la administración de matrícula y mensualidades del estudiante.

#### a. Capa de Diseño:



**b. Alcances:**

- ♣ Educación Básica Regular.
- ♣ Niveles de Primaria y Secundaria.
- ♣ Periodos de evaluaciones bimestrales.
- ♣ Turnos de mañana y tarde.
- ♣ Días de clase de lunes a viernes o lunes a sábado.
- ♣ Recaudación de pagos para las obligaciones de los alumnos a través de una o más entidades bancarias.

**c. Criterios técnicos:**

- ♣ Implementación en arquitectura 3 capas.
- ♣ Implementación en plataforma de escritorio.
- ♣ Empleo de tecnología Java como lenguaje de desarrollo.
- ♣ Utilización de MySQL como motor de base de datos.
- ♣ Utilización de una metodología orientada a objetos.
- ♣ Utilización de la notación UML.

**d. Requerimientos del negocio*****i. Control de cuota de ingreso:***

Actualmente el control de cuota de ingreso se realiza en archivo plano, lo que permitirá el sistema a implementar será llevar un mejor control de las cuotas de ingreso de cada alumno en una base de datos sin tener datos repetitivos y por ende confusión al momento de realizar la matrícula.

**ii. Control de matrícula:**

Actualmente el proceso de matrículas del centro de informática es manual los alumnos que llegan a matricularse en sus respectivos cursos tienen que hacer una larga cola, posteriormente al llegar con la secretaria el proceso de inscripción se realiza en Excel y demora por cada alumno.

**iii. Control de mensualidades**

Se hace un control día a día de las obligaciones de pago pendientes de los alumnos usando la información que se encuentra en los “cuadernos de cobros” y que se actualizan con: *Los cobros realizados en el mismo colegio y la información en los documentos bancarios que demuestran los cobros de las obligaciones de los alumnos.*




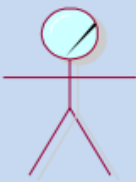
A su vez, los padres de familia tienen por su parte los bouchers de pagos bancarios y el documento “libreta de pensiones” (que brinda la institución educativa a cada alumno y que también se actualiza durante todo este proceso de control) que acreditan el estado de sus obligaciones.

***Responsables: Encargado de administración y finanzas.***

**e. Modelo del casos de uso del sistema**

**i. Actores del sistema**

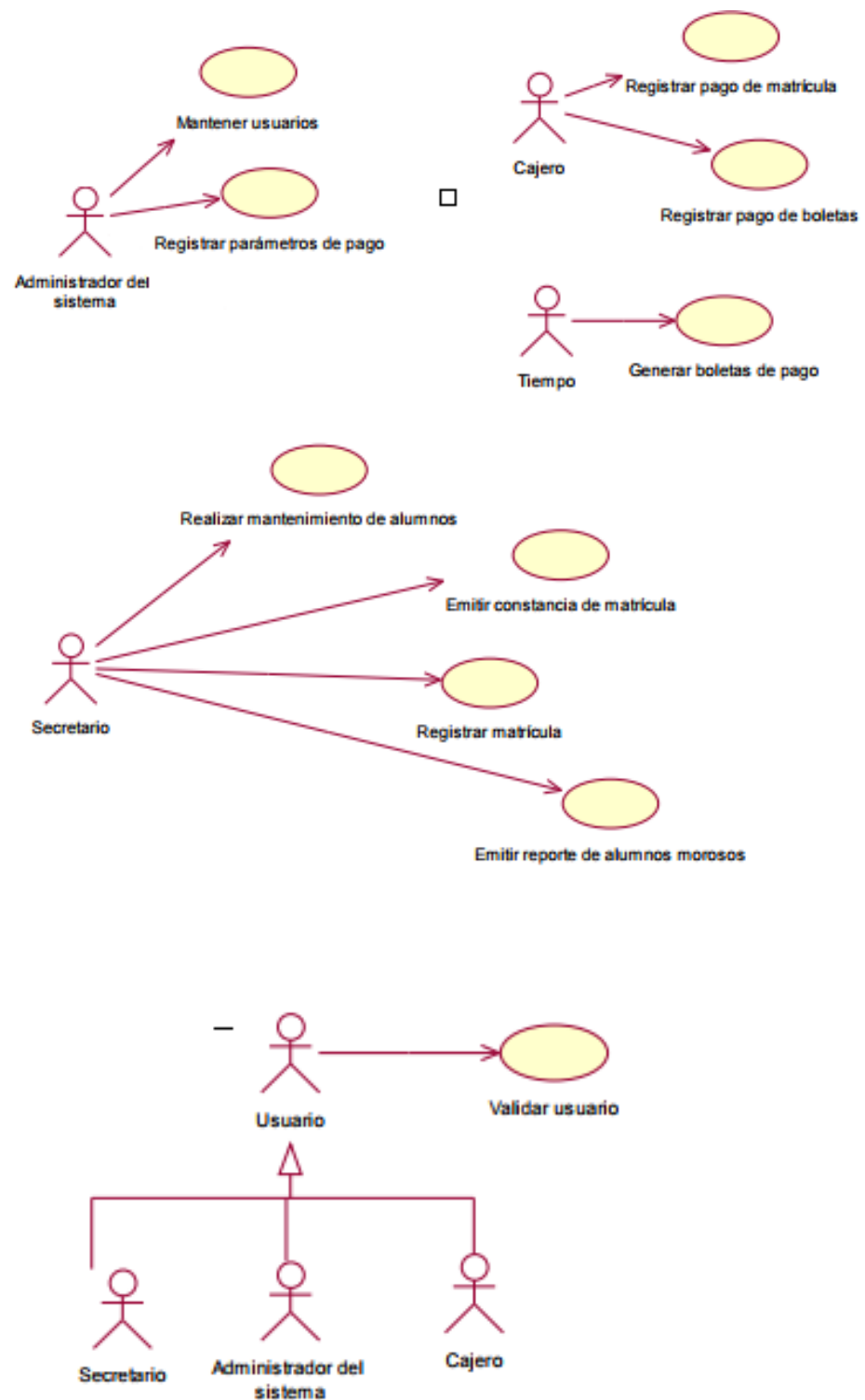
Los actores representan a los usuarios que interactúan con el Software de Gestión de matrícula y mensualidad del estudiante.

Actor	Descripción
 Secretaria	Encargada del ingreso y control de las matriculas, también verifica los pagos realizados por los alumnos.
 Administrador	Encargado de administrar el sistema.
 Operario	Permite darle mantenimiento al sistema.
 Alumno	El alumno debe llevar de forma obligatoria cursos de computación en el centro de informática.



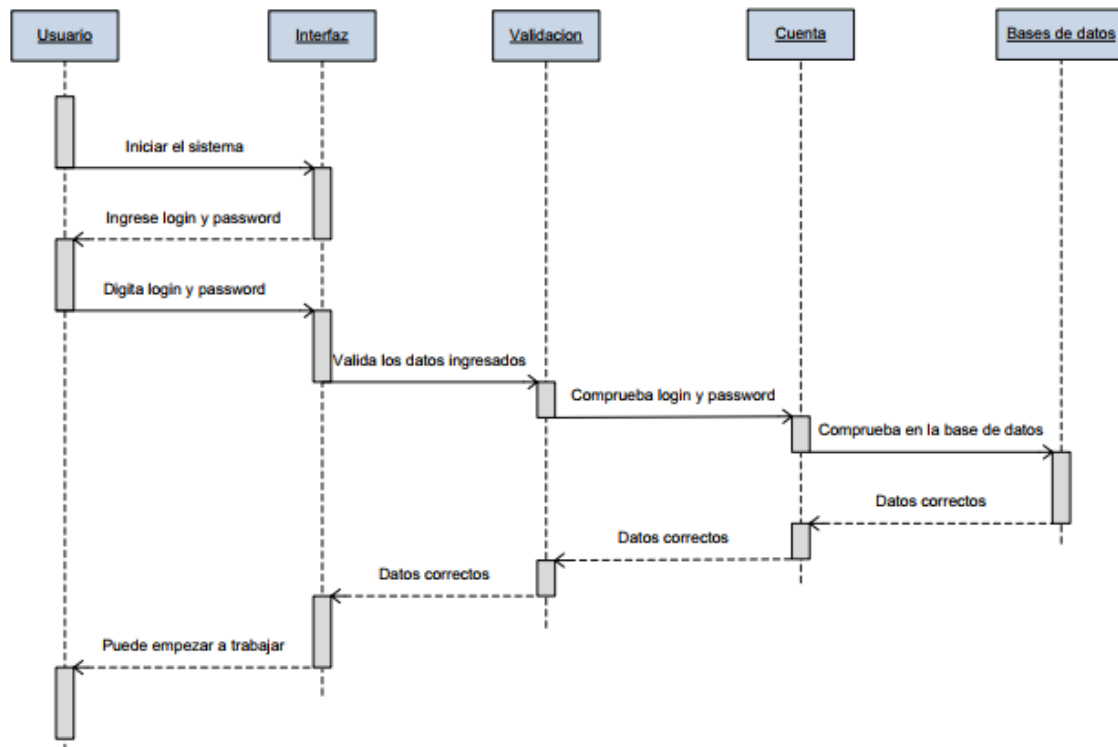
## ii. Casos de uso del sistema

### Caso de uso principal:



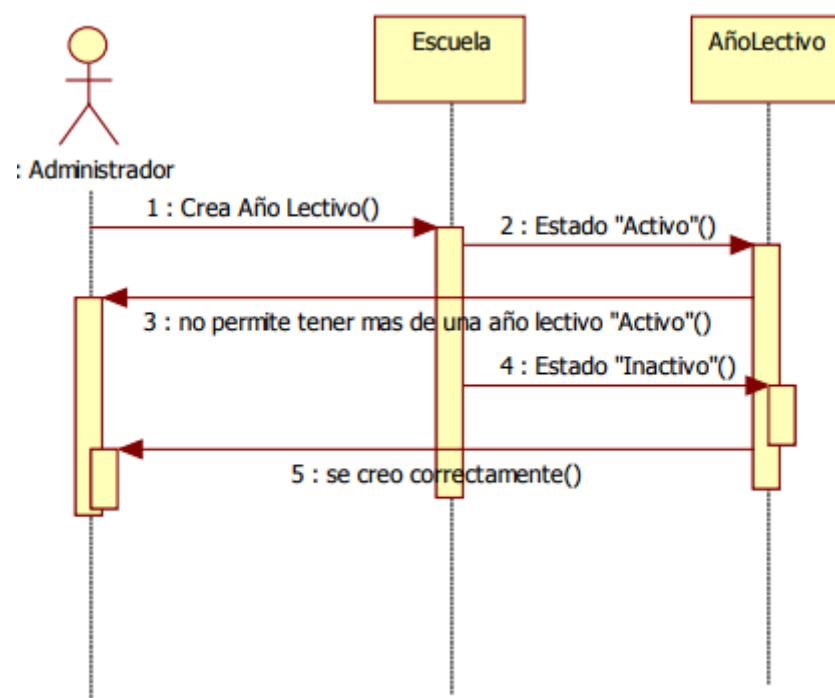
**Detalles de casos de uso:**

**- C01: Ingreso al sistema:**



- **C02: Programación del año escolar**

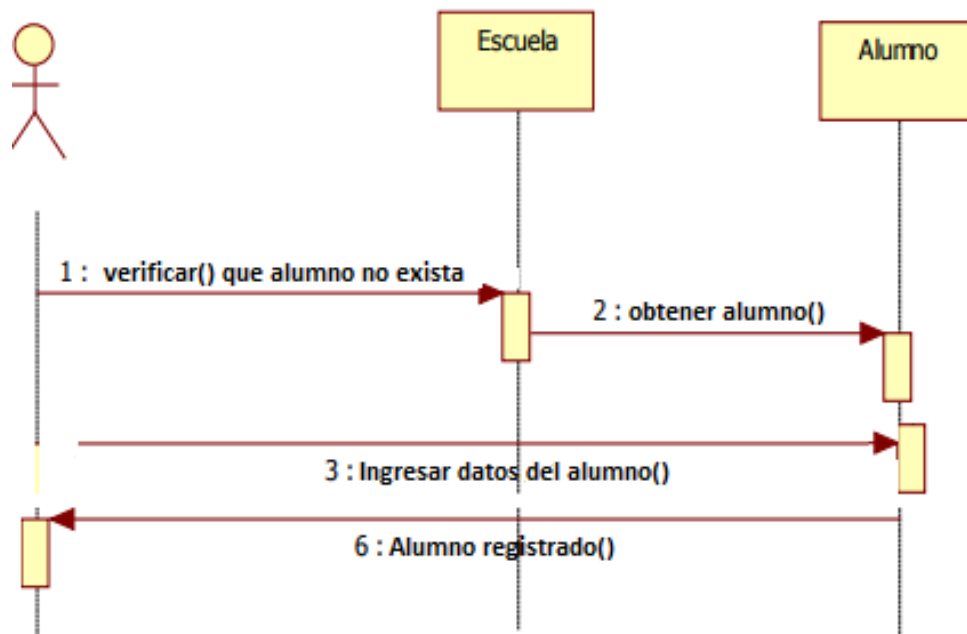
En este escenario el administrador podrá crear el año lectivo, previamente deberá poner en estado “Inactivo” al año lectivo actual o en el que se encuentra en proceso. Pero no puede tener más de un año lectivo en estado “Activo”.



- **C03: Registrar nuevo estudiante**

El propósito de este caso de uso es el de registrar en el sistema a un nuevo estudiante para la programación del año escolar.

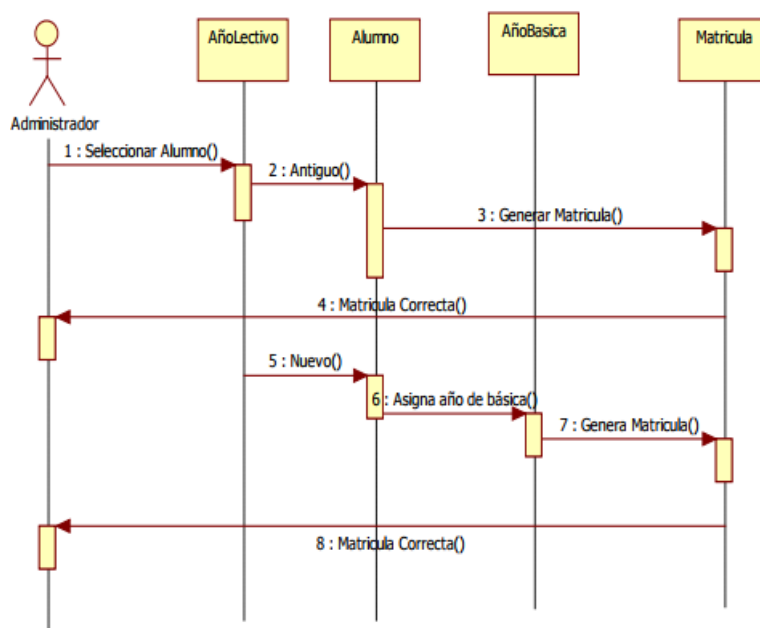
C03	AÑADIR AL SISTEMA UN ALUMNO
<b>Versión:</b>	1.0
<b>Actores:</b>	Alumno y Administrador
<b>Descripción:</b>	Creación de nuevos alumnos
<b>Pre-condición:</b>	El alumno no deberá existir en la base de datos
<b>Post-condición:</b>	Se crea un nuevo alumno en la base de datos
ACTOR	SISTEMA
1. Solicita pantalla de agregar alumno	2. Muestra pantalla solicitando el código de alumno
3. Ingresa el código de alumno	4. Valida el código
	5. Muestra pantalla para la captura de los datos del alumno
6. Ingresa los datos del alumno	7. Valida los datos ingresados
8. Confirma agregar alumno	
CAMINOS ALTERNATIVOS	
	5. Si el alumno existe muestra pantalla de error
<b>Frecuencia esperada:</b>	Debe usarse con regularidad
<b>Importancia:</b>	Es muy importante



### C04: Registrar matricula

El propósito de este caso de uso es matricular a un alumno y generar la boleta. Si es alumno antiguo se procederá a la ratificación de su matricula en caso contrario se ingresara la información del alumno nuevo al sistema.

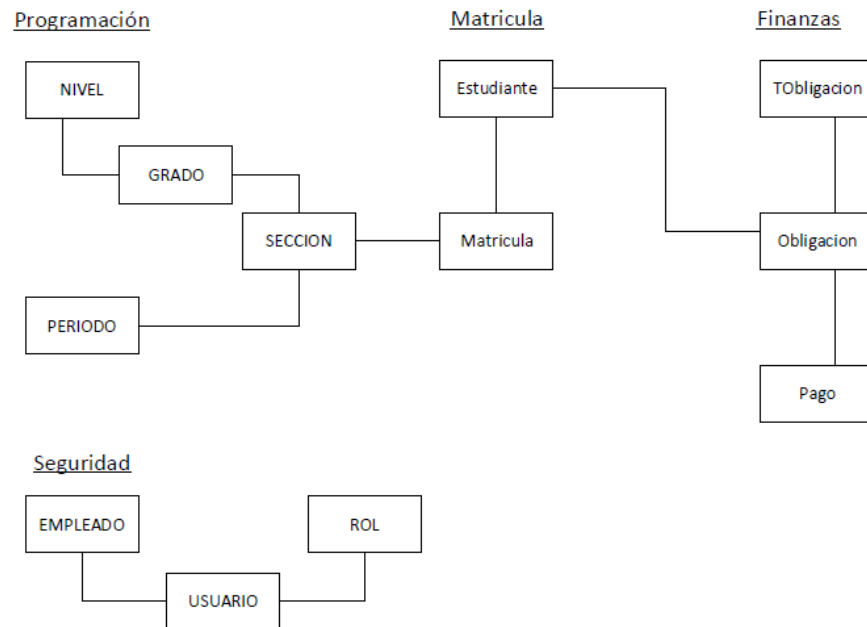
C04	REGISTRAR MATRICULA
<b>Versión:</b>	1.0
<b>Actores:</b>	Alumno y Secretaria
<b>Descripción:</b>	Se realiza el registro de la matrícula de alumnos
<b>Pre-condición:</b>	El alumno no deberá estar matriculado
<b>Post-condición:</b>	Se crea una nueva matricula y con ello las obligaciones de pago
ACTOR	SISTEMA
1. Solicita pantalla de registrar matricula	2. Muestra pantalla solicitando el código de matricula
3. Ingresa el código de matricula	4. Valida el código
	5. Muestra pantalla para la captura de los datos de la matricula
6. Ingresa los datos de matricula	7. Valida los datos ingresados
8. Confirma registro de matricula	
CAMINOS ALTERNATIVOS	
	5. Si la matricula existe muestra pantalla de error
<b>Frecuencia esperada:</b>	Debe usarse con regularidad
<b>Importancia:</b>	Es muy importante



**C05: Registrar pago de mensualidad**

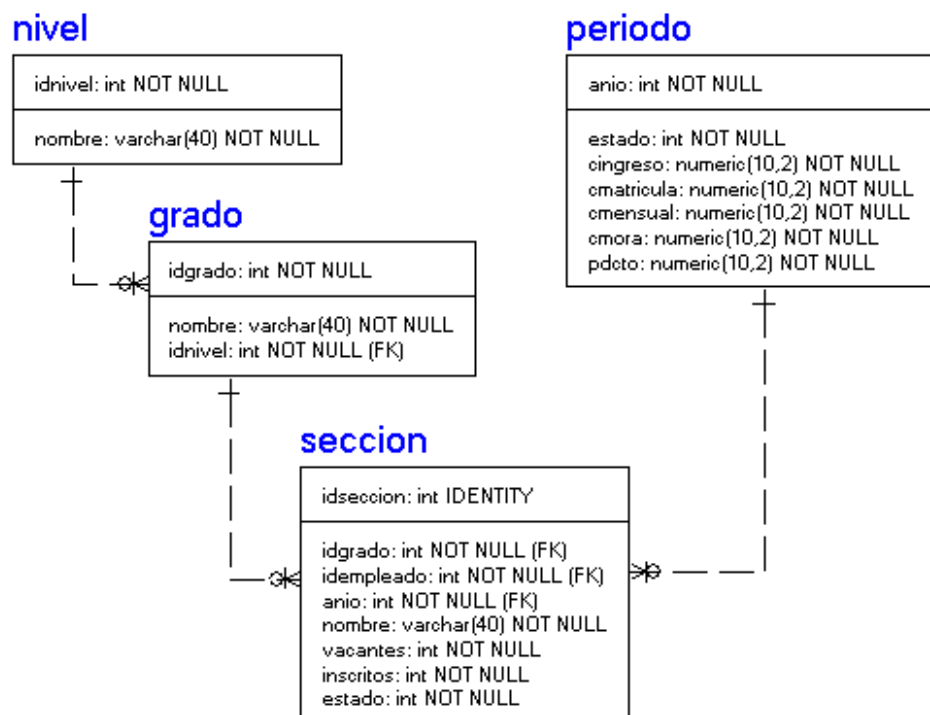
**f. Diseño e implementación de la base de datos**

**i. *Modelo conceptual***

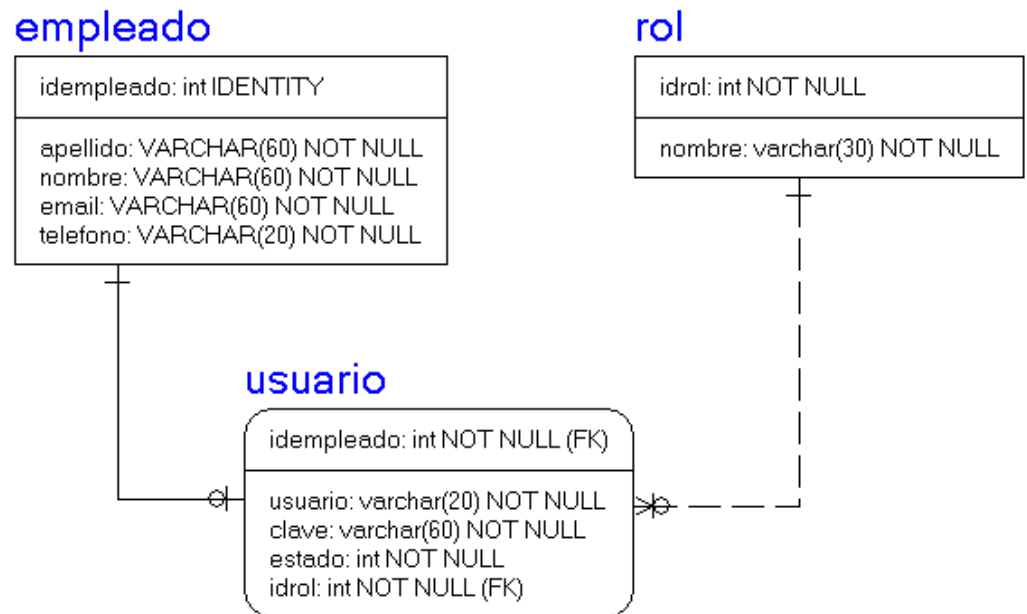


**ii. *Modelo lógico***

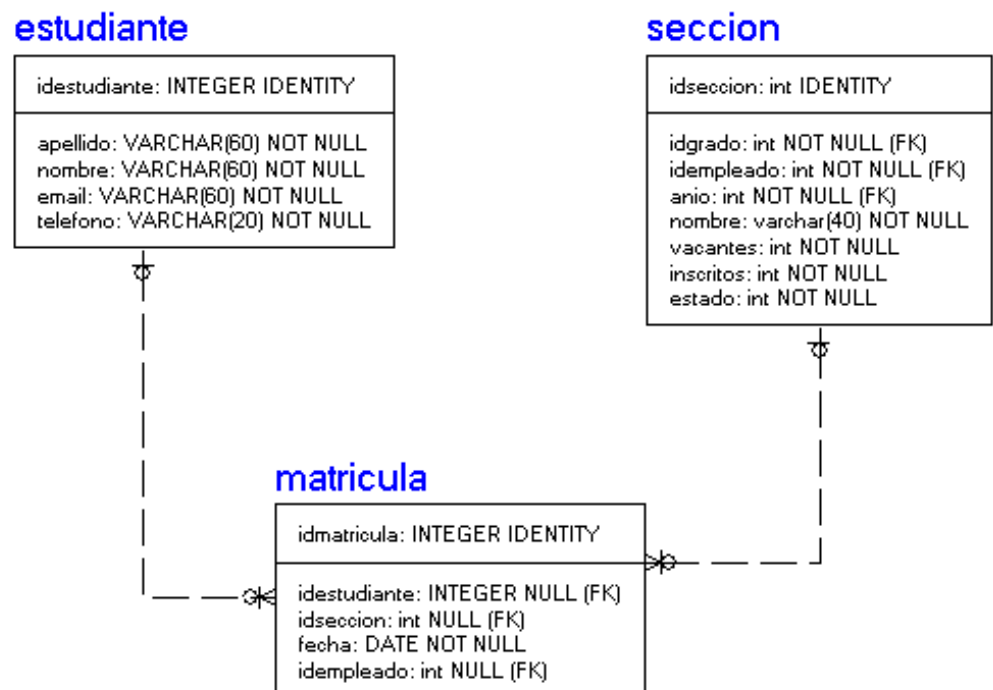
Esquema de Programación:



Esquema de Seguridad:



Esquema de Matricula:





### iii. Código

- Creación de la base de datos:

```

-----
-- base de datos
-----

DROP DATABASE IF EXISTS BDCOLEGIO;

create database BDCOLEGIO;

use BDCOLEGIO;

SET NAMES 'utf8';

-----
-- usuario de base de datos
-----

USE MYSQL;
GRANT ALL PRIVILEGES ON *.* TO 'colegio'@'%' IDENTIFIED BY 'admin' WITH GRANT OPTION;
GRANT ALL PRIVILEGES ON *.* TO 'colegio'@'localhost' IDENTIFIED BY 'admin' WITH GRANT OPTION;
FLUSH PRIVILEGES;
USE BDCOLEGIO;

```

- Esquema de seguridad:

```

-----
-- Esquema: Seguridad
-----

create table empleado(
    idempleado int not null auto_increment,
    apellido varchar(60) not null,
    nombre varchar(60) not null,
    email varchar(60) not null,
    telefono varchar(20) not null,
    constraint pk_empleado primary key(idempleado)
) engine = innodb;

insert into empleado(idempleado,apellido,nombre,
email,telefono) values
(1,'A','B','C','D'),
(2,'A','B','C','D'),
(3,'A','B','C','D'),
(4,'A','B','C','D'),
(5,'A','B','C','D');

create table rol(
    idrol int not null,
    nombre varchar(30) not null,
    constraint pk_rol primary key(idrol)
) engine = innodb;

insert into rol(idrol, nombre) values
(1,'Administrador'),
(2,'Coordinador'),
(3,'Operador'),
(4,'Usuario');

create table usuario(
    idempleado int not null,
    idrol int not null,
    usuario varchar(20) not null,
    clave varchar(60) not null,
    estado int not null check( estado in (0,1) ),
    constraint pk_usuario primary key(idempleado),
    foreign key fk_usuario_empleado(idempleado)
        references empleado( idempleado ),
    foreign key fk_usuario_rol(idrol)
        references rol( idrol )
) engine = innodb;

insert into usuario(idempleado,idrol,usuario,
clave, estado) values
(1,1,'gustavo',SHA('gustavo'),1),
(2,3,'claudia',SHA('claudia'),1),
(3,2,'ricardo',SHA('ricardo'),1),
(4,3,'karla',SHA('karla'),0);

```

- Esquema de Programación:

```
-- -----
-- Esquema: Programacion
-- -----

create table nivel(
  idnivel int not null,
  nombre varchar(40) not null,
  constraint pk_nivel primary key(idnivel)
) engine = innodb;

create table grado(
  idgrado int not null,
  idnivel int not null,
  nombre varchar(40) not null,
  constraint pk_grado primary key(idgrado),
  constraint fk_grado_nivel
    foreign key(idnivel)
    references nivel(idnivel)
) engine = innodb;

insert into nivel(idnivel, nombre) values
(1,'NIVEL PRIMARIA'),
(2,'NIVEL SECUNDARIA');

insert into grado(idgrado,idnivel,nombre) values
(1,1,'PRIMARIA - 1ER GRADO'),
(2,1,'PRIMARIA - 2DO GRADO'),
(3,1,'PRIMARIA - 3ER GRADO'),
(4,1,'PRIMARIA - 4TO GRADO'),
(5,1,'PRIMARIA - 5TO GRADO'),
(6,1,'PRIMARIA - 6TO GRADO'),
(7,2,'SECUNDARIA - 1ER GRADO'),
(8,2,'SECUNDARIA - 2DO GRADO'),
(9,2,'SECUNDARIA - 3ER GRADO'),
(10,2,'SECUNDARIA - 4TO GRADO'),
(11,2,'SECUNDARIA - 5TO GRADO');

create table periodo(
  anio      int not null,
  estado    int not null check( estado in (0,1) ),
  cingreso  numeric(10,2) not null,
  cmatricula numeric(10,2) not null,
  cmensual  numeric(10,2) not null,
  cmora     numeric(10,2) not null,
  pdcto     numeric(10,2) not null,
  primary key pk_periodo( anio )
) engine = innodb;

insert into periodo(anio,estado,cingreso,
cmatricula,cmensual,cmora,pdcto) values
(2017,0,2000.0,800.0,800.0,1.0,0.07),
(2018,1,2500.0,850.0,860.0,1.0,0.08);

create table seccion(
  idseccion int not null auto_increment,
  anio      int not null,
  idgrado   int not null,
  nombre    varchar(40) not null,
  vacantes  int not null,
  inscritos int not null,
  estado    int not null check( estado in (0,1) ),
  idempleado int not null,
  primary key pk_seccion(idseccion),
  foreign key fk_seccion_periodo(anio)
    references periodo(anio),
  foreign key fk_seccion_grado(idgrado)
    references grado(idgrado),
  foreign key fk_seccion_empleado(idempleado)
    references empleado(idempleado)
) engine = innodb;

-- Periodo: Primaria 2017
insert into seccion(idseccion,anio,idgrado,
nombre,vacantes,inscritos,estado,idempleado) values
(1,2017,1,'A',20,15,1,1),
(2,2017,2,'B',20,16,1,1),
(3,2017,3,'C',20,13,1,1),
(4,2017,4,'D',20,16,1,1),
(5,2017,5,'E',20,12,1,1),
```

## - Esquema de Matricula:

```

-----
-- Esquema: Matricula
-----

CREATE TABLE estudiante
(
    idestudiante    INTEGER AUTO_INCREMENT,
    apellido        VARCHAR(60) NOT NULL,
    nombre          VARCHAR(60) NOT NULL,
    email           VARCHAR(60) NOT NULL,
    telefono        VARCHAR(20) NOT NULL,
    PRIMARY KEY PK_ESTUDIANTE(idestudiante)
);

insert into estudiante(idestudiante,apellido,nombre,
email,telefono) values
(1,'Bbb','Cccc','Dddd','Fffff'),
(2,'Bbb','Cccc','Dddd','Fffff'),
(3,'Bbb','Cccc','Dddd','Fffff'),
(4,'Bbb','Cccc','Dddd','Fffff'),
(5,'Bbb','Cccc','Dddd','Fffff');

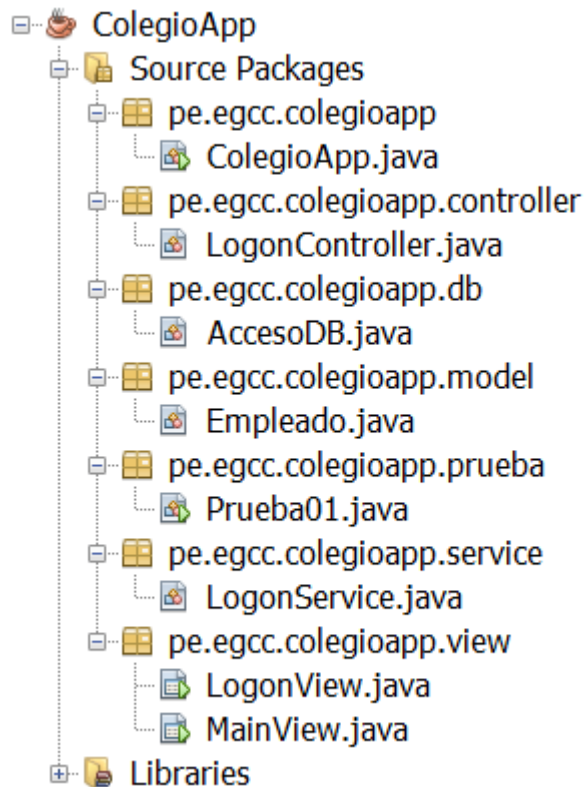
CREATE TABLE matricula
(
    idmatricula      INTEGER AUTO_INCREMENT,
    idestudiante     INTEGER NULL,
    idseccion        int NULL,
    fecha            DATE NOT NULL,
    idempleado       int NULL,
    PRIMARY KEY (idmatricula),
    FOREIGN KEY FK_MATRICULA_ESTUDIANTE (idestudiante) REFERENCES estudiante (idestudiante),
    FOREIGN KEY FK_MATRICULA_SECCION (idseccion) REFERENCES seccion (idseccion),
    FOREIGN KEY FK_MATRICULA_EMPLEADO (idempleado) REFERENCES empleado (idempleado)
);

insert into matricula(idmatricula,
idestudiante,idseccion,fecha,idempleado) values
(1,1,12,'2017-07-15',1),
(2,2,12,'2017-07-15',1),
(3,3,12,'2017-07-15',1);

```

### g. Paquetes, clases y métodos

Para ese proyecto se ha realizado la siguiente estructura con el objetivo de tener ordenado el código y las clases, vistas correctamente empaquetadas.



### h. Código Netbeans

ColegioApp.java:

```
public class ColegioApp {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        LogonView.main(args);
    }

}
```

## LogonController.java

```

package pe.egcc.colegioapp.controller;

import pe.egcc.colegioapp.service.LogonService;

/**
 *
 * @author DOCENTE
 */
public class LogonController {

    public void validar(String usuario, String clave) {

        LogonService service = new LogonService();
        service.validarUsuario(usuario, clave);

        // Manejo de sesiones

    }

}

```

## AccesoDB.java:

```

private AccesoDB() {}

public static Connection getConnection() throws SQLException {
    Connection cn = null;
    try {
        // Datos MySQL
        String driver = "com.mysql.jdbc.Driver";
        String url = "jdbc:mysql://localhost:3306/BDCOLEGIO";
        String user = "colegio";
        String pass = "admin";
        // Cargar el driver a memoria
        Class.forName(driver).newInstance();
        // Obtener el objeto Connection
        cn = DriverManager.getConnection(url, user, pass);
    } catch (SQLException e) {
        throw e;
    } catch (ClassNotFoundException e) {
        throw new SQLException("ERROR, no se encuentra el driver.");
    } catch (Exception e) {
        throw new SQLException("ERROR, no se tiene acceso al servidor.");
    }
    return cn;
}
}

```

## Modelo- Empleado.java

```

public class Empleado {

    private int idempleado;
    private String apellido;
    private String nombre;
    private String email;
    private String telefono;
    private int idrol;

    public Empleado() {
    }

    public int getIdempleado() {
        return idempleado;
    }

    public void setIdempleado(int idempleado) {
        this.idempleado = idempleado;
    }

    public String getApellido() {
        return apellido;
    }

```

---

## LogonService.Java

```

public class LogonService {

    public Empleado validarUsuario(String usuario, String clave){
        Empleado bean = null;
        Connection cn = null;
        try {
            cn = AccesoDB.getConnection();
            String sql = "select e.idempleado,e.apellido,"
                + "e.nombre,e.email,e.telefono,"
                + "u.idrol from empleado e "
                + "join usuario u "
                + "on e.idempleado = u.idempleado "
                + "where u.usuario = ? "
                + "and clave = SHA(?) ";
            PreparedStatement pstmt = cn.prepareStatement(sql);
            pstmt.setString(1, usuario);
            pstmt.setString(2, clave);
            ResultSet rs = pstmt.executeQuery();
            if(rs.next()){
                bean = new Empleado();
                bean.setApellido(rs.getString("apellido"));
                bean.setNombre(rs.getString("nombre"));
                bean.setEmail(rs.getString("email"));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

i. **Interfaces graficas**

i. *Login de usuario:*

En este módulo se realiza la validación de usuario según su perfil, luego el sistema esperará el ingreso de parte del usuario el ingreso del código del usuario y contraseña, respectivamente, si los datos son correctos, el sistema validará los datos, el sistema presenta la siguiente interfaz gráfica.



The image shows a user login interface designed to look like a clipboard. At the top, a blue header bar contains the text "Acceso de Usuario". Below this, the form includes a label "Usuario" followed by a text input field. Underneath is a label "Contraseña" followed by a password input field. Below the password field, there are two links: "Registrarme" and "¿Olvidaste tu contraseña?". At the bottom of the form is a large blue button with the white text "Ingresar".

## ii. Módulo de usuarios

En este módulo se permite registrar a los usuarios. En el caso que sea usuario nuevo previamente el administrador deberá buscar si existe el usuario e ingresar los datos en el menú mantenimiento/usuario, luego de ingresar la información solicitada en los campos posteriores, guardar y el usuario ya se encontrarán registrado, el sistema presenta la siguiente interfaz gráfica.

**Usuario**

Buscar  **Buscar**

**Datos Personales**

Codigo

Nombres

Apellidos

Edad

Direccion

Telefono

Correo

Sexo

Distrito

**Datos de Cuenta**

Usuario

Tipo de acceso

Contraseña

**Nuevo Listar Actualizar Eliminar Agregar**



### iii. Modulo del alumno

En este módulo se permite registrar a los alumnos. En el caso que sea alumno nuevo previamente el administrador deberá buscar si existe e ingresar los datos en el menú mantenimiento/usuario, luego de ingresar la información solicitada en los campos posteriores, guardar y el alumno ya se encontrará registrado, el sistema presenta la siguiente interfaz gráfica.

The screenshot displays the 'Mantenimiento Estudiante' (Student Maintenance) interface. At the top, there is a table listing students with columns for 'Código' (Code), 'Cédula' (ID), and 'Nombre' (Name). Below the table, there are four tabs: 'Datos Alumno' (Student Data), 'Datos Padre Familia' (Parent/Family Data), 'Datos Representante' (Representative Data), and 'Dato Adicional' (Additional Data). The 'Datos Alumno' tab is selected, showing a form with various fields for student information.

Código	Cédula	Nombre
1	102938456	Carlos Andrade
2	102939495	Carlos Estiguar
3	102939495	Marco Anrango
4	10097345435	Blanca Bravo

Below the table, the 'Datos Alumno' tab is active, showing the following fields:

- Código: 1
- Cédula: 102938456
- Apellidos y Nombres: Carlos Andrade
- Sexo: Masculino
- Dirección: Morales
- Fecha Nacimiento: 19/05/1992
- Celular: 23455679
- Teléfono: 23445678
- Institución anterior: Ninguno
- Nacionalidad: Ecuatoriana
- Lugar Nacimiento: Otavalo
- Estado: Activo
- Fecha Ingreso: 23/12/2020

**iv. Módulo de pagos**

Este módulo permite registrar a los pagos. En el caso que sea pago nuevo previamente el administrador deberá buscar si existe el pago e ingresar los datos en el menú mantenimiento/pagos, luego en ingresar la información solicitada en los campos posteriores, subir comprobante de pago y guardar así, el pago ya se encuentra registrado.

**Pagos**

Buscar  **Buscar**

**Datos de Pago**

Codigo de Pago

Codigo de Alumno

Fecha de Pago

Monto de Pago

Tipo de Pago

**Nuevo Listar Actualizar Eliminar Agregar**

Adjuntar boucher  No se ha seleccionado ningún archivo.

v. *Módulo de matrícula*

Este módulo permite registrar las matrículas. En el caso que sea matrícula nueva previamente el administrador deberá buscar si existe la matrícula e ingresar los datos en el menú mantenimiento/matrícula, luego en ingresar la información solicitada en los campos posteriores, guardar y generar la ficha de matrícula, y así la matrícula ya se encuentra registrada.



**Ficha de Matrícula**

Buscar  **Buscar**

**Datos de Matrícula**

Código de Matrícula

Código de Pago

Mes de Matrícula

Año de Matrícula

Tipo de Alumno

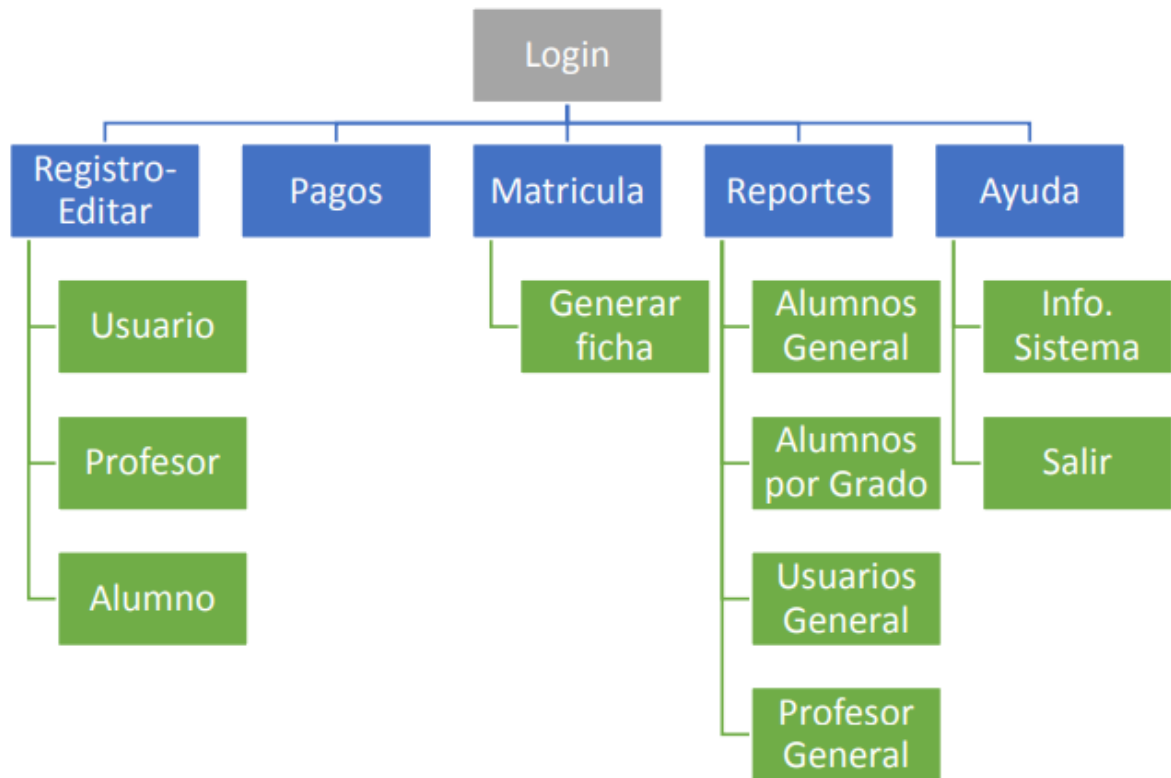
Grado

**Nuevo** **Listar** **Actualizar** **Eliminar** **Generar**

**Agregar**

#### j. **Prototipo**

Se realizó la integración de los prototipos como lo muestra el siguiente diagrama con los módulos desarrollados y la interfaz respectiva de cada uno de ellos con sus funcionalidades de acuerdo a las necesidades de los usuarios



## 4. **Conclusiones y Recomendaciones**

### a. **Conclusiones**

- i. La realización de este proyecto me ha permitido adquirir nuevos conocimientos, pasos y normas necesarios que se deben aplicar para el desarrollo de una aplicación de escritorio.
- ii. Se levantaron los requerimientos mediante información en la web y proyectos anteriores.

- iii. Se logró identificar y aplicar la respectiva metodología de acuerdo a la complejidad y diseño de sistema.
- iv. Considero que este proyecto ha aportado una visión profesional del programador, por la interacción y la puesta en práctica de esa interacción en el sistema.
- v. En general el desarrollo de este proyecto ha permitido ampliar los conocimientos técnicos y poner en práctica todos los conocimientos teóricos adquiridos.

**b. Recomendaciones**

- i. Se recomienda añadir un módulo de anuncios.
- ii. Del mismo modo se recomienda añadir un módulo de horario de clases y calendario académico.
- iii. También es recomendable añadir un módulo de evaluación y control de asistencia de docentes.

**5. Bibliografía:**

- [file:///C:/Users/lsalazar.ITECH/Downloads/cordova\\_ja-2.pdf](file:///C:/Users/lsalazar.ITECH/Downloads/cordova_ja-2.pdf)
- [http://www.repositorioacademico.usmp.edu.pe/bitstream/usmp/1030/1/cordova\\_ja.pdf](http://www.repositorioacademico.usmp.edu.pe/bitstream/usmp/1030/1/cordova_ja.pdf)
- [http://tesis.pucp.edu.pe/repositorio/bitstream/handle/123456789/1046/LOPEZ\\_RENGIFO\\_P\\_AOLO\\_SISTEMA\\_INFORMACION\\_COLEGIO.pdf?sequence=1](http://tesis.pucp.edu.pe/repositorio/bitstream/handle/123456789/1046/LOPEZ_RENGIFO_P_AOLO_SISTEMA_INFORMACION_COLEGIO.pdf?sequence=1)
- <https://repositorio.espe.edu.ec/bitstream/21000/5610/1/T-ESPE-033148.pdf>