UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
SEDE VIÑA DEL MAR - JOSÉ MIGUEL CARRERA

SISTEMA DE CONTROL DE HORARIO PARA ESCUELA "SAINT GREGORY COLLEGE"

Trabajo de titulación para optar al título de
Técnico Universitario en INFORMÁTICA.

Alumnos:
Felipe Javier Lobos Herrera
Sebastián Alonso Pulgar Figueroa

Profesora Guía:
Catherine Gómez Barrera

2019

RESUMEN

El presente trabajo de título, nombrado “Sistema de control de horario para escuela Saint Gregory College”, tiene como objetivo desarrollar un sistema informático que permita facilitar las labores de confección y administración de horarios para la escuela “Saint Gregory College”, institución ubicada en Reñaca Alto en la comuna de Viña del Mar. De carácter particular subvencionado, sus niveles de enseñanza van desde kínder hasta octavo básico. Cuenta con más de 25 docentes y alberga a más de 290 estudiantes.

Este sistema fue desarrolado en Java con NetBean IDE como entorno de desarrollo, utilizando los computadores provistos por la universiad, sumados a los computadores personales de cada uno de los integrantes.

Este informe está dividido en tres capítulos, los cuales serán explicados a continuación:

Capítulo 1: Se describe a fondo la institución a la cual está dirigida el sistema, explicando su situación actual con sus respectivos problemas, también se define el sistema propuesto y los beneficios que traerá para la institución.

Capítulo 2: En esta sección se define el medio ambiente computacional utilizado, detallando computadores y softwares utilizados, así como también la base de datos que se empleará.

Capítulo 3: En este último capítulo se listan las funcionalidades de la aplicación detallando a profundidad algunas de ellas, mostrando sus interfaces de usuario y procesos que llevan a cabo.

Para terminar, se exponen las conclusiones del trabajo y el código de la aplicación.

ÍNDICE

RESUMEN .....

INTRODUCCIÓN 1

1: ASPECTOS RELEVANTES DEL DISEÑO LÓGICO 2

1.1 Descripción de la organización 3

1.2 Descripción de la situación actual 5

1.3 Problemas detectados 6

1.4 Descripción del sistema propuesto 6

1.4.1 Objetivo Principal 6

1.4.2 Objetivos Específicos 7

1.4.3 Beneficios del Sistema 7

1.4.4 Descripción General de la Solución Propuesta 7

1.4.5 Diagrama de Flujo Administrativo 9

1.4.6 Estructura Funcional del Sistema 9

1.4.7 Información que se Manejará 10

1.4.8 Modelo Lógico de Datos 13

1.4.9 Estructura de Código 13

1.4.10 Condicionantes de diseño: 14

2: MEDIO AMBIENTE COMPUTACIONAL Y DESCRIPCIÓN DE ARCHIVOS 16

2.1 Descripción de la característica del recurso computacional. 16

2.1.1: Descripción del Hardware Utilizado. 16

2.1.2 Descripción del Software 17

2.2 Descripción de archivos. 17

3: DESCRIPCIÓN DE PROGRAMAS 24

3.1 Diagrama de Menús 25

3.2 Diagrama de Módulos 26
3.3 Listado de Programas: 27
3.4 Descripción detallada de programas seleccionados 29
3.4.1 Inicio de Sesión 29
3.4.2 Horario Por Profesor. 31
3.4.3 Horario Global 34
3.4.4 Gestionar Profesores. 36
3.4.5 Agregar Profesor. 39
3.4.6 Modificar Profesor. 40
3.4.7 Disponibilidad Profesor. 42
3.4.8 Gestionar Bloques de Horario 43
CONCLUSIONES 50

INTRODUCCIÓN

Para toda institución dedicada a la educación su objetivo principal es prestar un servicio educativo de prestigio, un servicio que logre cumplir la misión de la institución y que haga que ésta destaque sobre otras a la hora de formar a las personas y prepararlas para su futuro en la sociedad de la cual todos somos partícipes. Para esto las instituciones siguen distintos métodos educativos, distintas líneas de pensamiento, distintos enfoques o maneras de enseñar a los alumnos. Pero sin importar sus diferencias en cualquier ámbito toda institución se rige por un horario, el cual tiene gran importancia ya que de éste dependen muchas de las funciones institucionales. Ya sea el horario de secretaría o el horario de clases, todos necesitan estar bien organizados y sin fallos para que todo funcione y los servicios se entreguen como corresponden. Por esto, a la creación de los horarios se les designa mucho tiempo y esfuerzo para lograr que todo funcione como es debido, es un trabajo de más de una persona y completamente manual.

Hoy en día el acceso a la digitalización permite transformar procesos que tardaban semanas en ser completados, a procesos que tardan solo días o incluso horas en ser finalizados. De esta misma forma, en la institución educativa Saint Gregory College, se pretende confeccionar un sistema informático capaz de satisfacer cada una de las necesidades de la institución que se tienen al momento de crear su horario académico, reduciendo los posibles errores al mínimo y el tiempo de elaboración.

CAPITULO 1

ASPECTOS RELEVANTES DEL DISEÑO LOGICO

1: ASPECTOS RELEVANTES DEL DISEÑO LÓGICO

1.1 Descripción de la organización

"Saint Gregory College" es una institución escolar ubicada en Av. Octava 675 lote 363-A Reñaca Alto en la comuna de Viña del Mar. El establecimiento es de carácter particular subvencionado y cuenta con niveles de enseñanza básica y parvulario desde kínder hasta octavo básico, a pesar de tener 10 niveles de educación. Para la institución es habitual tener cursos repetidos pudiendo tener hasta 3 paralelos de un solo curso, esto se produce mucho más en cursos bajos como lo son primero y segundo básico debido a que existen muchos alumnos en estos niveles educativos.

El Colegio Saint Gregory es un colegio que propicia el desarrollo de un contexto afectivo para el aprendizaje desde el establecimiento de relaciones horizontales entre todos sus estamentos. Sin embargo, además de una instancia de educación formal, ha sido para sus docentes, alumnos y apoderados un valioso espacio de participación y acción colectiva en la que, de modo muy local y cotidiano, han puesto a prueba ideales de ser humano, sociedad y convivencia.

Uno de los elementos básicos que define la identidad del Colegio es la gran diversidad de los miembros que componen esta comunidad y su capacidad de convivir e ir dando vida a un sueño común, apoyado por un equipo directivo sólido, docentes involucrados con su labor y con familias comprometidas con el desempeño o desarrollo de sus pupilos.

El colegio tiene como objetivos principales ofrecer una formación académica de calidad mediante métodos de aprendizaje que permitan al alumno acceder a un conocimiento integral, ofrecer espacios de encuentro e integración de todos los integrantes de la unidad educativa en el ámbito deportivo, artístico, cultural y social, proteger y educar en la salud e higiene a los alumnos, mejorar la Calidad de la educación del establecimiento educacional, entre otros.

El colegio está dirigido por la **S**ra. Genoveva Araya Contreras y cuenta con más de 25 docentes, especialistas y funcionarios, teniendo un promedio de alumnos matriculados de 290 por año y cuenta con talleres de reforzamiento en lenguaje y matemáticas además de talleres recreativos especiales como talleres de danza, cine, fútbol, entre otros, que ayudan a fomentar el desarrollo integral de los alumnos.

Organigrama de la institución

A continuación, se presenta el organigrama de la institución (Figura 1-1):

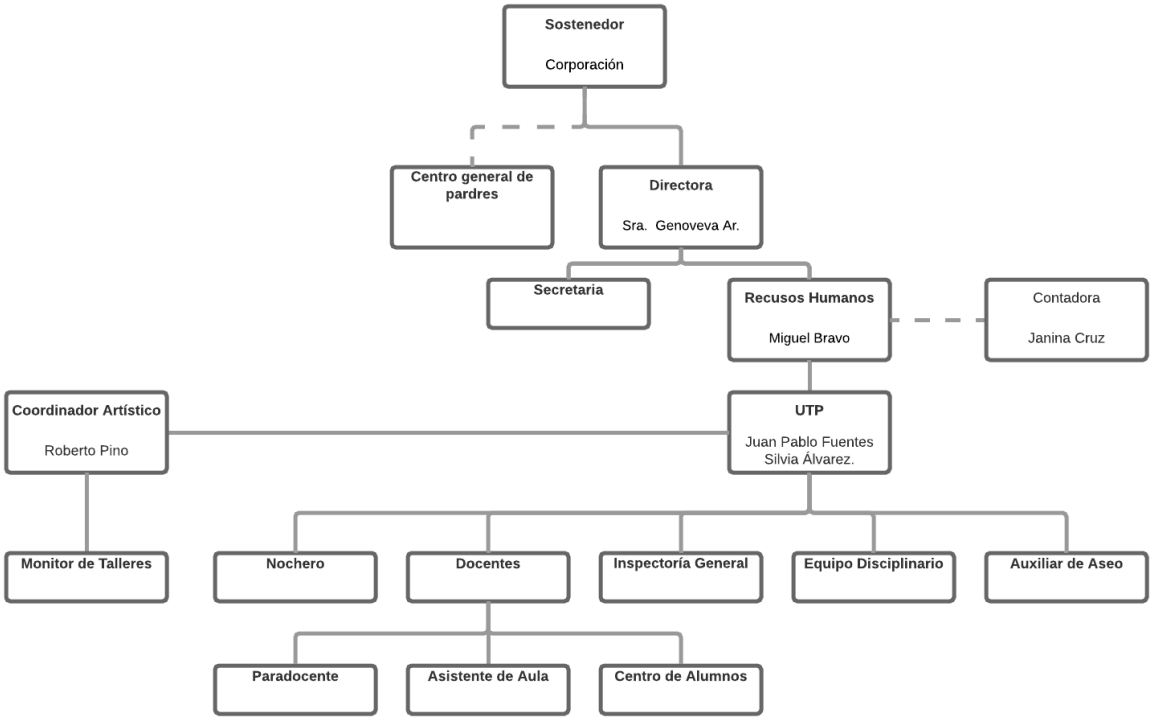


Figura 1-1: Organigrama "Saint Gregory College", Fuente: Directora Genoveva Ar.

Algunos cargos o funciones importantes son:

- Sostenedor: Entidad que percibe el dinero, es el dueño de la institución.
- Director: Administración financiera y pedagógica.
- Centro general de padres: Tiene como función apoyar proyectos para los estudiantes.
- Recursos Humanos: Encargado de financiamiento y contabilidad.
- UTP: Encargado de monitorear la Gestión pedagógica.
- Coordinador Artístico: Encargado de gestionar los talleres.

1.2 Descripción de la situación actual

Hasta el momento la institución ha organizado sus horarios de clases mediante el trabajo conjunto de tres funcionarios, mayormente uno, la directora del establecimiento que se encarga de la construcción y asignación de los horarios y los otros dos funcionarios de UTP se encargan de la revisión de este. El proceso inicia una vez que todos los cursos ya tienen sus salas asignadas y todos los profesores tienen sus contratos listos, en los cuales se especifica cuántas horas trabajan, y que días y horarios pueden tomar clases o asistir a la institución, esto implica que ya se conoce su disponibilidad horaria y la cantidad de horas que debe trabajar a la semana cada uno de ellos.

Para el desarrollo del proceso se usa una plantilla de horarios de gran tamaño (100cm x 80cm) la cual tiene como columnas a los cursos y como filas las horas, este formato se repite 5 veces (uno por cada día de la semana), de esta forma se tiene todos los horarios de la institución donde se imparten las clases, esta plantilla puede variar por año, si aumentan o disminuyen la cantidad de cursos por año o si cambia la ley y se cambia la cantidad de horas que deben estudiar los alumnos en la institución.

Una vez que ya se tiene la plantilla actualizada para el año que corresponde, la directora se dispone a rellenar cada casilla con el nombre del profesor que estará en ese momento (en ese horario y ese curso). La directora debe hacer calzar todos los horarios según los contratos de los profesores y también tratando de seguir un criterio en el cual un profesor que solo hace una asignatura no tenga que venir varias veces a la semana solo a impartir una clase. Es un proceso enteramente manual.

Una vez finalizada la asignación de horarios, se procede a revisar la planilla por parte de UTP en busca de errores, como pueden ser topes, incumplimiento de contrato o que a un profesor se le asigne una hora donde no puede asistir, en caso de encontrar un error se procede a avisarle a la directora para que esta pueda re-asignar horarios para corregir los errores siguiendo los mismos criterios, cabe destacar que el criterio especial para los profesores de solo una clase será ignorado de ser necesario, este proceso se repite hasta que no se encuentran más errores en la revisión y tarda un promedio de un mes en ser acabado.

Una vez finalizado todo el proceso de asignación de horarios toda la plantilla se traslada a un Excel en el cual solo se transforma a un formato más compacto (plantilla por profesor, por curso, etc.) para que puedan ser impresos y entregados a los profesores o a los funcionarios pertinentes.

1.3 Problemas detectados

A continuación, se presentan los problemas encontrados:

- La detección de errores se efectúa de manera manual revisando uno a uno los bloques de horarios lo que toma una gran extensión de tiempo.
- A pesar de todo el tiempo que se ocupa para la detección de topes de horarios o inconsistencias en los horarios, el proceso está sujeto a errores humanos y usualmente quedan problemas sin detectar, los cuales tienen que ser resueltos en el inicio de jornadas escolares.
- No se puede consultar de manera específica ningún horario ya que están todos juntos en una planilla y solo se separan una vez que se cree que el horario está acabado y se procede a traspasar el horario a las planillas de Excel, que se usan para formatear el horario, ya sea por curso, profesor o asignatura.
- El error más frecuente que aparece corresponde a la disponibilidad horaria de cada profesor (los días y horas que pueden asistir a la institución) ya que esta información suele olvidarse con facilidad y se le asignan horarios a profesores en los cuales no pueden asistir.

1.4 Descripción del sistema propuesto

1.4.1 Objetivo Principal

Crear un sistema informático que permita facilitar la confección y administración de los horarios académicos para la institución, otorgando las herramientas necesarias para que el proceso sea claro y rápido.

1.4.2 Objetivos Específicos

- Creación de una base de datos que registre y administre los datos de profesores, cursos, asignaturas, salas, así como también el horario.
- Creación de un sistema con interfaz fácil y cómoda de utilizar, que permite realizar consultas rápidas.
- Que el programa mantenga una constante vigilancia sobre qué cambios realiza el usuario respecto al horario, en busca de los posibles errores que puedan suceder y así alertar al usuario.
- La capacidad de imprimir directamente el horario sin la necesidad de utilizar otro programa.

1.4.3 Beneficios del Sistema

- Evitar la inconsistencia, topes u otros conflictos en el horario que se pudiesen generar al crearlo a mano, otorgando las facilidades necesarias para detectar estos errores.
- Reducir los tiempos de creación y consulta de los horarios.

1.4.4 Descripción General de la Solución Propuesta

La solución propuesta es crear un programa informático de administración de horarios que sea simple, fácil de manipular y de entender, el programa funcionará en uno de los computadores de la institución, el programa contará con niveles de autorización, no tendrá conexión web debido a que es un sistema escolar básico, toda la información y datos que maneje el programa se guardarán en la base de datos que estará albergada en el mismo computador.

En temas de seguridad, se harán copias de seguridad cada vez que el programa se cierre o cada vez que el usuario lo desee pudiendo estas ser trasladadas de manera cómoda a cualquier dispositivo externo como lo puede ser un pendrive. No habrá seguridad de encriptación de datos ya que no se trabajará con información que requiera ser protegida y es de uso público dentro y fuera de la institución.

El sistema permitirá el ingreso y modificación de los datos de profesores, cursos, asignaturas y salones, así como también el formato de los bloques de horarios, dichos datos serán almacenados en la base de datos y utilizados al momento de diseñar el horario.

Como requisito importante en el ingreso de datos para la creación del horario se debe tener la disponibilidad horaria de los profesores, ya que existen profesores que por distintos motivos no podrán asistir a ciertos días o bloques de horarios, el usuario podrá seleccionar por profesor los bloques o días en los cuales puede asistir (por defecto todos están no disponibles). El programa no permitirá que se le asigne un bloque horario a un profesor al cual no pueda asistir.

El programa dispondrá ante el usuario una pantalla con una cuadrícula de bloques de horarios en las cuales se podrá hacer clic en cada una de ellas para asignar un profesor, un curso, una sala y una asignatura desplegando un listbox por cada uno de los atributos, este listbox contendrá los datos almacenados en la base de datos.

Al momento de ir llenando la cuadrícula de bloques de horarios, el programa se encargará de buscar errores y los alertará al usuario en una pequeña ventana similar a la venta de notificaciones ubicada abajo a la derecha de cualquier computador con sistema operativo Windows 10. Las distintas alertas responden a los errores que se quieren evitar los cuales son:

- Topes de Horario: Donde un profesor aparece en dos horarios simultáneamente.
- Exceso de Horas: Donde un profesor se le está asignando un número mayor de horas a las que les corresponde por contrato.
- Falta de Horas Profesor: Donde un profesor no está cumpliendo las horas de su contrato.
- Falta de Horas Curso: Donde algún curso no tenga todos sus días con clases.

1.4.5 Diagrama de Flujo Administrativo

A continuación, se presenta el diagrama de flujo administrativo para este sistema (figura 1-3):

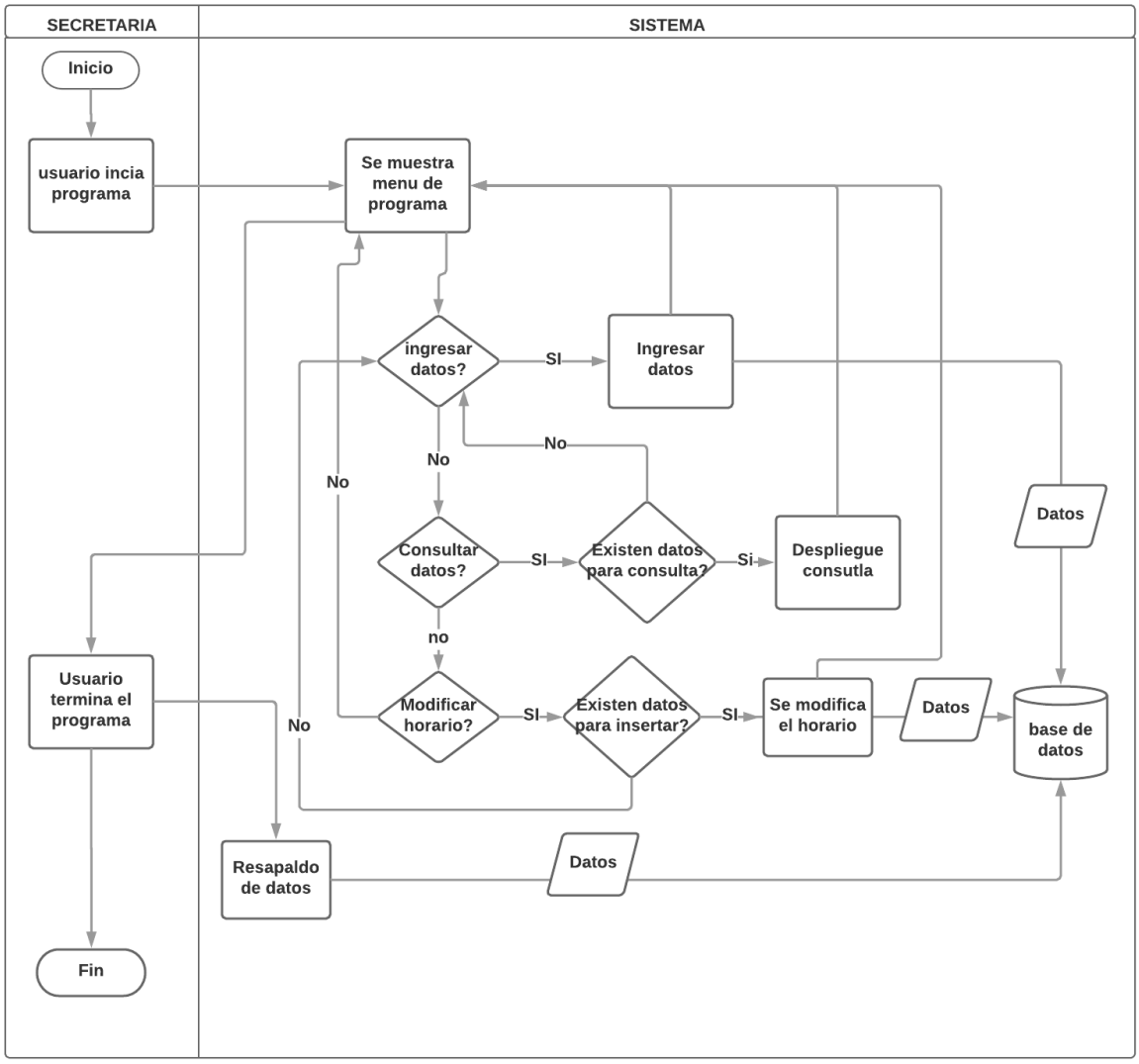


Figura 1-2: Diagrama de Flujo Administrativo.

1.4.6 Estructura Funcional del Sistema

A continuación, se presentan las funcionalidades que tendrá el sistema

- Mantenedor de Profesores: Permitirá el ingreso, modificación, consulta y eliminación de profesores.
- Mantenedor de usuarios: Permitirá el ingreso, modificación, consulta y eliminación de usuarios que harán uso del sistema.

- Mantenedor de Asignaturas: Permitirá el ingreso, modificación, consulta y eliminación de las asignaturas.
- Gestor Disponibilidad de Profesores: Permite asignar los bloques de horario en el que el profesor puede asistir.
- Mantenedor de Cursos: Permitirá el ingreso, modificación, consulta y eliminación de los cursos.
- Mantenedor de Salas: Permitirá el ingreso, modificación, consulta y eliminación de las salas.
- Mantenedor de Implementos: Permitirá el ingreso, modificación, consulta y eliminación de implementos que puedan estar en alguna sala (computadores, instrumentos musicales, elementos de laboratorios, entre otros).
- Gestor de Bloque Horario: Permite el ingreso, modificación y consulta sobre la hora de inicio y la hora de término de los bloques de horario.
- Gestor de Horarios: Permite la visualización, ingreso, validación y modificaciones de las planillas de horarios, tanto específicas (Por profesor, curso o asignatura) como globales.
- Detector de Errores: El programa alertará al usuario de cualquier error que exista en el horario, los cuales son identificados bajo un código específico.
- Inicio de sesión: Permite al usuario iniciar sesión en el sistema con su propia cuenta personal

1.4.7 Información que se Manejará

Entradas:

- Datos Profesor (Rut, nombre, apellido paterno, apellido materno, dirección, Email, horas por contrato, asignatura(s) a cargo, Curso(s) a cargo).
- Datos Bloque Horario (Hora de inicio, hora de término, número de bloque).
- Datos Curso (Código curso, sala curso, cantidad de alumnos).
- Datos Asignatura (Nombre, código de asignatura).
- Datos Implementos (Nombre implemento, código implemento, descripción, cantidad total).

- Datos Sala (Código sala, nombre sala, cantidad máxima de alumnos, código de implemento, cantidad implementos).
- Datos de Usuarios (Nombre de usuario, Contraseña)
- Disponibilidad horaria de los profesores: Hace referencia a los bloques de horarios que cada profesor tendrá disponible para hacer clases al momento de crear el horario.
- Datos del Horario: Son los datos que contendrá cada bloque de horario (profesor, curso, asignatura, sala).

Salidas:

- Listado de profesores: Mostrará un listado detallado de cada profesor.
- Listado de asignaturas: Mostrará un listado de todas las asignaturas con su respectivo código y nombre.
- Listado cursos: Mostrará un listado de todos los cursos con su respectivo código.
- Listado de salas: Mostrará el listado de las salas registradas con sus implementos, si es que tiene.
- Planilla global de horarios: Mostrará de forma gráfica y clara la planilla de horarios completa de la semana.
- Planilla de horarios por profesor: Mostrará de forma gráfica y clara la planilla de horarios por un profesor en específico.
- Planilla de horarios por asignatura: Mostrará de forma gráfica y clara la planilla de horarios por una asignatura en específico.
- Planilla de horarios por curso: Mostrará de forma gráfica y clara la planilla de horarios por un curso en específico.
- Alerta: Mostrará una alerta o advertencia cuando se detecte un error en el horario.

Entidades:

Se plantea manejar las siguientes entidades:

1. Profesor: Guardará todos los datos de los profesores de la institución necesarios para la administración del horario.
2. Curso: Guardará la información de cada curso.

3. Asignatura: Tendrá todas las asignaturas impartidas en la institución.
4. Curso_Asignatura: Guarda información sobre qué asignaturas tendrá cada curso, con horarios de cátedra y laboratorio.
5. Sala: Guardará el nombre de la sala y la cantidad de alumnos que puede albergar.
6. Implementos: Tendrá el nombre de los implementos, la cantidad de estos y una breve descripción opcional.
7. Implementos_sala: Contiene la información de cuantos implementos tiene asignados una sala.
8. Formato_bloque: Guardará el formato de los bloques de horario, su hora de inicio y hora de término, esto permite que sea modificable a futuro.
9. Horario: Es la entidad en la que se administra toda la información necesaria del horario, la cual se usará para desplegar el horario por pantalla
10. Horario_disponibilidad_profesor: Guardará los bloques de horario en los cuales un profesor puede asistir.
11. Usuario: Guardará los usuarios válidos para ingresar al sistema.

1.4.8 Modelo Lógico de Datos

A continuación, se presenta el modelo relacional:

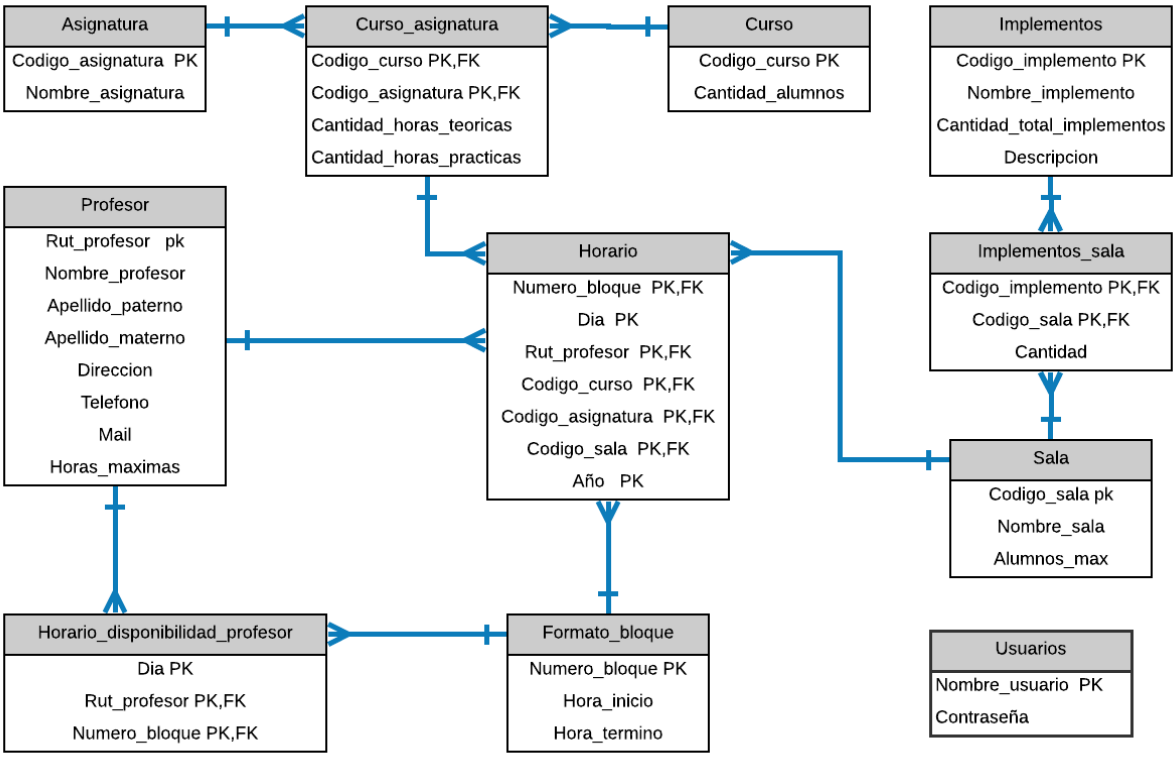


Figura 1-3: Diagrama de modelo relacional.

1.4.9 Estructura de Código

A continuación, se presenta la estructura de código que se utilizará:

- Codigo_asignatura: Corresponde a un código alfanumérico con las iniciales de la asignatura y un número correlativo (mat1, leng1, mus1, mat2, por ejemplo).
- Codigo_curso: Corresponde a un código alfanumérico con el grado del curso y el paralelo (1A, 3A, 3B, por ejemplo).
- Codigo_implemento: Corresponde a un código alfanumérico con las iniciales del implemento y un correlativo (guit1, tamb2, prob1, por ejemplo).
- Codigo_alerta: Corresponde a un número correlativo que se inicia en 1.
- Rut_profesor: Corresponde al Rut de cada profesor (sin puntos, con guión).

- Año: Corresponde a una cifra de cuatro dígitos correspondiente al año (2018 por ejemplo).
- Dia: Corresponde a un número correlativo de un dígito (1=lunes, 2 = martes, etc…)
- Numero_bloque: Corresponde a un número correlativo que se inicia en 1.
- Nombre_usuario: Corresponde al nombre que haya ingresado la persona (alfanumérico).

1.4.10 Condicionantes de diseño:

- El lenguaje que se usará para el desarrollo del programa será Java.
- El sistema que se usará para la gestión de base de datos será MariaDB.
- El ambiente computacional donde el programa será instalado es un PC stand alone marca HP con procesador Intel Pentium de 2,8 GHz, 2GB de memoria RAM y Windows 7 ultimate de 32bits.
- Toda la información que se ingrese o cambie en el programa se guardará una vez el programa se cierre o cuando el usuario lo requiera, el usuario podrá traspasar este archivo de guardado a otro equipo con la aplicación instalada, también se hará un archivo de respaldo (backup), cada cierta cantidad de cambios, en el mismo equipo en caso de cierre inesperado del sistema, son archivos ocultos al usuario y visibles para el sistema y solo serán necesitados en caso de corrupción o error al cargar el archivo de guardado.

CAPITULO 2

MEDIO AMBIENTE COMPUTACIONAL Y DESCRIPCION DE ARCHIVOS

2: MEDIO AMBIENTE COMPUTACIONAL Y DESCRIPCIÓN DE ARCHIVOS

2.1 Descripción de las características del recurso computacional.

En este capítulo se detallarán todos los componentes de hardware y software que se utilizarán en el desarrollo, creación e instalación de la aplicación para la institución “Saint Gregory College”.

2.1.1: Descripción del Hardware Utilizado.

A continuación, se describirán las características del recurso computacional utilizados tanto para el desarrollo del sistema como para el usuario final de éste.

Hardware utilizado para el desarrollo:

El sistema será desarrollado tanto en los computadores personales de cada uno, como en los computadores provistos por la Universidad Técnica Federico Santa María Sede, Viña del Mar.

- Equipo 1: Computador Provisto por la universidad
  - Procesador: Intel Core i5-7500 CPU (4 CPUs) 3,4GHz.
  - Memoria RAM: 8Gb
  - Sistema Operativo: Windows 10 Pro 64 bits.
  - Disco Duro: 1Tb

- Equipo 2: Computador Personal
  - Procesador: AMD A6-5200 Quad-Core processor 2,0GHz.
  - Memoria RAM: 8Gb.
  - Sistema Operativo: Windows 8.1 Pro 64 bits.
  - Disco Duro: 500Gb
- Equipo 3: Computador Personal
  - Procesador: AMD FX(tm) 6300 Six-core processor 3,5 GHz.
  - Memoria RAM: 8Gb.
  - Sistema Operativo: Windows 10 Pro 64 bits.
  - Disco Duro: 1Tb

Hardware del usuario final:

- Procesador Intel Pentium 2,8GHz.
- Memoria RAM: 2Gb.
- Sistema Operativo Windows 7 Ultimate 32 bits.
- Disco Duro:500Gb

2.1.2 Descripción del Software.

A continuación, se describirán las características del software usado en el desarrollo del sistema.

- Sistema Operativo: Windows 10
- Herramienta de Desarrollo de Software:
  - JAVA NetBeans IDE 8.2: Se usará debido a la facilidad que entrega para crear interfaces gráficas.
  - MariaDB: Debido a que la creación de la aplicación se quiere que sea de costo cero para la institución, se usará un gestor de base de datos gratuito como lo es MariaDB.

2.2 Descripción de archivos.

Nomenclatura a utilizar:

Varchar:	Cadena de carácter de largo variable.
Tinyint:	Numérico (0,255).
Integer(Int):	Numérico, entero largo (-2,147,483,648 a 2,147,483,647).
Smallint :	Numérico, entero largo (-32768 a 32767).
Time:	Tipo de dato tiempo con formato HH:MM:SS. (Horas:Minutos:Segundos).
Year:	Numérico de cuatro dígitos.

Tablas o archivos que utilizará el sistema:

2.2.1 Tabla Profesor.

Nombre: Profesor.

Descripción: Tabla que almacenará todos los datos de los profesores.

Clave Primaria: Rut_profesor.

Clave Foránea: No tiene.

Nombre	**Tipo**	**Descripción**
Rut_profesor	Varchar(10)	Rut del profesor.
Nombre_profesor	Varchar(30)	Nombres del profesor.
Apellido_paterno	Varchar(20)	Apellido paterno del profesor.
Apellido_materno	Varchar(20)	Apellido materno del profesor.
Direccion	Varchar(50)	Dirección de vivienda del profesor.
Telefono	Varchar(10)	Teléfono de contacto del profesor.
Email	Varchar(30)	Email de contacto del profesor.
Horas_maximas	Tinyint(2)	Corresponde a la cantidad máxima de horas que puede trabajar según su contrato.

2.2.2 Tabla Curso.

Nombre: Curso.

Descripción: Tabla que almacenará los datos de los cursos.

Clave Primaria: Codigo_curso.

Clave Foránea: No tiene

Nombre	Tipo	Descripción
Codigo_curso	Varchar(6)	Código identificador del curso.
Cantidad_alumnos	Tinyint(2)	Corresponde a la cantidad actual de alumnos matriculados en tal curso.

2.2.3 Tabla Sala.

Nombre: Sala.

Descripción: Tabla que almacenará los datos de las salas

Clave Primaria: Codigo_sala.

Clave Foránea: No tiene.

Nombre	Tipo	Descripción
Codigo_sala	Varchar(6)	Código identificador de la sala.
Nombre_sala	Varchar(20)	Corresponde al nombre de la sala.
Alumnos_max	Tinyint(2)	Corresponde a la cantidad máxima de alumnos que soporta la sala.

2.2.4 Tabla Asignatura.

Nombre: Asignatura.

Descripción: Tabla que almacenará los datos de las asignaturas.

Clave Primaria: Codigo_asignatura.

Clave Foránea: No tiene.

Nombre	Tipo	Descripción
Codigo_asignatura	Varchar(6)	Código identificador de asignatura, compuesto por parte del nombre de una

		asignatura y un número correlativo.
Nombre_asignatura	Varchar(20)	Nombre de la asignatura.

2.2.5 Tabla Formato_bloque.

Nombre: Formato_bloque.

Descripción: Tabla que almacenará los datos del formato de bloques de horario, para luego ser usados en el horario mismo.

Clave Primaria: Numero_bloque.

Clave Foránea: No tiene.

Nombre	**Tipo**	**Descripción**
Numero_bloque	Tinyint(2)	Código correlativo para identificar un bloque.
Hora_inicio	Time	Corresponde a la hora de inicio del bloque.
Hora_termino	Time	Corresponde a la hora de término del bloque.

2.2.6 tabla Implemento.

Nombre: Implemento.

Descripción: Tabla que almacenará los datos de los implementos.

Clave Primaria: Codigo_implemento.

Clave Foránea: No tiene.

Nombre	**Tipo**	**Descripción**
Codigo_implemento	Varchar(6)	Código identificador de un implemento.
Nombre_implemento	Varchar(20)	Corresponde al nombre del implemento.
Cantidad_total_implementos	Smallint(3)	Corresponde a la cantidad total de ese implemento existente en el establecimiento.

2.2.7 Tabla Curso_asignatura.

Nombre: Curso_asignatura.

Descripción: Tabla que almacenará la información sobre qué asignatura tendrá cada curso, con cantidad de horas prácticas y teóricas.

Clave Primaria: Clave compuesta por Codigo_curso + Codigo_asignatura.

Clave Foránea: Codigo_curso (referencia Curso), Codigo_asignatura (referencia Asignatura).

Nombre	**Tipo**	**Descripción**
Codigo_curso	Varchar(6)	Código identificador del curso.
Codigo_asignatura	Varchar(6)	Código identificador de la asignatura, compuesto por parte del nombre de una asignatura y un número correlativo.
Cantidad_horas_teoricas	Tinyint(2)	Corresponde a la cantidad de horas teóricas para la asignatura en tal curso.
Cantidad_horas_practicas	Tinyint(2)	Corresponde a la cantidad de horas prácticas para la asignatura en tal curso.

2.2.8 Tabla Implementos_sala.

Nombre: Implementos_sala.

Descripción: Tabla que almacenará cuantos implementos estan asignados a una cierta sala

Clave Primaria: Clave compuesta por Codigo_implemento + Codigo_sala.

Clave Foránea: Codigo_implemento (referencia Implementos), Codigo_sala (referencia Sala).

Nombre	**Tipo**	**Descripción**
Codigo_sala	Varchar(6)	Código identificador de la sala.

Codigo_implemento	Varchar(6)	Código identificador de un implemento.
Cantidad	Smallint(3)	Corresponde a la cantidad de implementos que tiene asignada una sala.

2.2.9 Tabla Usuarios.

Nombre: Usuarios

Descripción: Tabla que almacenará los usuarios válidos.

Clave Primaria: Nombre_usuario

Clave Foránea: No tiene.

Nombre	**Tipo**	**Descripción**
Nombre_usuario	Varchar(20)	Nombre del usuario el cual no puede repetirse entre todos los usuarios registrados.
Contraseña	Varchar(20)	Contraseña del usuario que se usará para validar el ingreso al sistema.

2.2.10 Tabla horario_disponibilidad_profesor.

Nombre: Horario_disponibilidad_profesor.

Descripción: Tabla que almacenará los bloques de horario a los cuales un profesor puede asistir en un cierto día.

Clave Primaria: Clave compuesta por Dia + Rut_profesor + Numero_bloque.

Clave Foránea: Rut_profesor (referencia Profesor), Numero_bloque (referencia Formato_bloque).

Nombre	**Tipo**	**Descripción**

Rut_profesor	Varchar(10)	Rut del profesor.
Numero_bloque	Tinyint(2)	Código correlativo para identificar un bloque.
Dia	Tinyint(1)	Corresponde a un número correlativo de un dígito (1=Lunes, 2 = Martes, etc…).

2.2.11 Tabla Horario.

Nombre: Horario.

Descripción: Tabla que almacenará toda la información del horario.

Clave Primaria: Clave compuesta por Numero_bloque + Dia + Rut_profesor + Codigo_Curso + Codigo_asignatura + Codigo_sala + Año.

Clave Foránea: Numero_bloque (referencia Formato_bloque), Rut_profesor (Referencia Profesor), Codigo_curso (Referencia Curso_asignatura) +

Codigo_asignatura (Referencia Curso_asignatura), Codigo_sala (referencia Sala).

Nombre	**Tipo**	**Descripción**
Numero_bloque	Tinyint(2)	Código correlativo para identificar un bloque.
Dia	Tinyint(1)	Corresponde a un número correlativo de un dígito (1=Lunes, 2 = Martes, etc…).
Rut_profesor	Varchar(10)	Rut del profesor.
Codigo_Curso	Varchar(6)	Código identificador del curso.
Codigo_asignatura	Varchar(6)	Código identificador de asignatura.
Codigo_sala	Varchar()	Código identificador de la sala.
Año	Year	Corresponde al año en el cual existe este horario.

CAPITULO 3

DESCRIPCIÓN DE PROGRAMAS

3: DESCRIPCIÓN DE PROGRAMAS

3.1 Diagrama de Menús

A continuación, se muestra el diagrama de menús del sistema

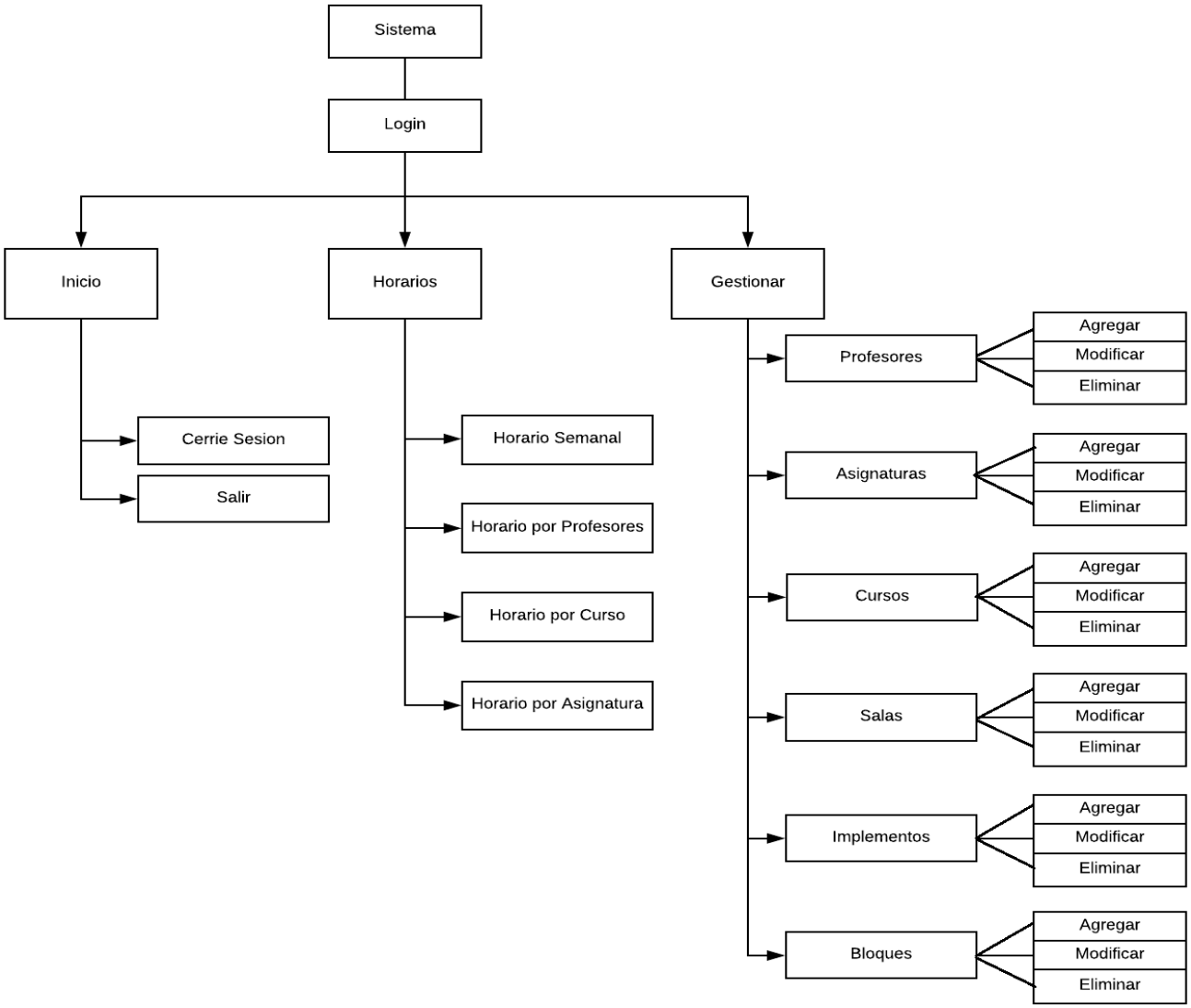


Figura 3-1: Diagrama de menús.

En el instante en que el usuario ingrese a alguna de la opciones de “Gestionar” (ejemplo: Gestionar > Profesores) se listaran los datos correspondientes (ejemplo: La listado detallado de todos los profesores existentes).

3.2 Diagrama de Módulos

A continuación, se muestra el diagrama de modulos del sistema.

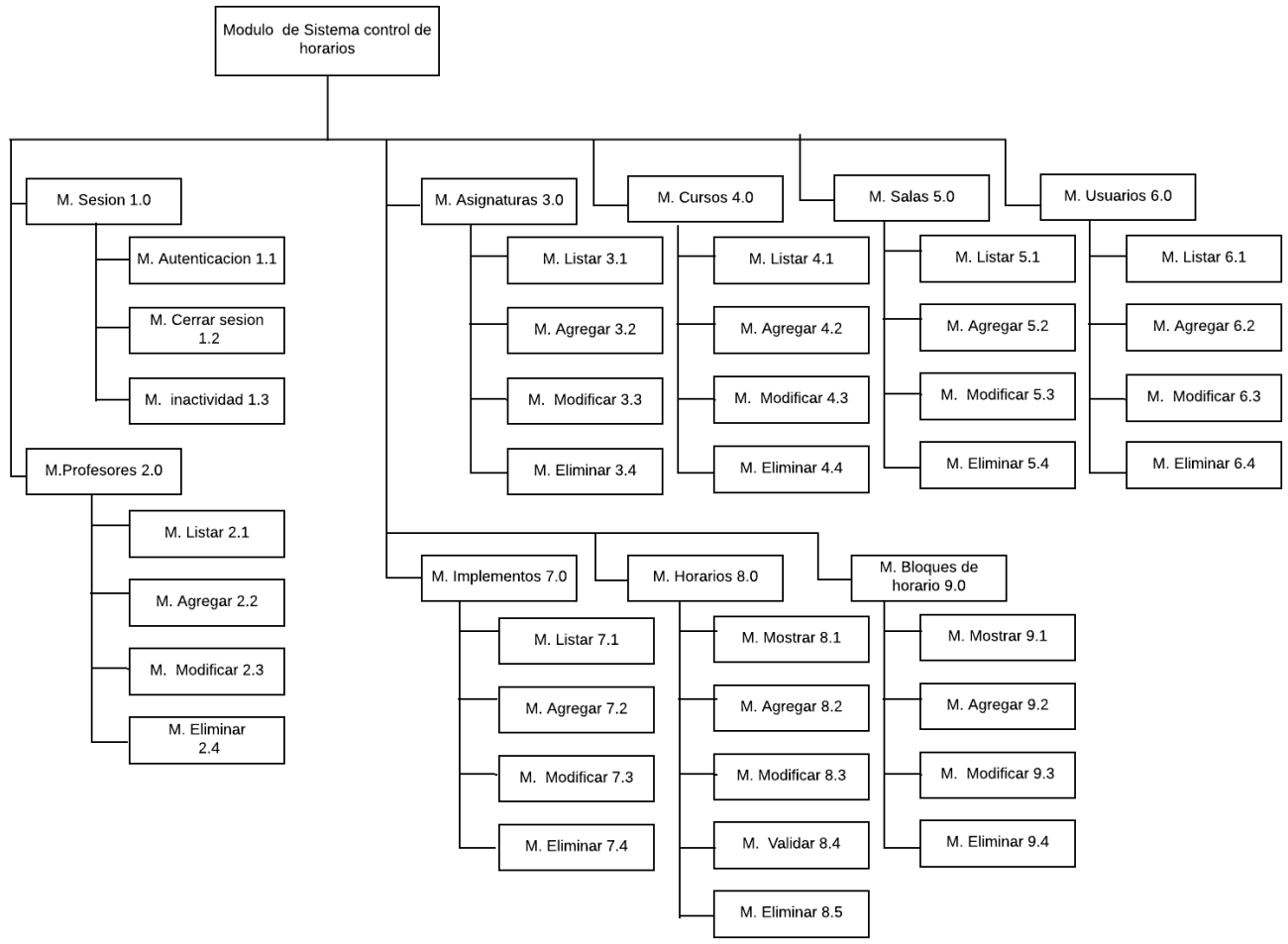


Figura 3-2: Diagrama de Módulos.

3.3 Listado de Programas:

A continuación, se presenta una lista con la totalidad de los programas que conforman el sistema. Señalando con un (*) los programas que serán detallados en el siguiente punto.

N°	Nombre del Programa	Objetivo
1	Inicio de Sesión (*)	Controlar el ingreso a la aplicación principal.
2	Cerrar Sesión	Cerrar la sesión actual y volver a la ventana de "Inicio de Sesión".
3	Salir	Cierra completamente el sistema.
4	Horario por Profesores (*)	Permite seleccionar un profesor y desplegar el horario con todos los bloques de horario en el que se encuentra dicho profesor, también permite editar el horario.
5	Horario por Cursos(*)	Permite seleccionar un curso y desplegar el horario con todos los bloques de horario en el que se encuentra dicho curso, también permite editar el horario.
6	Horario por Asignaturas	Permite seleccionar una asignatura y desplegar el horario con todos los bloques de horario en el que se encuentra dicha asignatura, también permite editar el horario.
7	Horario Global (*)	Despliega cinco tablas de horario (una por cada día de la semana) donde cada una de ellas está dividida por los cursos y los bloques de horarios y se puede apreciar al profesor que le corresponde a un curso en un cierto bloque horario.
8	Gestionar Profesores (*)	Pantalla principal del mantenedor de profesores en la cual se lista la totalidad de profesores existentes, permite eliminar un profesor seleccionado y también muestra los respectivos accesos a Agregar, Modificar, y Disponibilidad de Horario.
9	Agregar Profesor (*)	Permite agregar un nuevo profesor
10	Modificar Profesor (*)	Permite modificar el profesor seleccionado en "Gestionar Profesor"
11	Gestionar Asignaturas	Pantalla principal del mantenedor de asignaturas en la cual se lista la totalidad de asignaturas existentes, permite eliminar

		una asignatura seleccionada y también muestra los respectivos accesos a Agregar, Modificar y Eliminar.
12	Agregar Asignatura	Permite agregar una nueva asignatura.
13	Modificar Asignatura	Permite modificar la asignatura seleccionada en “Gestionar Asignaturas”.
14	Gestionar Cursos	Pantalla principal del mantenedor de cursos en la cual se lista la totalidad de cursos existentes, permite eliminar un curso seleccionado y también muestra un listado de las asignaturas registradas para un curso al ser seleccionado y los respectivos accesos a Agregar, Modificar y Eliminar.
15	Agregar Curso	Permite agregar un nuevo curso.
16	Modificar Curso	Permite modificar el curso seleccionado en “Gestionar Cursos”
17	Gestionar Salas	Pantalla principal del mantenedor de salas en la cual se lista la totalidad de salas existentes, permite eliminar una sala seleccionada y también muestra un listado de los implementos que se encuentran en una sala al ser seleccionada y los respectivos accesos a Agregar, Modificar y Eliminar.
18	Agregar Sala	Permite agregar una nueva sala.
19	Modificar Sala	Permite modificar la sala seleccionada en “Gestionar Sala”.
20	Gestionar Implementos	Pantalla principal del mantenedor de implementos en la cual se lista la totalidad de implementos existentes, permite eliminar un implemento seleccionado y también muestra los respectivos accesos a Agregar, Modificar y Eliminar.
21	Agregar Implementos	Permite agregar un nuevo implemento.
22	Modificar Implementos	Permite modificar el implemento seleccionado en “Gestionar Implementos”.
23	Gestionar Usuarios	Pantalla principal del mantenedor de usuarios en la cual se lista la totalidad de usuarios existentes, permite eliminar un

		usuario seleccionado y también muestra los respectivos accesos a Agregar, Modificar y Eliminar.
25	Agregar Usuario	Permite agregar un nuevo usuario.
26	Modificar Contraseña Usuario	Permite modificar la contraseña a un usuario seleccionado.
27	Disponibilidad Profesor(*)	Permite visualizar y editar la disponibilidad total del profesor seleccionado en “Gestionar Profesores”.
28	Gestionar Bloques de Horario (*)	Permite visualizar, agregar, modificar y eliminar los bloques de horarios de los cuales se basan todas las tablas de horario del sistema.

3.4 Descripción detallada de programas seleccionados

3.4.1 Inicio de Sesión

- **Nombre:** Inicio de Sesión.
- **Objetivo:** Este programa permitirá controlar el ingreso a la aplicación mediante un control de usuarios (Nombre de usuario y Contraseña de Usuario).
- **Diagrama de Bloques:**

- **Reglas del Proceso:**
  - La pantalla de inicio de Sesión (Figura 3-3) será el primer programa en ejecutar una vez iniciada la aplicación.
  - Cuando el usuario ingrese su nombre de usuario y contraseña en los cuadros de texto correspondiente y presione el botón “Aceptar” de la pantalla de inicio de sesión, el programa verificará dichos datos con los almacenados en la base de datos.

- Si los datos ingresados son válidos, esta ventana se cerrará y ejecutará la ventana de menú principal.
- Si los datos ingresados no son válidos, se mostrará un mensaje de error indicando el problema (Figura 3-4).

- **Diseño de Pantalla :**

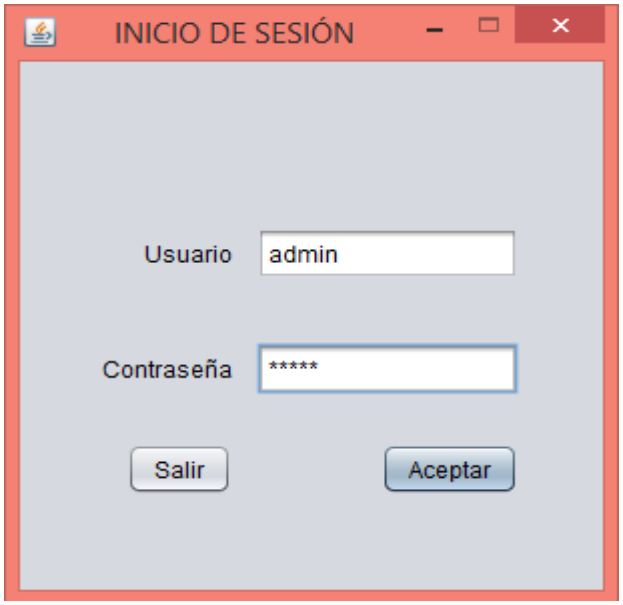


Figura 3-3: Pantalla Inicio de Sesión.

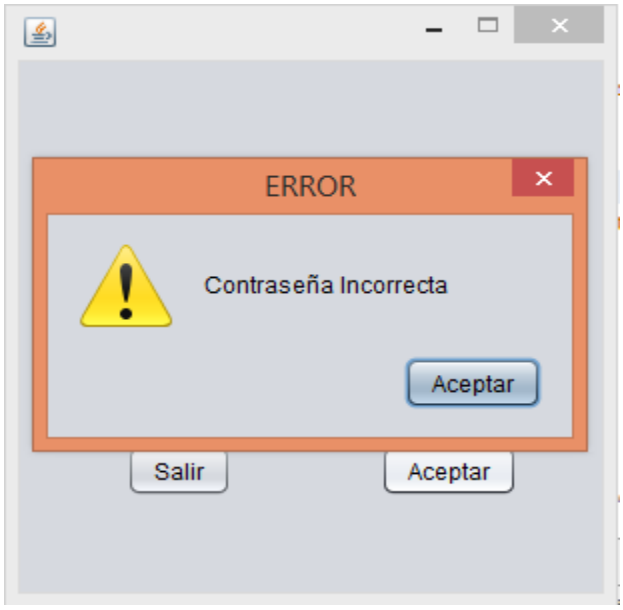
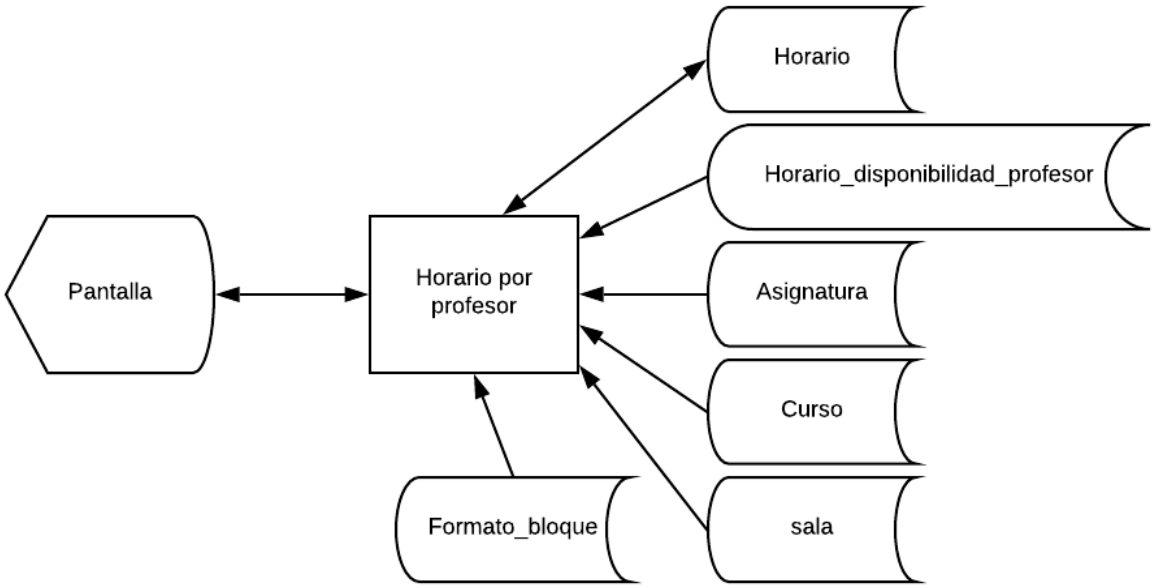


Figura 3-4: Pantalla Inicio de Sesión – Contraseña incorrecta.

- **Referencias: Anexos (pag. 51– pag. 57).**

3.4.2 Horario Por Profesor.

- **Nombre:** Horario por Profesor
- **Objetivo:** Mostrar de forma clara y sencilla el horario de cada profesor registrado en la base de datos, además de permitir editar dicho horario de una manera intuitiva, mostrando en donde tiene disponibilidad el profesor en cuestión y señalando los errores que pudiesen ocurrir.
- **Diagrama de Bloques:**



- **Reglas del Proceso:**
  - Una vez el usuario haya ingresado en la sección (Horarios –> Horario Por Profesores), se desplegará la pantalla de horario por profesores (Figura 3-5).
  - El usuario puede seleccionar el profesor por el cual desea consultar el horario en la parte superior de la pantalla (Figura 3-7), al momento de cambiar de profesor seleccionado, se cargarán desde la base de datos todos los bloques de horarios (Tabla Horario) en los cuales aparece, además del bloque en los cuales tiene disponible para ocupar (Tabla Horario_disponibilidad_profesor).
  - El usuario puede seleccionar una casilla de la tabla para editarla si gusta. Presionando el botón “Borrar” de la parte inferior derecha, el programa eliminará los datos de esa casilla volviéndola a dejar “Disponible” para ese profesor.

- De la misma manera, el usuario puede insertar un nuevo horario ocupado para el profesor presionando el botón “Insertar”. Al momento de presionarlo (y habiendo seleccionado previamente una casilla en estado “Disponible”, de lo contrario, se enviará un mensaje de error, el programa también verificará que la el profesor no exceda su cantidad de horas máximas con la nueva hora seleccionada) el programa tomará los datos de Asignatura, Curso y Sala que el usuario haya seleccionado y los ingresará en la casilla al mismo tiempo que se almacena ese nuevo dato en la base de datos (tabla Horario).
- En caso de generarse algún error, como tope de horario entre cursos o profesores, se le notificará al usuario con un mensaje en la parte inferior de la tabla y la operación no se llevará a cabo (Figura 3-6, se genera un error con un curso que intenta ser asignado nuevamente en un espacio de tiempo donde ya había sido asignado).

- **Diseños de Pantalla:**

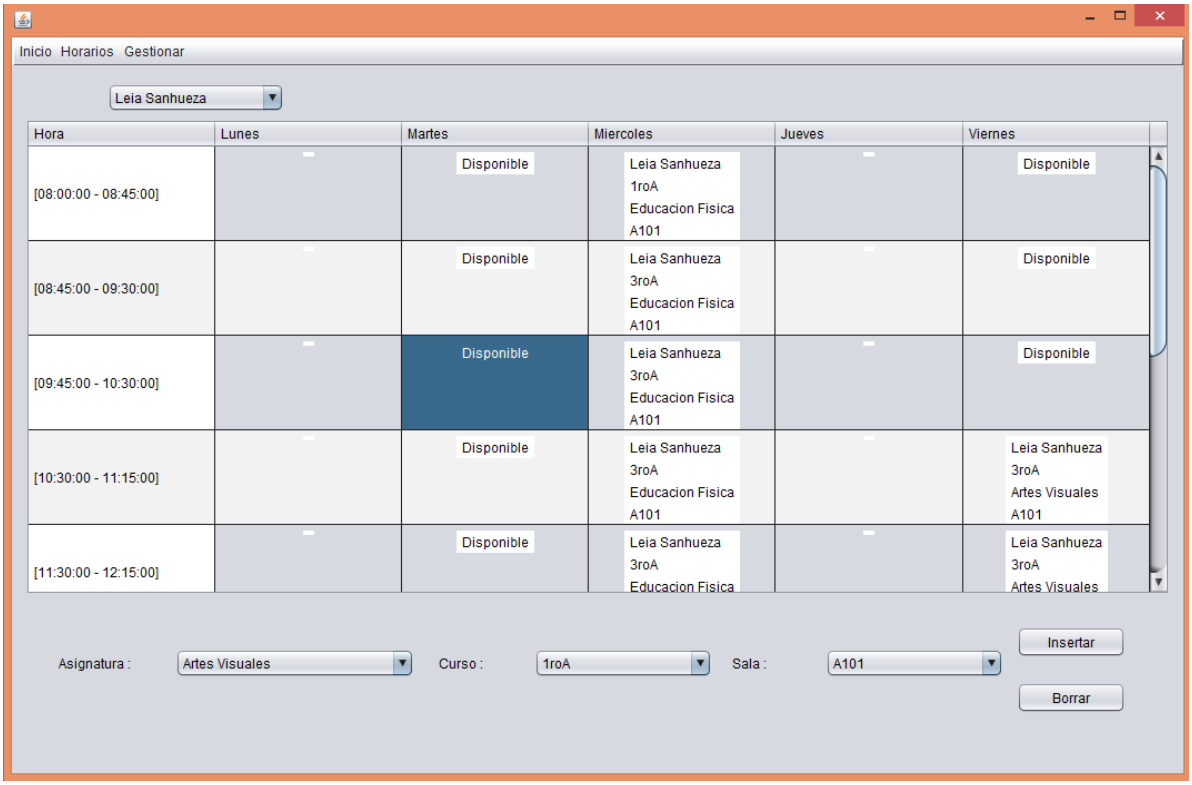


Figura 3-5: Pantalla de Horario por Profesor.

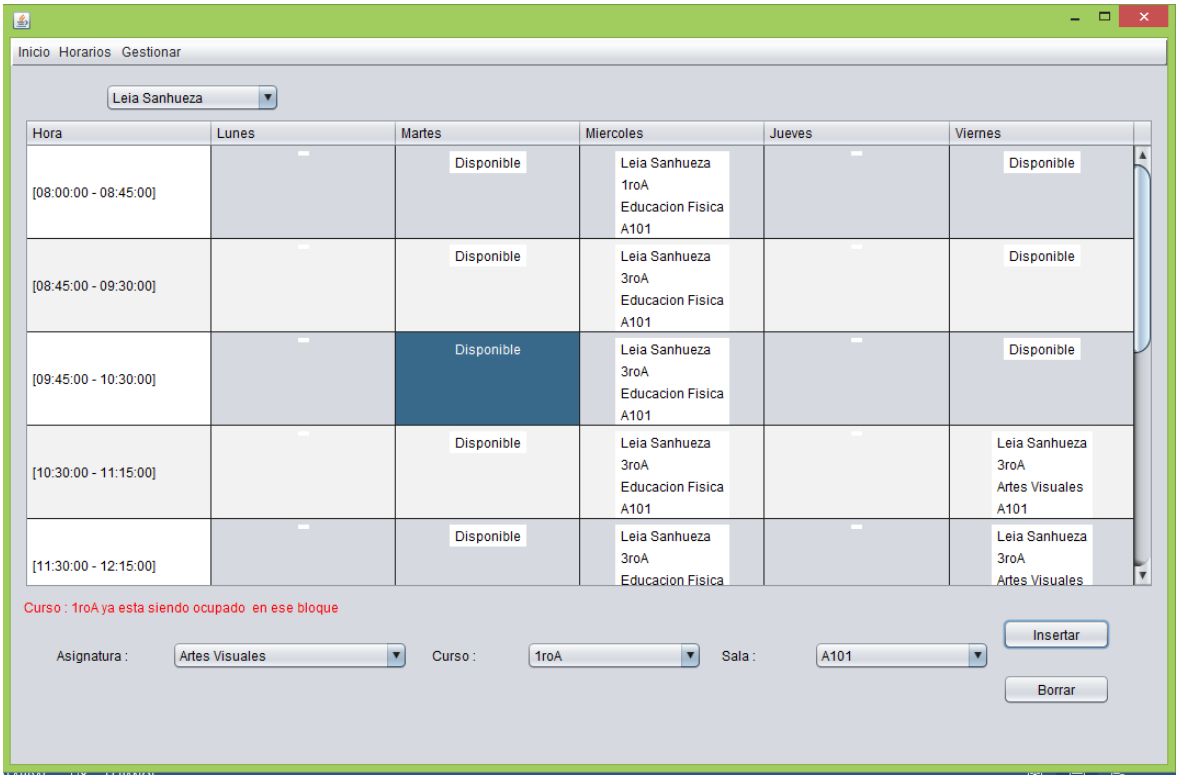


Figura 3-6: Pantalla de Horario Por Profesor – Error de inserción.

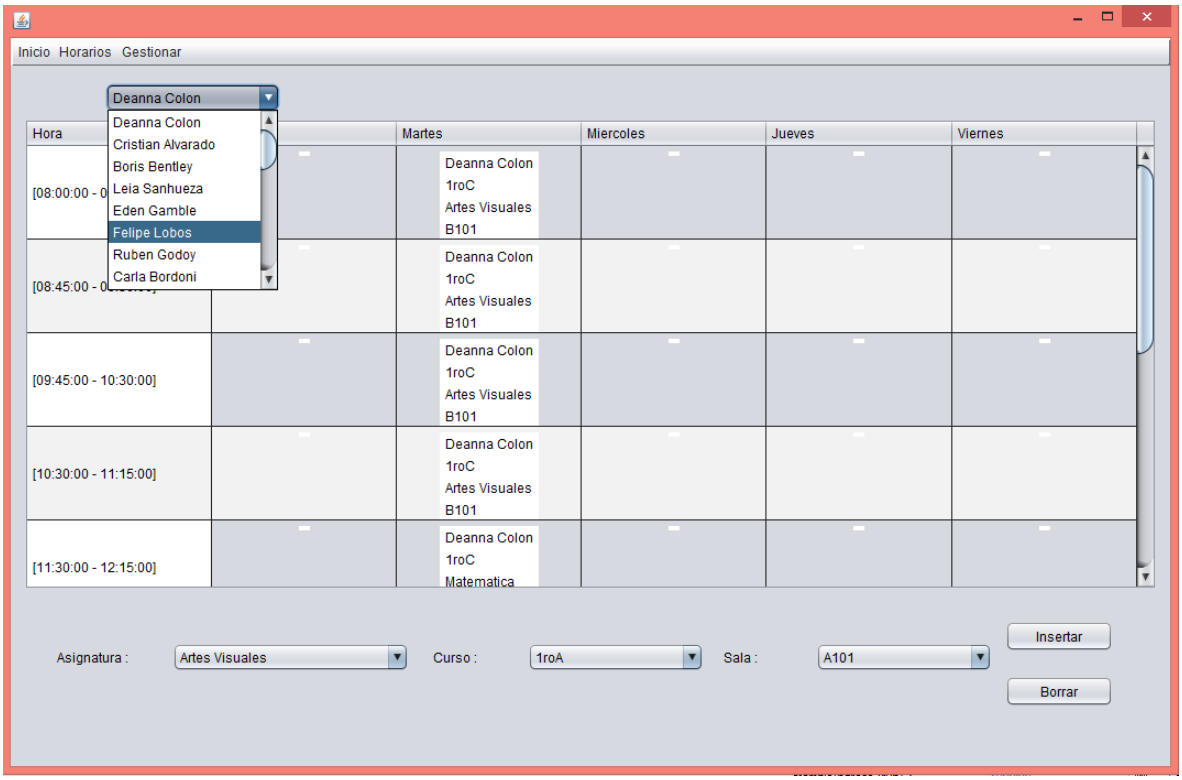
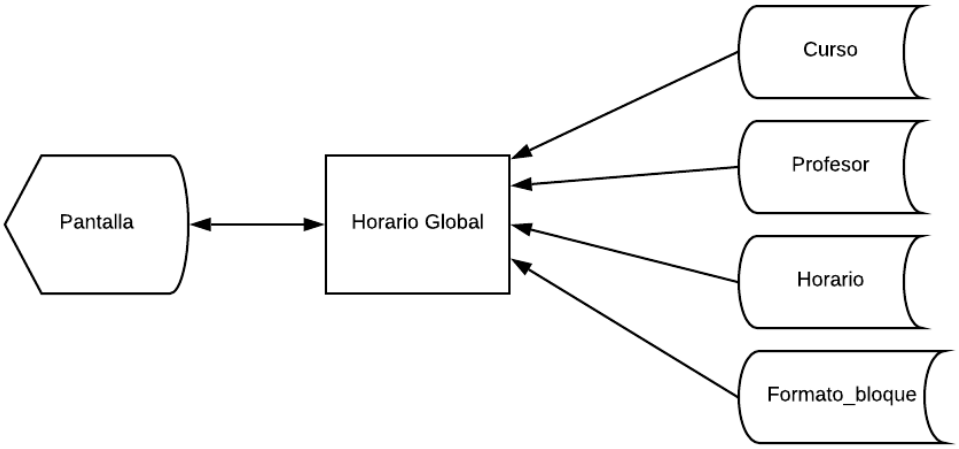


Figura 3-7: Pantalla de Horario Por Profesor (Seleccionar un profesor).

- **Referencias: Anexos (pag. 69 – pag.81)**

3.4.3 Horario Global

- **Nombre:** Horario Global.
- **Objetivo:** Mostrar de forma clara y sencilla el horario semanal de todos los cursos existentes en la institución.
- **Diagrama de bloques:**



- **Reglas del Proceso:**
  - Se accede a través del menú principal, ingresando en la sección “Horarios” y luego “Horario Global”, lo que desplegará por pantalla horarios (uno por cada día de la semana) en los cuales las columnas son cada uno de los cursos existentes en la institución y las filas corresponden a los bloques de horario (Figura 3-8 y Figura 3-9).
  - Esta pantalla solo permite visualizar y no editar los horarios.

- **Diseños de Pantalla:**

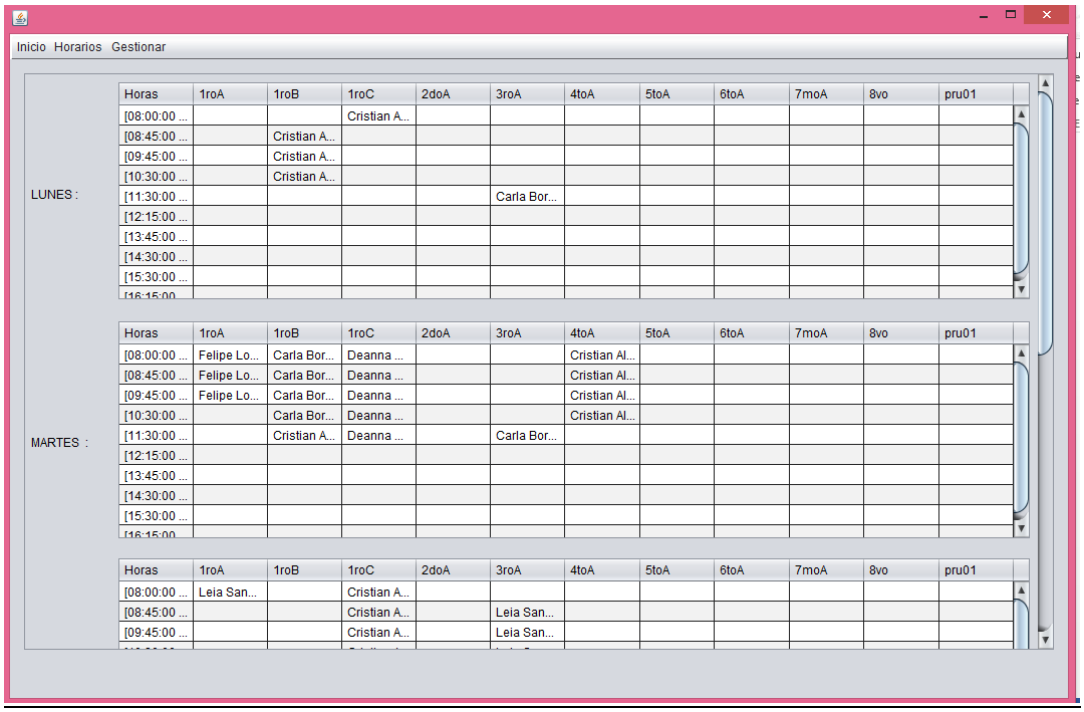


Figura 3-8 Pantalla de horario Global (lunes y martes).

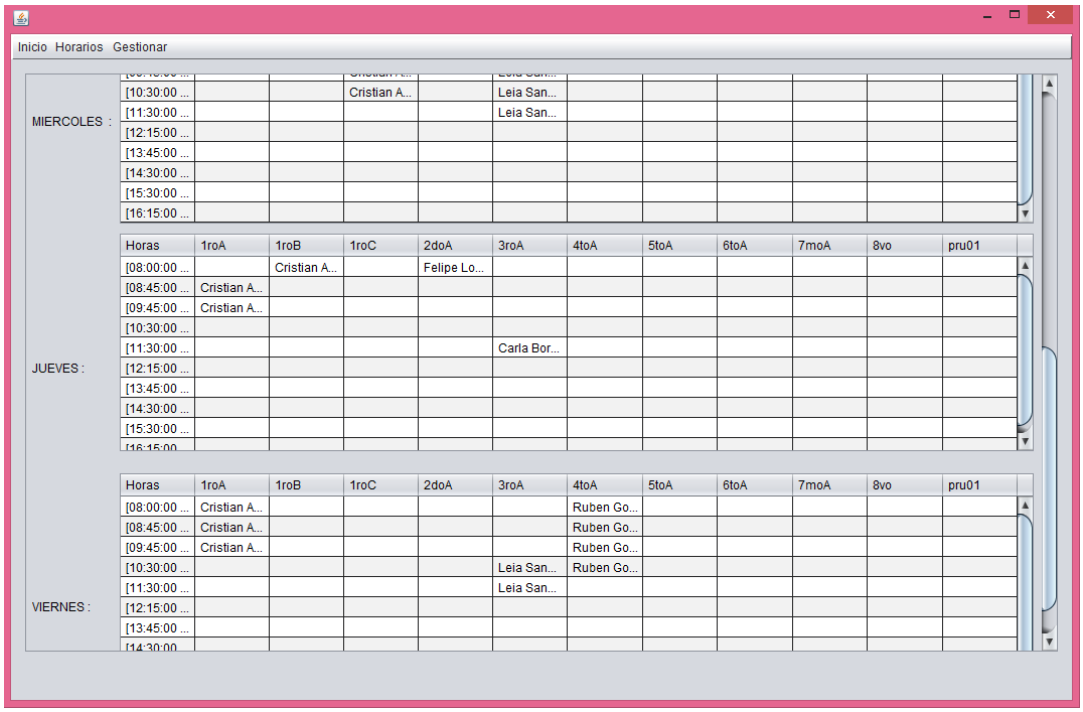


Figura 3-9 Pantalla de horario Global (miércoles, jueves y viernes).

- **Referencias: Anexos (pag. 81 – pag. 90)**

3.4.4 Gestionar Profesores.

- **Nombre:** Gestionar Profesores.
- **Objetivo:** Es la pantalla principal del mantenedor de profesores, en ella se puede visualizar la totalidad de profesores existentes con sus respectivos datos y acceder a las demás ventanas del mantenedor de profesores como es “Agregar Profesores”.
- **Diagrama de Bloques:**

- **Reglas del Proceso:**
  - Una vez el usuario ha ingresado a “Gestionar Profesores” desde el menú principal, se carga en la aplicación la pantalla de “Gestionar Profesores” (Figura 3-10).
  - En esta pantalla se listan la totalidad de profesores existentes con sus respectivos datos.
  - Cuenta con un cuadro de búsqueda en la parte superior izquierda de la pantalla que filtra en tiempo real los profesores, buscando en cada uno de los campos de la tabla de la base de datos de cada profesor según el texto que haya ingresado el usuario (Figura 3-11).
  - En la parte inferior de la pantalla se encuentran los botones con los que se controla el mantenedor de profesores, los botones son “Modificar”, “Eliminar”, “Ver/Editar Disponibilidad” solo se activan cuando el usuario ha seleccionado un profesor en la grilla.
  - Si el usuario lo desea puede eliminar un profesor presionando el botón “Eliminar” lo que enviará un mensaje de alerta (Figura 3-12 y Figura 3-13) para que el usuario tenga que confirmar dicha acción.

- **Diseños de Pantalla:**

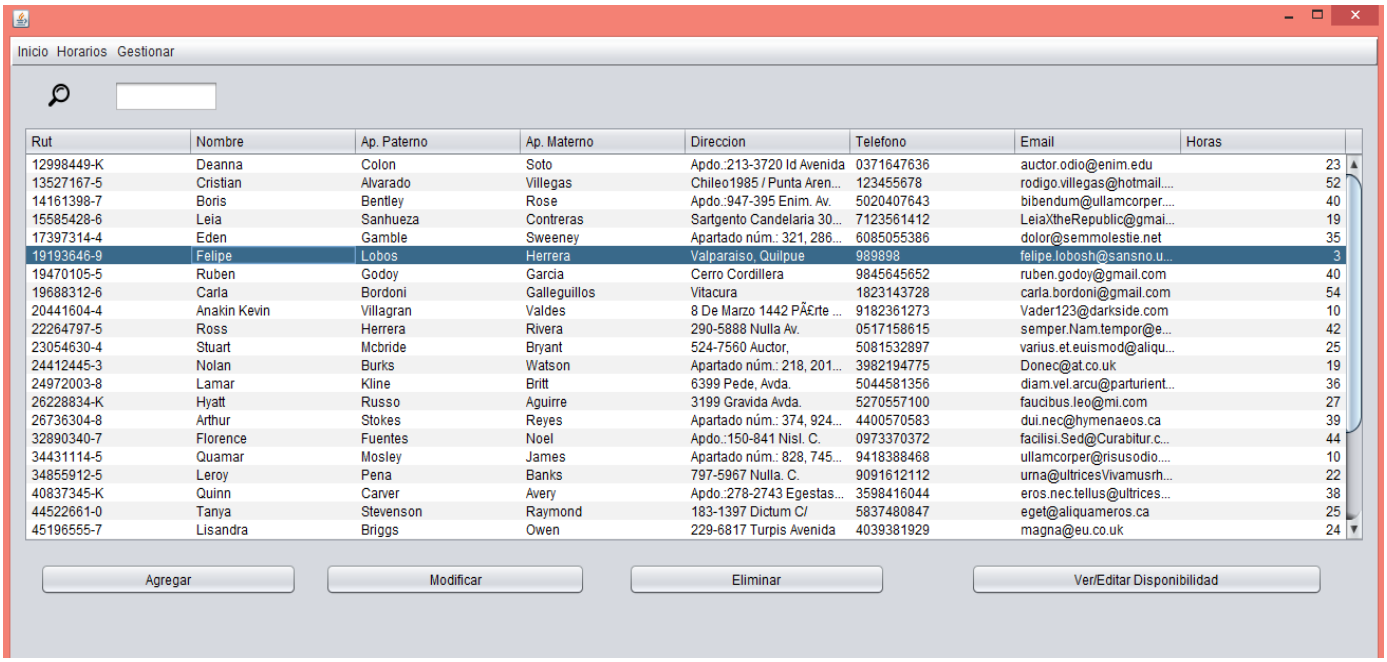


Figura 3-10: Pantalla de "Gestionar Profesores".

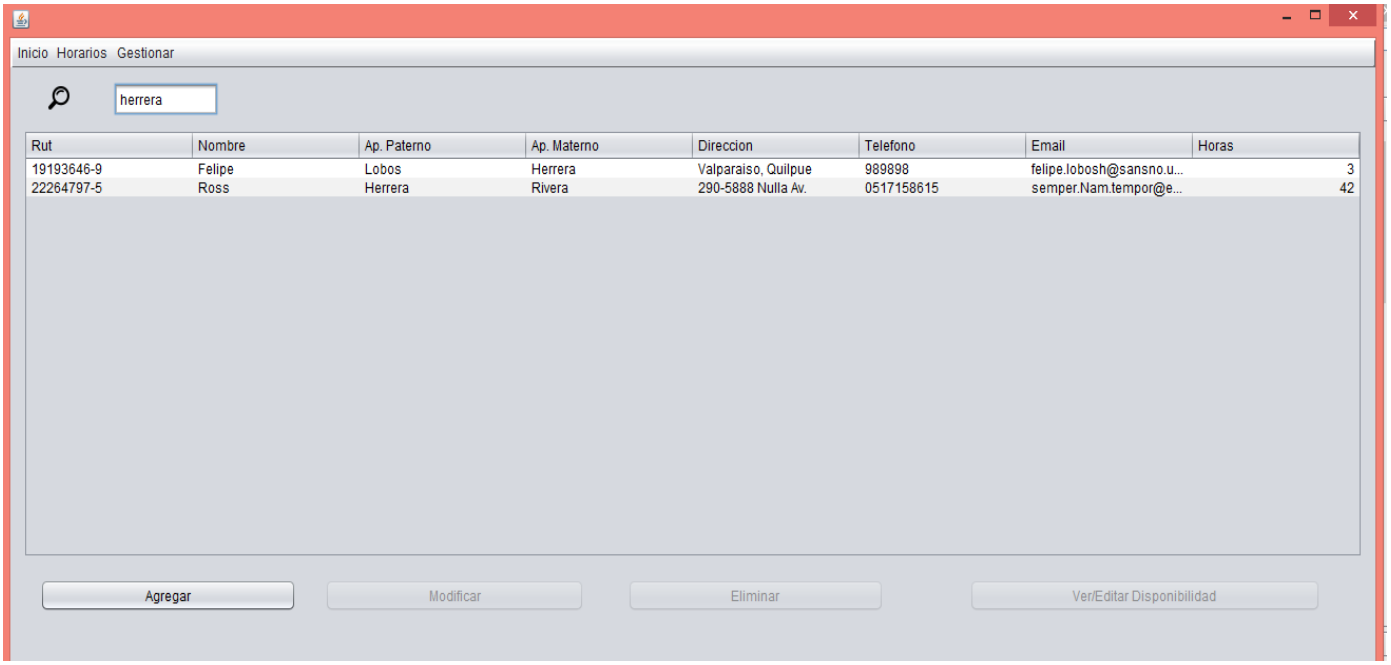


Figura 3-11: Pantalla de "Gestionar Profesores" (Filtrado).

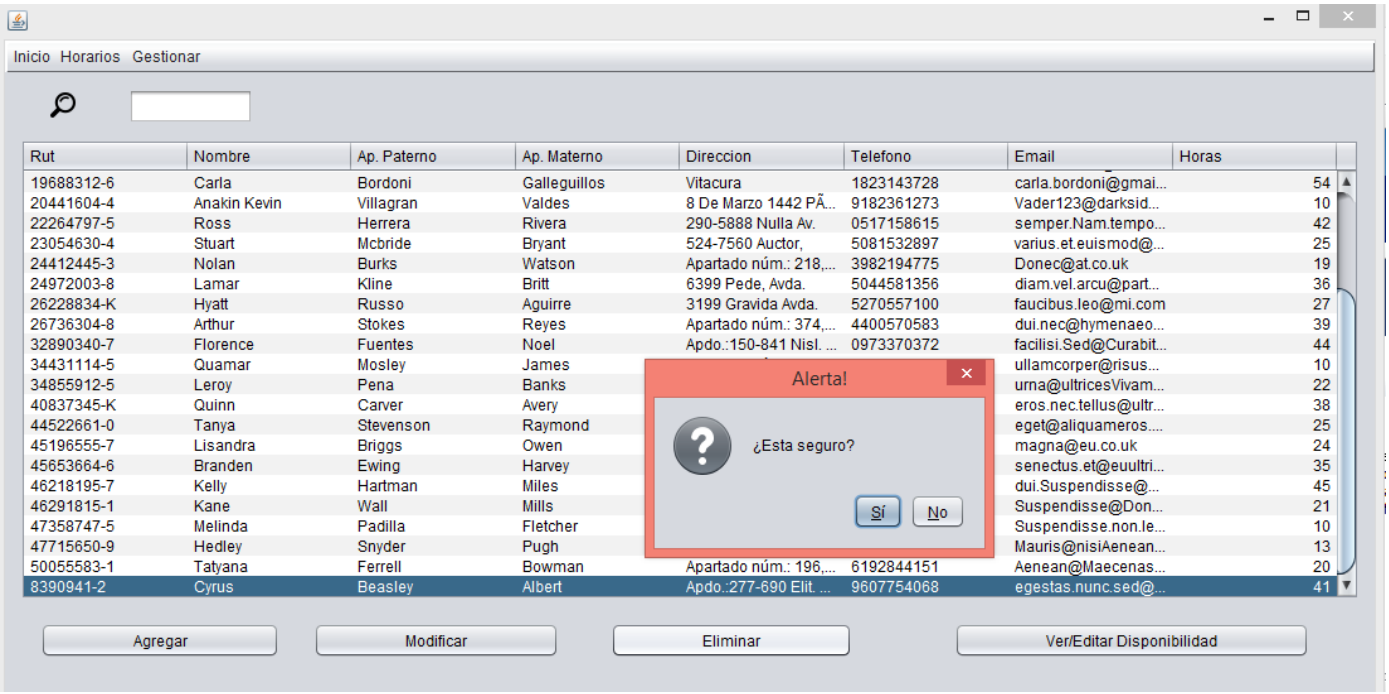


Figura 3-12: Pantalla de "Gestionar Profesores" (alerta de eliminacion).

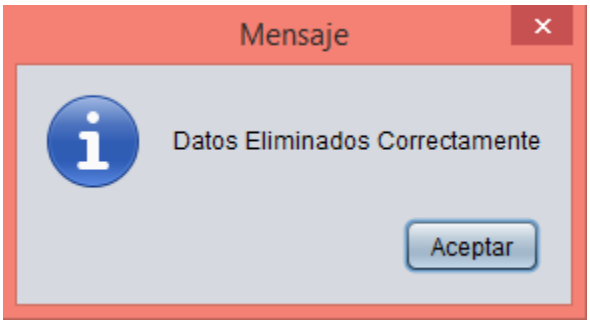


Figura 3-13: Pantalla de "Gestionar Profesores" (alerta de datos eliminados).

- **Referencias: Anexos (pag. 90 – pag. 100)**

3.4.5 Agregar Profesor.

- **Nombre:** Agregar Profesor.
- **Objetivo:** Permite al usuario ingresar un nuevo profesor al sistema, otorgando todas las facilidades y validaciones correspondientes para que no haya inconsistencia
- **Diagrama de Bloques:**

- **Reglas del Proceso:**
  - Se ingresa a través de la pantalla de "Gestionar Profesores" presionando el botón "Agregar" de la parte inferior, lo cual desplegara una nueva ventana emergente (Figura 3-14).
  - Una vez desplegada la nueva ventana de "Agregar Profesor", el usuario deberá ingresar manualmente por teclado cada uno de los campos que se exige para el ingreso de un nuevo profesor.
  - La aplicación valida el ingreso correcto del rut de profesor, así como también del email y el teléfono, remarcando los cuadros de texto con rojo, en caso de encontrarse un error o con verde, en caso de que el dato esté correcto (Figura 3-14).
  - Presionando el botón "Guardar" de la ventana de "Agregar profesor" en la parte inferior y en caso de que no falten campos por rellenar, ni se presenten errores de validación en los campos de rut, email o teléfono, la aplicación procede a guardar en la base de datos el nuevo profesor.
  - Si el usuario presiona el botón "Volver" de la parte inferior de la ventana, esta se cerrará volviendo a la pantalla de "Gestionar Profesores". Cualquier cambio que no se haya guardado es descartado.

- **Diseños de Pantalla:**

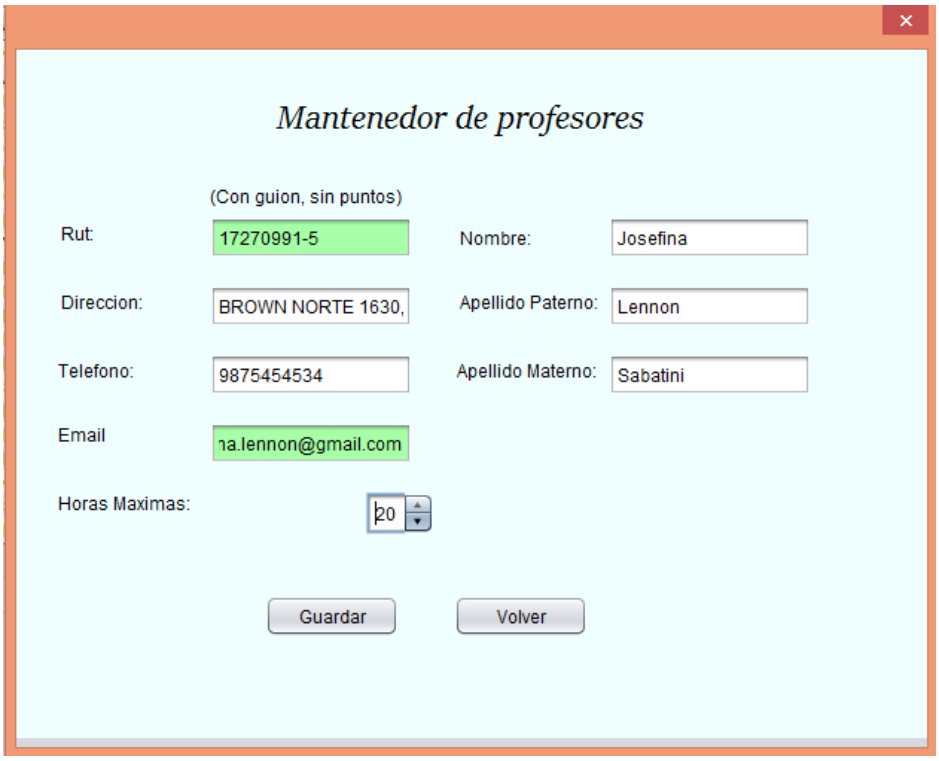


Figura 3-14: Pantalla de "Agregar Profesor".

- **Referencias: Anexos (pag. 100 – pag. 118)**

3.4.6 Modificar Profesor.

- **Nombre:** Modificar Profesor.
- **Objetivo:** Permite al usuario modificar los datos almacenados en la base de datos de un profesor que haya sido seleccionado en la pantalla de "Gestionar Profesores"
- **Diagrama de Bloques:**

- **Reglas del Proceso:**
    - Se accede a través del botón “Modificar” de la pantalla de “Gestionar Profesores”, lo que despliega una nueva ventana idéntica a la de “Agregar Profesor”, pero con los datos pre cargados por defecto del profesor seleccionado en la pantalla de “Gestionar Profesores” (Figura 3-15).
    - Todos los campos son modificables a excepción del rut.
    - “Modificar Profesor” cumple con las mismas validaciones de teléfono y email que “Agregar Profesor”.
    - Una vez que el usuario haya realizado todos los cambios que desee, debe presionar el botón “Guardar Cambios”, lo que guardara en la base de datos todos los cambios realizados, y enviará un mensaje al usuario notificando la accion.
    - Si el usuario presiona el botón “Volver” de la parte inferior de la ventana, esta se cerrará volviendo a la pantalla de “Gestionar Profesores”. Cualquier cambio que no se haya guardado es descartado.
- **Diseños de Pantalla:**

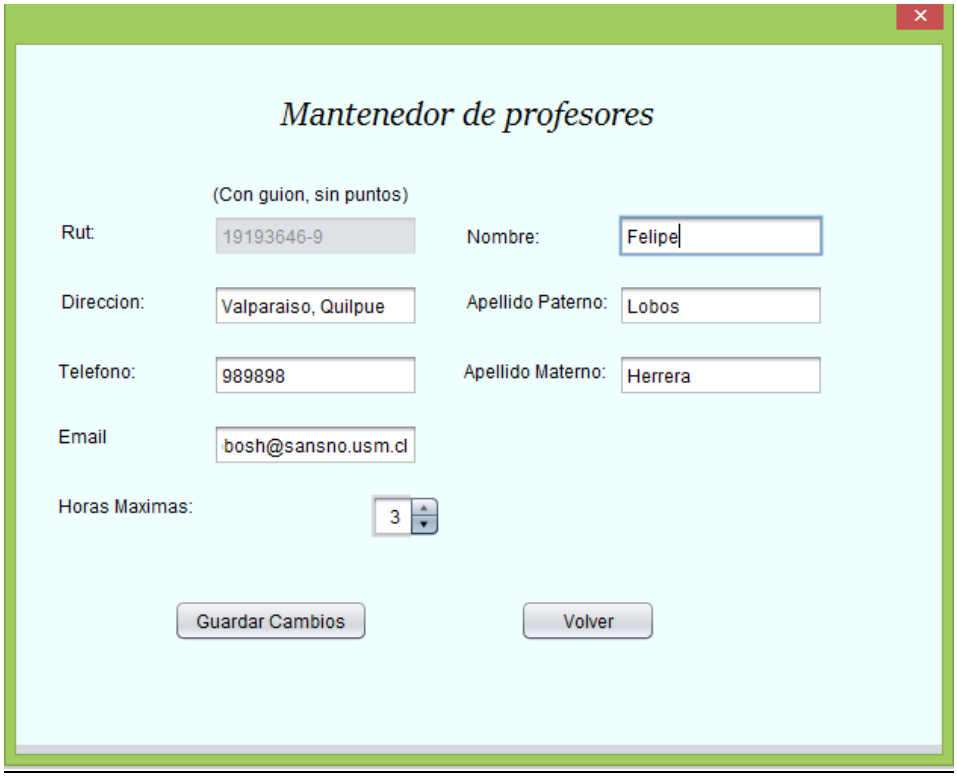
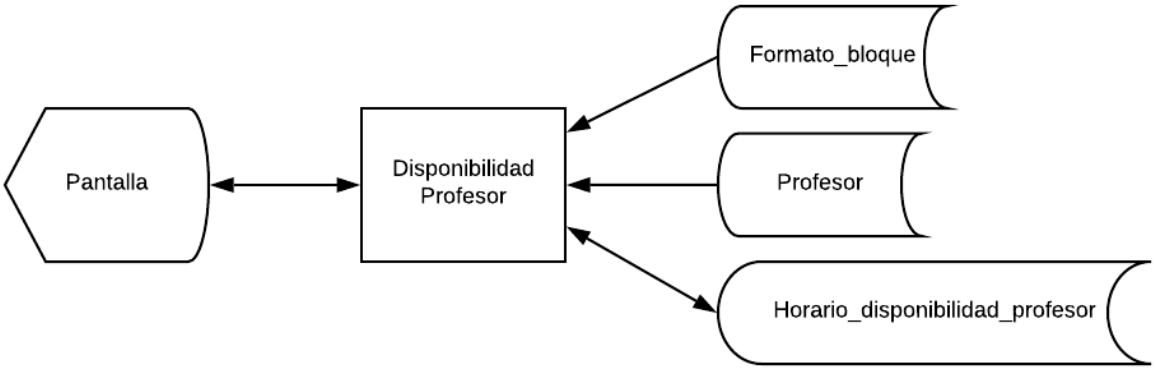


Figura 3-15: Ventana “Modificar Profesor”

- **Referencias: Anexos (pag. 100– pag. 118)**

3.4.7 Disponibilidad Profesor.

- **Nombre:** Disponibilidad Profesor.
- **Objetivo:** Permitir al usuario visualizar y editar la disponibilidad con que cada profesor cuenta para realizar actividades en la institución, mostrándolo a través de una tabla con bloques de horarios.
- **Diagrama de Bloques:**



- **Reglas del Proceso:**
  - Se accede a través de la pantalla de "Gestionar Profesores" presionando el botón de "Ver/Editar Disponibilidad" habiendo previamente seleccionado el profesor el cual desea ver su disponibilidad.
  - Se despliega una nueva ventana en la que se aprecia una tabla de horario en la que las filas son los bloques de horario y las columnas los días de la semana (Figura 3-16).
  - Se marca con un "Disponible" en cada celda donde el profesor tenga disponible para realizar actividades.
  - El usuario puede hacer click en una celda de la tabla para cambiar entre "Disponible" y vacío (lo que representa que el profesor no tiene disponible en dicho horario).
  - Una vez el usuario estime conveniente, puede guardar los cambios realizados presionando el botón "Guardar", lo que guarda cualquier cambio realizado en la base de datos.

- El usuario puede presionar el botón “Volver” lo que lo devuelve a la pantalla de “Gestionar Profesores” y descarta cualquier cambio que no haya sido guardado.

- **Diseños de Pantalla:**

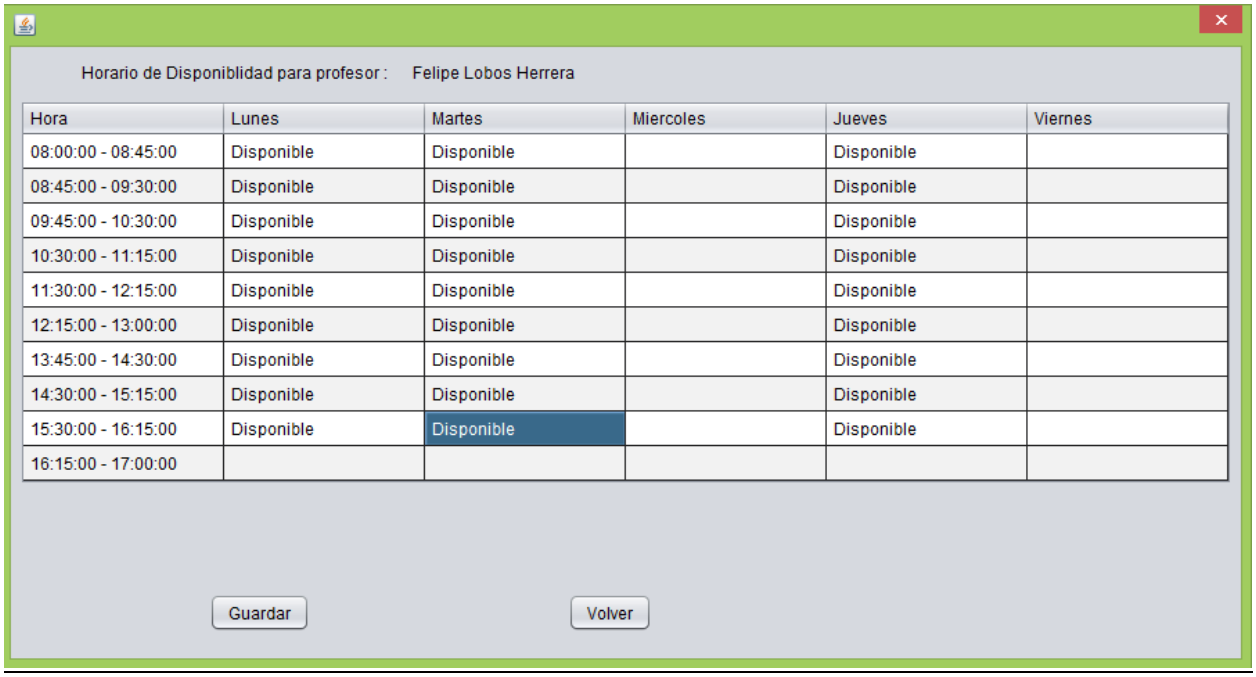
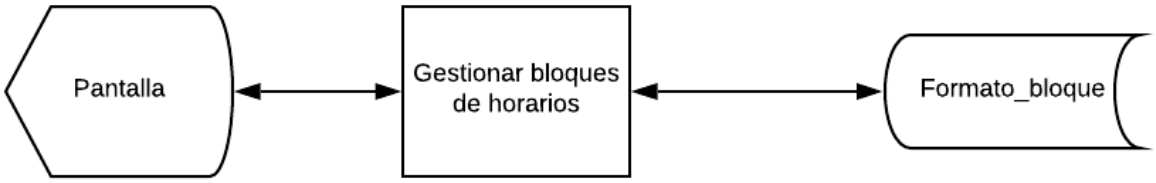


Figura 3-16 Pantalla de “Disponibilidad Profesor”.

- **Referencias: Anexos (pag. 118– pag. 125)**

3.4.8 Gestionar Bloques de Horario

- **Nombre:** Gestionar Bloques Horario.
- **Objetivo:** Visualizar y editar los bloques de horarios que manejan todas las tablas de horarios del sistema.
- **Diagrama de Bloques:**



- **Reglas del Proceso:**
    - Ingresando a través de la sección “Gestionar” del menú principal y eligiendo la opción “Bloques de Horario”, se desplegará por pantalla una tabla donde se pueden visualizar todos los bloques de horarios que constan de un número de bloque, una hora de inicio y una hora de término (Figura3-17).
    - En la parte superior derecha de la pantalla se encuentran dos selectores de hora, lo cuales se utilizan para modificar la hora de inicio (selector de arriba) y la hora de término (selector de abajo).
    - El usuario puede ingresar un nuevo bloque de horario ajustando los selectores de tiempo y pulsando el botón “Agregar”. El programa se encarga de validar que el nuevo bloque de horario no tenga ningún error en temas de las horas de inicio y término (la hora de inicio del nuevo bloque no puede ser inferior a la hora de término del bloque anterior, la hora de término del nuevo bloque no puede ser inferior a la hora de inicio del mismo bloque).
    - El usuario también puede editar algún bloque de horario existente seleccionando uno de la tabla (los selectores de hora se ajustarán automáticamente a los datos del bloque seleccionado), ajustando los selectores de hora y presionando el botón “Editar”.
    - Así mismo el usuario puede eliminar un bloque de horario, seleccionando uno de la tabla y presionando el botón “Eliminar” lo que enviará un mensaje de alerta que exigirá una confirmación por parte del usuario.
    - Cuando el usuario desee guardar todos los cambios realizados, puede presionar el botón “Guardar Cambios”, lo que guardará todo cambio realizado.

- **Diseños de Pantalla:**

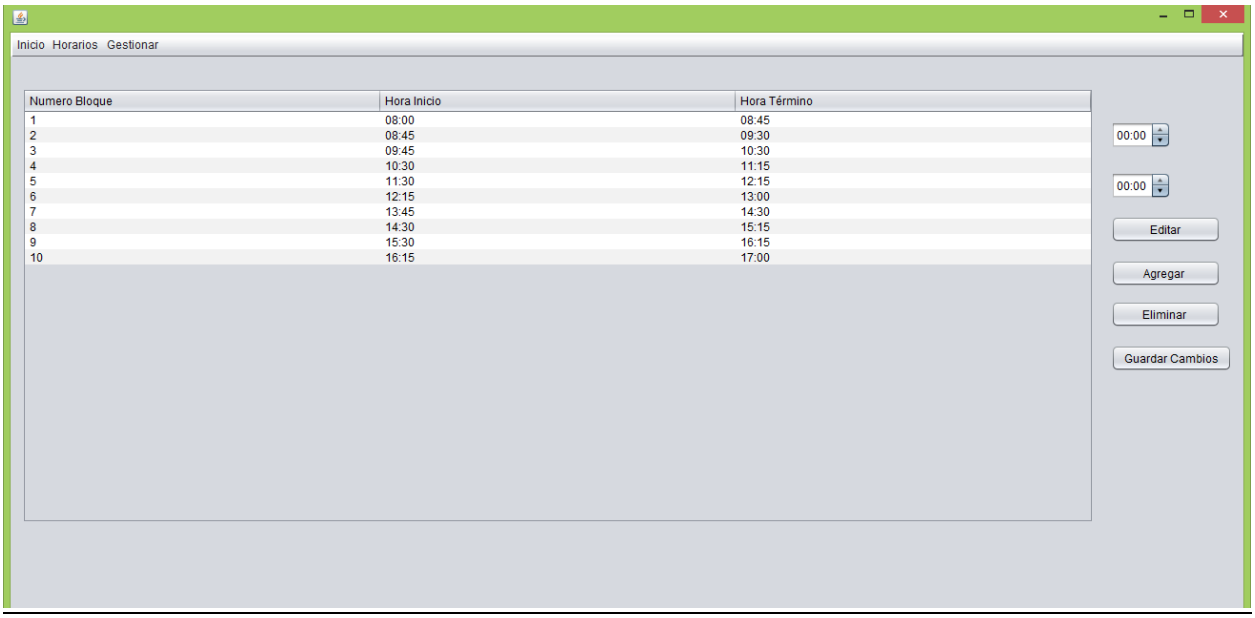
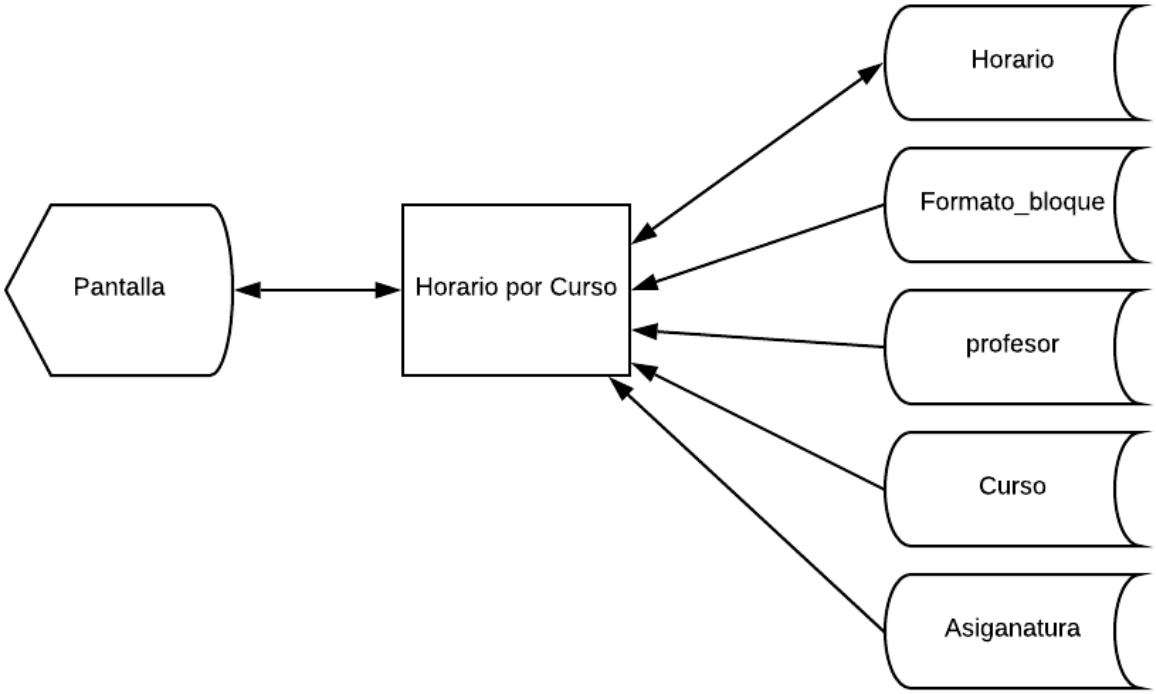


Figura 3-17 Pantalla de "Gestionar Bloques Horario".

- **Referencias: Anexos (pag. 125– pag. 136)**

3.4.9 Horario por Curso

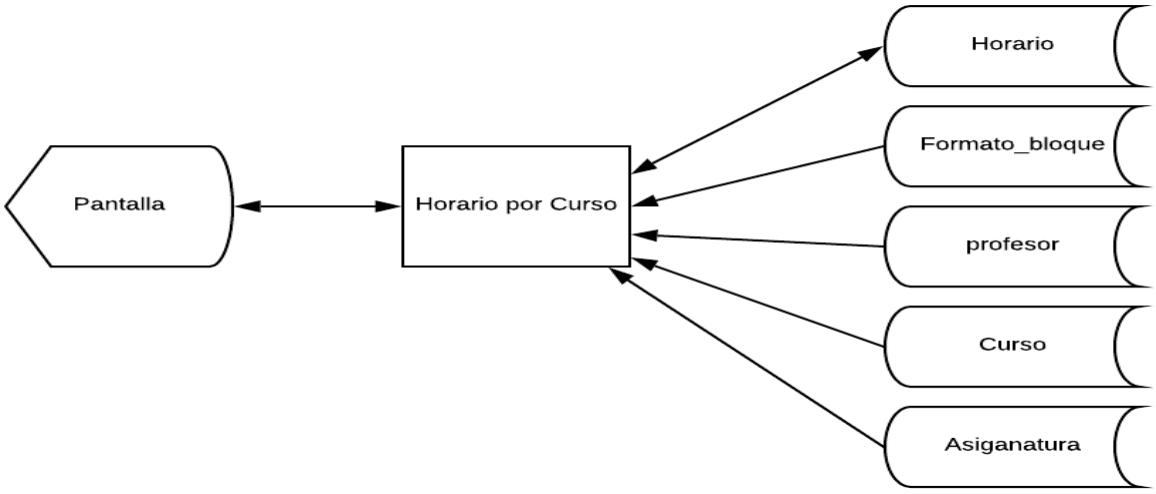
- **Nombre:** Horario por Curso.
- **Objetivo:** Mostrar de forma clara y sencilla el horario de cada curso registrado en la base de datos, además de permitir editar dicho horario de una manera intuitiva.
- **Diagrama de Bloques:**



- **Reglas del Procesos:**
  - Una vez el usuario haya ingresado en la sección (Horarios –> Horario Por Curso), se desplegará la pantalla de horario por curso (Figura 3-17).
  - El usuario puede seleccionar el curso por el cual desea consultar el horario en la parte superior de la pantalla (Figura 3-19), al momento de cambiar de curso seleccionado, se cargarán desde la base de datos todos los bloques de horarios (Tabla Horario) en los cuales aparece.
  - El usuario puede seleccionar una casilla de la tabla para editarla si gusta. Presionando el botón “Borrar” de la parte inferior derecha, el programa eliminará los datos de esa casilla volviéndola a dejar libre para ese curso.

- De la misma manera, el usuario puede insertar un nuevo horario ocupado para el curso presionando el botón “Insertar”. Al momento de presionarlo (y habiendo seleccionado previamente una casilla libre, de lo contrario, se enviará un mensaje de error, el programa también verificará que el curso no exceda su cantidad de horas máximas para esa asignatura o profesor con la nueva hora seleccionada) el programa tomará los datos de Asignatura, Profesor y Sala que el usuario haya seleccionado y los ingresará en la casilla al mismo tiempo que se almacena ese nuevo dato en la base de datos (tabla Horario).
- En caso de generarse algún error, como tope de horario entre cursos o profesores, se le notificará al usuario con un mensaje en la parte inferior de la tabla y la operación no se llevará a cabo (Figura 3-18, se genera un error al intentar asignar una nuevo bloque de hora a un profesor que ya alcanzó su limite de horas).

- **Diagrama de bloques:**



- **Diseño de Pantalla:**

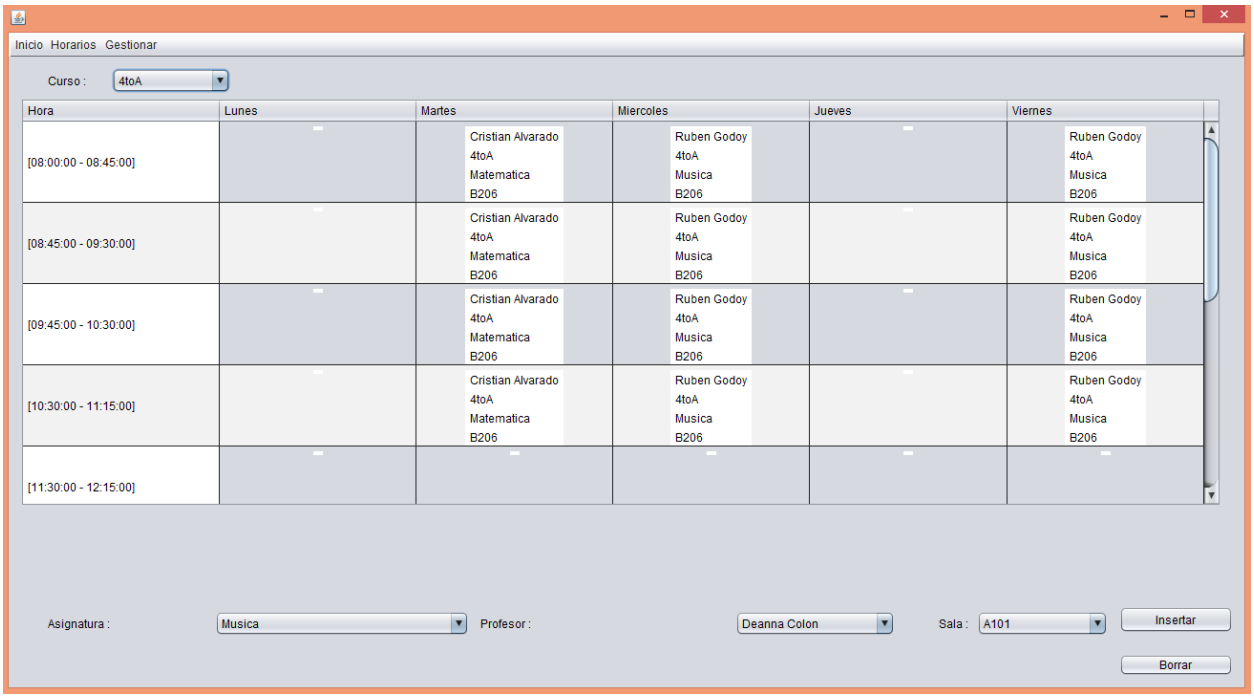


Figura 3-17: Pantalla "Horario por Cursos".

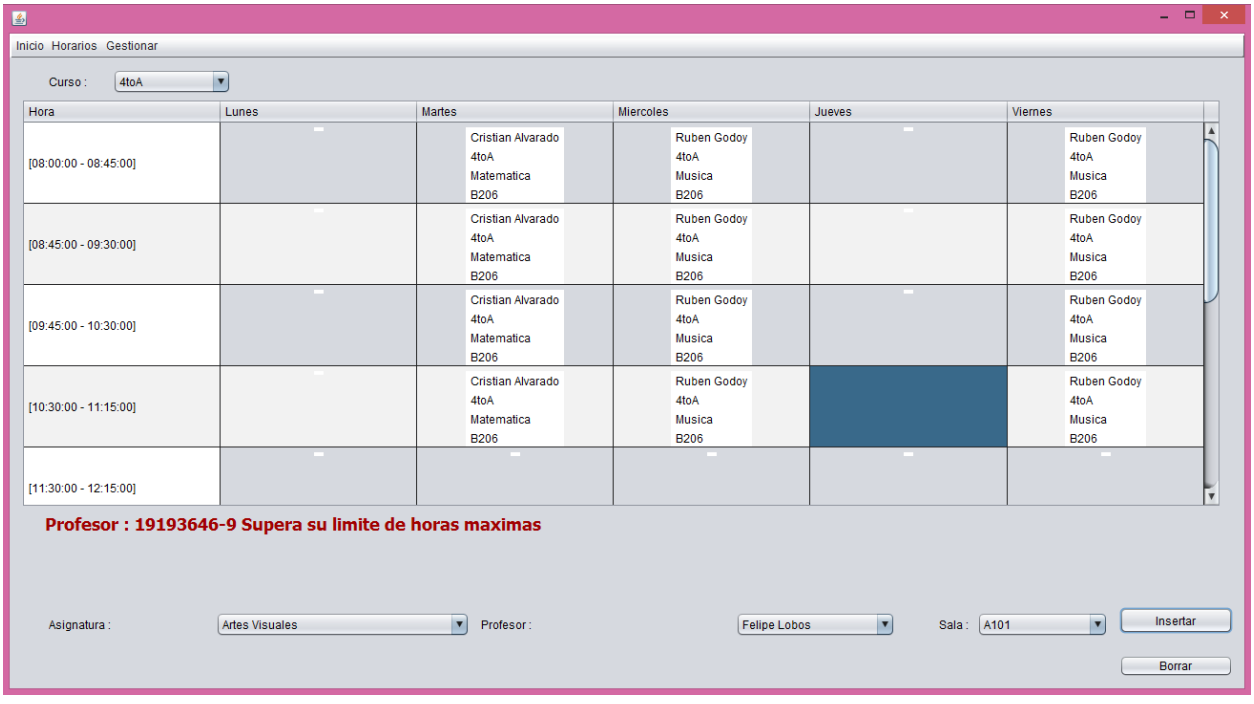


Figura 3-18: Pantalla de "Horario por Curso" –Error de inserción.

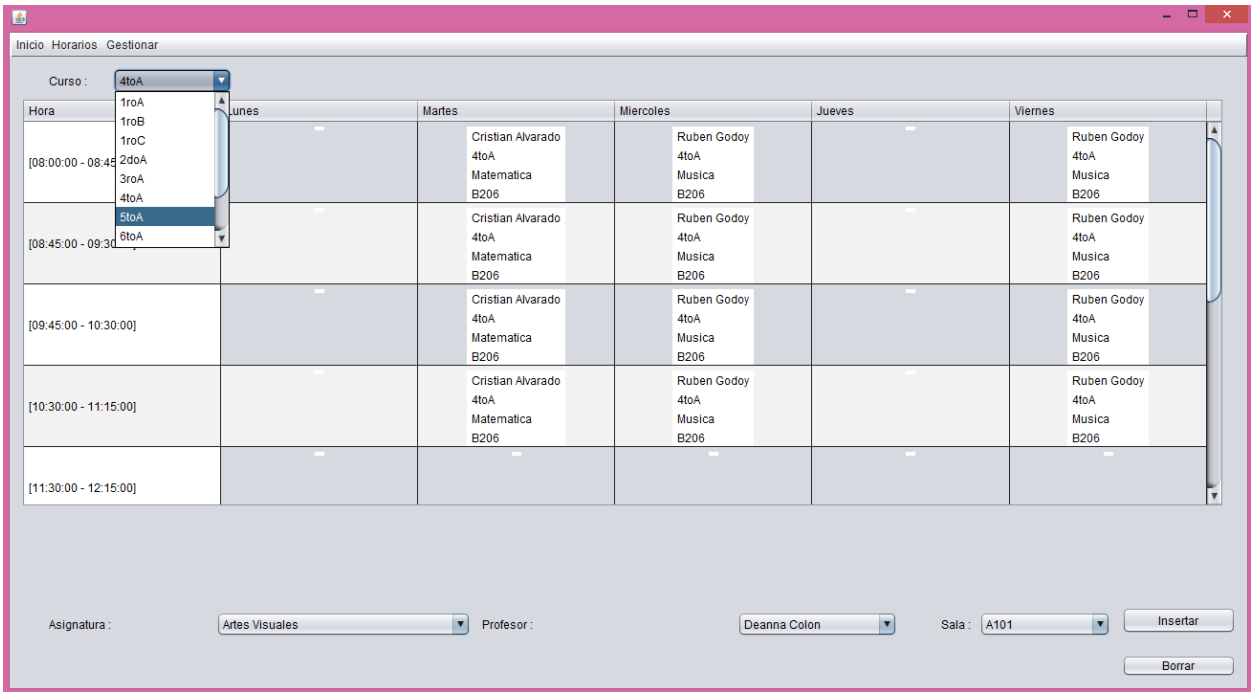


Figura 3-19: Pantalla de "Horario por Curso" (Seleccionar un curso).

- **Reaferencias: Anexos (pag. 136– pag. 149).**

CONCLUSIONES

Este proyecto nace bajo la premisa de ayudar a la institución escolar donde uno de los integrantes del equipo cursó su enseñanza básica, nos contactamos con la escuela y está nos comunicó su mayor problema, la creación de horarios, una tarea importante la cual tiene muchos puntos que tomar en cuenta y que les tomaba meses en completar.

Aceptamos el desafío de desarrollar un sistema que facilitara dicha labor y la primera decisión fue de que lenguaje utilizar, ya que la institución previamente nos había pedido que fuese una aplicación de escritorio, nos inclinamos por usar Java con NetBeans IDE como entorno de desarrollo, ya que es era algo en lo que ya habíamos trabajado durante nuestro periodo universitario por lo cual teníamos un mayor conocimiento en dicho lenguaje.

La fase de análisis del proyecto nos tomó unos tres meses en completar y la fase de desarrollo otros tres meses. El análisis y diseño fue nuestro punto más débil y se notó, debido a nuestra poca experiencia en el tema hubo que hacer correcciones fuertes durante la fase de desarrollo, puntos que no consideramos y otros que sobraban, lo que nos retrasó a la hora de programar. sumado a las limitaciones del lenguaje y el entorno, limitaciones que no conocíamos a la hora de elegir en qué trabajar. Por temas de tiempo hubo que desechar ideas, sin embargo, se logró cumplir con el propósito principal del proyecto.

En proyecciones a futuro sin duda deberíamos mejorar el diseño de la aplicación, lograr que sea más vistoso y cómodo para el usuario, así como también la seguridad del sistema.

A la fecha en la cual está siendo redactado este informe, la aplicación no está implementada aún, debido a una falta de coordinación entre los tiempos de la institución y nuestros tiempos personales, por lo que no se ha podido llegar a la fase de implementación.

ANEXOS

Inicio Sesión

```
import java.io.IOException;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Arrays;
import java.util.logging.Level;
import java.util.logging.Logger;
import javafx.application.Application;
import javax.swing.JOptionPane;

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
/**
 *
 * @author felipe
 */
public class Login extends javax.swing.JFrame {

    /**
     * Creates new form Login
     */
    Conectar conexion = new Conectar();

    public Login() {
        initComponents();
        this.setLocationRelativeTo(null);

        this.getRootPane().setDefaultButton(btnAceptar);
        //btnAceptar.requestFocus();
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {
```

```
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        txtUsuario = new javax.swing.JTextField();
        btnAceptar = new javax.swing.JButton();
        btnSalir = new javax.swing.JButton();
        txtContraseña = new javax.swing.JPasswordField();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("INICIO DE SESIÓN");
        setLocation(new java.awt.Point(0, 0));
        setResizable(false);

        jLabel1.setText("Usuario");

        jLabel2.setText("Contraseña");

        txtUsuario.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                txtUsuarioKeyTyped(evt);
            }
        });

        btnAceptar.setText("Aceptar");
        btnAceptar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnAceptarActionPerformed(evt);
            }
        });

        btnSalir.setText("Salir");
        btnSalir.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnSalirActionPerformed(evt);
            }
        });

        txtContraseña.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                txtContraseñaKeyTyped(evt);
            }
        });

        javax.swing.GroupLayout layout = new
    javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
```

```
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(41, 41, 41)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(jLabel2)
                    .addComponent(jLabel1)
                    .addComponent(btnSalir))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING,
false)
                    .addComponent(btnAceptar)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                        .addComponent(txtUsuario,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.PREFERRED_SIZE, 131,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(txtContraseña,
javax.swing.GroupLayout.PREFERRED_SIZE, 131,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap(42, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(89, 89, 89)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE
)
                    .addComponent(jLabel1)
                    .addComponent(txtUsuario, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(34, 34, 34)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE
)
                    .addComponent(jLabel2)
                    .addComponent(txtContraseña,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
26, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE
)
                .addComponent(btnAceptar)
                .addComponent(btnSalir))
            .addGap(51, 51, 51))
      );

      pack();
   }// </editor-fold>

   private void btnAceptarActionPerformed(java.awt.event.ActionEvent evt) {
      // TODO add your handling code here:
      String sql = "SELECT * FROM usuarios where nomUsuario = '" +
txtUsuario.getText() + "'";

      ResultSet rs = null;
      char[] pass = txtContraseña.getPassword();
      String pass2 = pass.toString();
      String passRs = "";
      String userRs = "";
      rs = conexion.ConsultarUsuario(sql);
      try {
         rs.next();
         passRs = rs.getString("contraseña");
         userRs = rs.getString("nomUsuario");

      } catch (SQLException ex) {
         //JOptionPane.showMessageDialog(null,"Usuario Invalido");
         //Logger.getLogger(Login.class.getName()).log(Level.SEVERE, null, ex);
      }
      if (!txtUsuario.getText().equals("")) {
         if (userRs.equals(txtUsuario.getText())) {
            if (Arrays.equals(pass, passRs.toCharArray())) {
               Menu menu = null;
               try {
                  try {
                     menu = new Menu();
                  } catch (IOException ex) {
                     Logger.getLogger(Login.class.getName()).log(Level.SEVERE, null,
ex);
                  }
               } catch (SQLException ex) {
                  Logger.getLogger(Login.class.getName()).log(Level.SEVERE, null, ex);
```

```
                }
                menu.setVisible(true);
                this.setVisible(false);
            } else {

                JOptionPane.showMessageDialog(null, "Contraseña Incorrecta", "ERROR",
JOptionPane.WARNING_MESSAGE);
            }

        } else {
            JOptionPane.showMessageDialog(null, "Usuario Invalido", "ERROR",
JOptionPane.WARNING_MESSAGE);
        }
    }


}

private void btnSalirActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    //this.dispose();
    System.exit(0);
}

private void txtUsuarioKeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    int Chr = evt.getKeyChar();
    if (Chr == 39) {
        evt.consume();
    }
}

private void txtContraseñaKeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    int Chr = evt.getKeyChar();
    if (Chr == 39) {
        evt.consume();
    }
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
```

```
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Login.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Login().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnAceptar;
    private javax.swing.JButton btnSalir;
    private javax.swing.JLabel jLabel1;
```

```
    private javax.swing.JLabel jLabel2;
    private javax.swing.JPasswordField txtContraseña;
    private javax.swing.JTextField txtUsuario;
    // End of variables declaration
}
```

Menu

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

import java.awt.BorderLayout;
import java.awt.Dimension;
import java.awt.Frame;
import java.awt.Image;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.imageio.ImageIO;
import javax.swing.ImageIcon;
import javax.swing.JLabel;

/**
 *
 * @author labinf10
 */
public class Menu extends javax.swing.JFrame {

    /**
     * Creates new form Menu
     */
    public Menu() throws SQLException, IOException {
        this.panel = new JPListaProfesores(this);
        initComponents();
        this.setLocationRelativeTo(null);
        this.setState(Frame.NORMAL);

    }
```

```
    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jMenuItem3 = new javax.swing.JMenuItem();
        jMenuItem4 = new javax.swing.JMenuItem();
        jPopupMenu1 = new javax.swing.JPopupMenu();
        jPanel1 = new javax.swing.JPanel();
        jLabel2 = new javax.swing.JLabel();
        jLabel1 = new javax.swing.JLabel();
        jMenuBar1 = new javax.swing.JMenuBar();
        jMenu8 = new javax.swing.JMenu();
        jMenuItem28 = new javax.swing.JMenuItem();
        jMenu3 = new javax.swing.JMenu();
        jMenuItem1 = new javax.swing.JMenuItem();
        jMenuItem2 = new javax.swing.JMenuItem();
        jMenuItem5 = new javax.swing.JMenuItem();
        jMenuItem6 = new javax.swing.JMenuItem();
        jMenuGestionar = new javax.swing.JMenu();
        itemProfe = new javax.swing.JMenuItem();
        itemAsignaturas = new javax.swing.JMenuItem();
        itemCursos = new javax.swing.JMenuItem();
        itemSalas = new javax.swing.JMenuItem();
        itemImplementos = new javax.swing.JMenuItem();
        itemBloque = new javax.swing.JMenuItem();
        itemUsuarios = new javax.swing.JMenuItem();

        jMenuItem3.setText("jMenuItem3");

        jMenuItem4.setText("jMenuItem4");

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setMinimumSize(new java.awt.Dimension(1070, 700));

        jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/Imagenes/logotipos2.png"))); //
NOI18N

        javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
```

```
jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGap(295, 295, 295)
            .addComponent(jLabel1)
            .addGap(146, 146, 146)
            .addComponent(jLabel2)
            .addContainerGap(216, Short.MAX_VALUE))
    );
    jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGap(105, 105, 105)
                    .addComponent(jLabel2))
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addContainerGap()
                    .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE,
611, javax.swing.GroupLayout.PREFERRED_SIZE)))
            .addContainerGap(31, Short.MAX_VALUE))
    );

    jMenuBar1.setBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAISED));

    jMenu8.setText("Inicio");

    jMenuItem28.setText("Cerrar Sesion");
    jMenuItem28.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jMenuItem28ActionPerformed(evt);
        }
    });
    jMenu8.add(jMenuItem28);

    jMenuBar1.add(jMenu8);

    jMenu3.setText("Horarios");

    jMenuItem1.setText("Horario Por Profesores");
    jMenuItem1.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
        jMenuItem1ActionPerformed(evt);
    }
});
jMenu3.add(jMenuItem1);

jMenuItem2.setText("Horario por Asignaturas");
jMenuItem2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jMenuItem2ActionPerformed(evt);
    }
});
jMenu3.add(jMenuItem2);

jMenuItem5.setText("Horario Por Crusos");
jMenuItem5.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jMenuItem5ActionPerformed(evt);
    }
});
jMenu3.add(jMenuItem5);

jMenuItem6.setText("Horario Global");
jMenuItem6.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jMenuItem6ActionPerformed(evt);
    }
});
jMenu3.add(jMenuItem6);

jMenuBar1.add(jMenu3);

jMenuGestionar.setText("Gestionar");

itemProfe.setText("Profesores");
itemProfe.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        itemProfeActionPerformed(evt);
    }
});
jMenuGestionar.add(itemProfe);

itemAsignaturas.setText("Asignaturas");
itemAsignaturas.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        itemAsignaturasActionPerformed(evt);
    }
```

```
});
jMenuGestionar.add(itemAsignaturas);

itemCursos.setText("Cursos");
itemCursos.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        itemCursosActionPerformed(evt);
    }
});
jMenuGestionar.add(itemCursos);

itemSalas.setText("Salas");
itemSalas.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        itemSalasActionPerformed(evt);
    }
});
jMenuGestionar.add(itemSalas);

itemImplementos.setText("Implementos");
itemImplementos.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        itemImplementosActionPerformed(evt);
    }
});
jMenuGestionar.add(itemImplementos);

itemBloque.setText("Bloques de Horario");
itemBloque.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        itemBloqueActionPerformed(evt);
    }
});
jMenuGestionar.add(itemBloque);

itemUsuarios.setText("Usuarios");
itemUsuarios.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        itemUsuariosActionPerformed(evt);
    }
});
jMenuGestionar.add(itemUsuarios);

jMenuBar1.add(jMenuGestionar);

setJMenuBar(jMenuBar1);
```

```
        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, 1282,
Short.MAX_VALUE)
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addContainerGap())
        );

        pack();
    }// </editor-fold>
    //mantenedores
    JPListaProfesores panel;
    JPCursos cursos = new JPCursos();
    JPAsig asignaturas = new JPAsig();
    JPImplementos implementos = new JPImplementos();
    JPSala salas = new JPSala();
    JPUsuarios usuaarios = new JPUsuarios();

    protected void actualizarP(String sql) throws SQLException {
        panel.F5();
    }
    private void itemProfeActionPerformed(java.awt.event.ActionEvent evt) {
        String sql = "SELECT * FROM profesores";
        //panel.mostrar(sql);

        Dimension tamaño = this.jPanel1.getSize();
        BorderLayout borde = new BorderLayout();

        panel.setSize(panel.getPreferredSize());

        this.jPanel1.removeAll();
        this.jPanel1.revalidate();
        this.revalidate();
```

```
    this.jPanel1.setLayout(borde);

    this.jPanel1.add(panel, BorderLayout.CENTER);
    this.jPanel1.setSize(tamaño);
    this.jPanel1.revalidate();
    this.jPanel1.repaint();



}

private void itemCursosActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Dimension tamaño = this.getSize();
    BorderLayout borde = new BorderLayout();

    cursos.setSize(cursos.getPreferredSize());

    this.jPanel1.removeAll();
    this.jPanel1.revalidate();
    this.revalidate();

    this.jPanel1.setLayout(borde);

    this.jPanel1.add(cursos, BorderLayout.CENTER);
    this.jPanel1.setSize(tamaño);
    this.jPanel1.revalidate();
    this.jPanel1.repaint();
}

private void itemAsignaturasActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    //asignaturas.mostrar(sql);
    Dimension tamaño = this.jPanel1.getSize();
    BorderLayout borde = new BorderLayout();

    asignaturas.setSize(asignaturas.getPreferredSize());

    this.jPanel1.removeAll();
    this.jPanel1.revalidate();
    this.revalidate();

    this.jPanel1.setLayout(borde);
```

```
    this.jPanel1.add(asignaturas, BorderLayout.CENTER);
    this.jPanel1.setSize(tamaño);
    this.jPanel1.revalidate();
    this.jPanel1.repaint();

}

private void itemImplementosActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Dimension tamaño = this.getSize();
    BorderLayout borde = new BorderLayout();

    implementos.setSize(implementos.getPreferredSize());

    this.jPanel1.removeAll();
    this.jPanel1.revalidate();
    this.revalidate();

    this.jPanel1.setLayout(borde);

    this.jPanel1.add(implementos, BorderLayout.CENTER);
    this.jPanel1.setSize(tamaño);
    this.jPanel1.revalidate();
    this.jPanel1.repaint();
}

private void itemSalasActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String sql = "SELECT * FROM salas";

    Dimension tamaño = this.getSize();
    BorderLayout borde = new BorderLayout();

    salas.setSize(salas.getPreferredSize());

    this.jPanel1.removeAll();
    this.jPanel1.revalidate();
    this.revalidate();

    this.jPanel1.setLayout(borde);

    this.jPanel1.add(salas, BorderLayout.CENTER);
    this.jPanel1.setSize(tamaño);
    this.jPanel1.revalidate();
    this.jPanel1.repaint();
}
```

```
private void itemUsuariosActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Dimension tamaño = this.getSize();
    BorderLayout borde = new BorderLayout();

    usuaarios.setSize(usuaarios.getPreferredSize());

    this.jPanel1.removeAll();
    this.jPanel1.revalidate();
    this.revalidate();

    this.jPanel1.setLayout(borde);

    this.jPanel1.add(usuaarios, BorderLayout.CENTER);
    this.jPanel1.setSize(tamaño);
    this.jPanel1.revalidate();
    this.jPanel1.repaint();
}

private void itemBloqueActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Dimension tamaño = this.getSize();
    BorderLayout borde = new BorderLayout();
    JPBloque bloque = new JPBloque();
    bloque.setSize(bloque.getPreferredSize());

    this.jPanel1.removeAll();
    this.jPanel1.revalidate();
    this.revalidate();

    this.jPanel1.setLayout(borde);

    this.jPanel1.add(bloque, BorderLayout.CENTER);
    this.jPanel1.setSize(tamaño);
    this.jPanel1.revalidate();
    this.jPanel1.repaint();
}

private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Dimension tamaño = this.getSize();
    BorderLayout borde = new BorderLayout();
    JPHorarioProfesores hProfesores = new JPHorarioProfesores();
    hProfesores.setSize(hProfesores.getPreferredSize());
```

```
    this.jPanel1.removeAll();
    this.jPanel1.revalidate();
    this.revalidate();

    this.jPanel1.setLayout(borde);

    this.jPanel1.add(hProfesores, BorderLayout.CENTER);
    this.jPanel1.setSize(tamaño);
    this.jPanel1.revalidate();
    this.jPanel1.repaint();
}

private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Dimension tamaño = this.getSize();
    BorderLayout borde = new BorderLayout();
    JPHorarioAsignaturas hAsignatras = new JPHorarioAsignaturas();
    hAsignatras.setSize(hAsignatras.getPreferredSize());

    this.jPanel1.removeAll();
    this.jPanel1.revalidate();
    this.revalidate();

    this.jPanel1.setLayout(borde);

    this.jPanel1.add(hAsignatras, BorderLayout.CENTER);
    this.jPanel1.setSize(tamaño);
    this.jPanel1.revalidate();
    this.jPanel1.repaint();


}

private void jMenuItem5ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    Dimension tamaño = this.getSize();
    BorderLayout borde = new BorderLayout();
    JPHorarioCursos hCursos = new JPHorarioCursos();
    hCursos.setSize(hCursos.getPreferredSize());

    this.jPanel1.removeAll();
    this.jPanel1.revalidate();
    this.revalidate();

    this.jPanel1.setLayout(borde);
```

```
        this.jPanel1.add(hCursos, BorderLayout.CENTER);
        this.jPanel1.setSize(tamaño);
        this.jPanel1.revalidate();
        this.jPanel1.repaint();
    }

    private void jMenuItem6ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            // TODO add your handling code here:
            Dimension tamaño = this.getSize();
            BorderLayout borde = new BorderLayout();

            JPHorarioG  prueba = new JPHorarioG();

            prueba.setSize(prueba.getPreferredSize());

            this.jPanel1.removeAll();
            this.jPanel1.revalidate();
            this.revalidate();

            this.jPanel1.setLayout(borde);

            this.jPanel1.add(prueba, BorderLayout.CENTER);
            this.jPanel1.setSize(tamaño);
            this.jPanel1.revalidate();
            this.jPanel1.repaint();
        } catch (SQLException ex) {
            Logger.getLogger(Menu.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    private void jMenuItem28ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        Login inicio = new Login();
        inicio.setVisible(true);
        this.dispose();
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
">
```

```
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(() -> {
            try {
                new Menu().setVisible(true);
            } catch (SQLException ex) {
                Logger.getLogger(Menu.class.getName()).log(Level.SEVERE, null, ex);
            } catch (IOException ex) {
                Logger.getLogger(Menu.class.getName()).log(Level.SEVERE, null, ex);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JMenuItem itemAsignaturas;
```

```
    private javax.swing.JMenuItem itemBloque;
    private javax.swing.JMenuItem itemCursos;
    private javax.swing.JMenuItem itemImplementos;
    private javax.swing.JMenuItem itemProfe;
    private javax.swing.JMenuItem itemSalas;
    private javax.swing.JMenuItem itemUsuarios;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JMenu jMenu3;
    private javax.swing.JMenu jMenu8;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JMenu jMenuGestionar;
    private javax.swing.JMenuItem jMenuItem1;
    private javax.swing.JMenuItem jMenuItem2;
    private javax.swing.JMenuItem jMenuItem28;
    private javax.swing.JMenuItem jMenuItem3;
    private javax.swing.JMenuItem jMenuItem4;
    private javax.swing.JMenuItem jMenuItem5;
    private javax.swing.JMenuItem jMenuItem6;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPopupMenu jPopupMenu1;
    // End of variables declaration
}
```

Horario por Profesores

```
import java.awt.Color;
import java.awt.TextArea;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.ParseException;
import java.util.Vector;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.DefaultComboBoxModel;
import javax.swing.DefaultListModel;
import javax.swing.JList;
import javax.swing.JTable;
import javax.swing.ListSelectionModel;
import javax.swing.table.DefaultTableModel;

/*
 * To change this license header, choose License Headers in Project Properties.
```

```
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
/**
 *
 * @author felipe
 */
public class JPHorarioProfesores extends javax.swing.JPanel {

    /**
     * Creates new form JPHorario
     */
    JList lista = new JList();
    JTable tabla = new JTable();
    String rutGlobal = "";

    Color rojo = new Color(161, 0, 0);
    Color verde = new Color(0, 161, 0);
    Color blanco = new Color(255, 255, 255);

    public JPHorarioProfesores() {
        initComponents();
        //MultiLineTableCellRenderer renderer = new MultiLineTableCellRenderer();

        //set TableCellRenderer into a specified JTable column class
        // tbHorario.setDefaultRenderer(String[].class, renderer);
        //or, set TableCellRenderer into a specified JTable column
        //tbHorario.getColumnModel().getColumn(1).setCellRenderer(renderer);
        //tbHorario.setDefaultRenderer(Object.class, new MultiLineCellRenderer());
        DefaultTableModel dtm = new DefaultTableModel() {
            @Override
            public boolean isCellEditable(int row, int column) {
                //all cells false
                return false;
            }
        };



        tbHorario.getColumn("Lunes").setCellRenderer(new JListRenderer());
        tbHorario.getColumn("Martes").setCellRenderer(new JListRenderer());
        tbHorario.getColumn("Miercoles").setCellRenderer(new JListRenderer());
        tbHorario.getColumn("Jueves").setCellRenderer(new JListRenderer());
        tbHorario.getColumn("Viernes").setCellRenderer(new JListRenderer());
        tbHorario.setCellEditor(new JListEditor());
        /*
```

```
        final MyListModel dlm2 = new MyListModel(new Object[]{"aaa", "bbb"});

        MyListModel[] datos = {dlm2, dlm2, dlm2, dlm2, dlm2, dlm2}; // Cantidad de
columnas de la tabla
        DefaultTableModel model = (DefaultTableModel) tbHorario.getModel();
        model.addRow(datos);
        tbHorario.setModel(model);
         */
        //ProfesoresCmb profe = (ProfesoresCmb) cmbProfes.getSelectedItem();
        //String rut = profe.getRut();

        try {
            tbHorario.setModel(Funciones.FormatearBloquesHorarioConMLM(tbHorario));

        } catch (SQLException ex) {
            Logger.getLogger(JPHorarioProfesores.class.getName()).log(Level.SEVERE,
null, ex);
        }
        //lista.setModel(model);
        tbHorario.setRowHeight(85);
        tbHorario.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
        tbHorario.setCellSelectionEnabled(true);
        //tbHorario.getModel().setValueAt(jList1.getModel(), 1, 1);
        //tbHorario.getModel().setValueAt( v.getText(), 2, 2);

        v.setVisible(false);
        k.setVisible(false);
        n.setVisible(false);
        tbHorario.setShowGrid(true);

        try {
            llenarComboProfesor();
            llenarComboSala();
            llenarComboCurso();
            llenarComboAsig();
        } catch (SQLException ex) {
            Logger.getLogger(JPHorarioProfesores.class.getName()).log(Level.SEVERE,
null, ex);
        }

        cmbProfes.setSelectedIndex(0);
        ProfesoresCmb profe = (ProfesoresCmb) cmbProfes.getSelectedItem();
        String rut = profe.getRut();
        rutGlobal = rut;
        try {
            F5("select * from horario where rut ='" + rut + "'", rut);
```

```
        } catch (SQLException ex) {
            Logger.getLogger(JPHorarioProfesores.class.getName()).log(Level.SEVERE,
null, ex);
        }

        cmbProfes.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                ProfesoresCmb profe = (ProfesoresCmb) cmbProfes.getSelectedItem();
                String rut = profe.getRut();
                rutGlobal = rut;
                try {
                    F5("select * from horario where rut ='" + rut + "'", rut);
                } catch (SQLException ex) {

Logger.getLogger(JPHorarioProfesores.class.getName()).log(Level.SEVERE, null, ex);
                }
            }
        });
    }

    private void llenarComboAsig() throws SQLException {

        Conectar bd = new Conectar();
        ResultSet rs = bd.Consultar("SELECT * FROM asignaturas");
        Vector<AsignaturasCmb> asignaturas = new Vector<AsignaturasCmb>();
        while (rs.next()) {

            AsignaturasCmb asigAux = new AsignaturasCmb(rs.getString("codigoAsig"),
                    rs.getString("nombre"));
            asignaturas.addElement(asigAux);
        }
        if (!asignaturas.isEmpty()) {
            DefaultComboBoxModel modelo = new DefaultComboBoxModel();
            for (int i = 0; i < asignaturas.size(); i++) {
                modelo.addElement(asignaturas.get(i));
            }
            cmbAsginaturas.setModel(modelo);
        }
    }

    private void llenarComboSala() throws SQLException {
        Conectar bd = new Conectar();
        ResultSet rs = bd.Consultar("SELECT * FROM salas");
        Vector<SalasCmb> salas = new Vector<SalasCmb>();
        while (rs.next()) {
            SalasCmb salaAux = new SalasCmb(rs.getString("codigoSala"),
```

```
                rs.getString("nombreSala"), rs.getInt("alumnosMax"));
            salas.addElement(salaAux);
        }
        if (!salas.isEmpty()) {
            DefaultComboBoxModel modelo = new DefaultComboBoxModel();
            for (int i = 0; i < salas.size(); i++) {
                modelo.addElement(salas.get(i));
            }
            cmbSala.setModel(modelo);
        }
    }

    private void llenarComboCurso() throws SQLException {
        Conectar bd = new Conectar();
        ResultSet rs = bd.Consultar("SELECT * FROM cursos");
        Vector<CursosCmb> cursos = new Vector<CursosCmb>();
        while (rs.next()) {
            CursosCmb salaAux = new CursosCmb(rs.getString("codigoCurso"),
                    rs.getString("codigoSala"));
            cursos.addElement(salaAux);
        }
        if (!cursos.isEmpty()) {
            DefaultComboBoxModel modelo = new DefaultComboBoxModel();
            for (int i = 0; i < cursos.size(); i++) {
                modelo.addElement(cursos.get(i));
            }
            cmbCurso.setModel(modelo);
        }
    }

    private void llenarComboProfesor() throws SQLException {
        Conectar bd = new Conectar();
        ResultSet rs = bd.Consultar("SELECT * FROM profesores");
        Vector<ProfesoresCmb> profesores = new Vector<ProfesoresCmb>();
        while (rs.next()) {
            ProfesoresCmb profeAux = new ProfesoresCmb(rs.getString("rut"),
                    rs.getString("nombre"), rs.getString("apePate"));
            profesores.addElement(profeAux);
        }
        if (!profesores.isEmpty()) {
            DefaultComboBoxModel modelo = new DefaultComboBoxModel();
            for (int i = 0; i < profesores.size(); i++) {
                modelo.addElement(profesores.get(i));
            }
            cmbProfes.setModel(modelo);
        }
```

```
}

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jScrollPane1 = new javax.swing.JScrollPane();
    tbHorario = new javax.swing.JTable();
    jScrollPane3 = new javax.swing.JScrollPane();
    v = new javax.swing.JTextArea();
    jScrollPane6 = new javax.swing.JScrollPane();
    k = new javax.swing.JTextArea();
    jScrollPane7 = new javax.swing.JScrollPane();
    n = new javax.swing.JTextArea();
    cmbProfes = new javax.swing.JComboBox<>();
    jLabel6 = new javax.swing.JLabel();
    jLabel7 = new javax.swing.JLabel();
    jLabel8 = new javax.swing.JLabel();
    cmbAsginaturas = new javax.swing.JComboBox<>();
    cmbCurso = new javax.swing.JComboBox<>();
    cmbSala = new javax.swing.JComboBox<>();
    btnBorrar = new javax.swing.JButton();
    btnInsertar = new javax.swing.JButton();
    lblError = new javax.swing.JLabel();

    tbHorario.setModel(new javax.swing.table.DefaultTableModel(
        new Object [][] {

        },
        new String [] {
            "Hora", "Lunes", "Martes", "Miercoles", "Jueves", "Viernes"
        }
    ) {
        boolean[] canEdit = new boolean [] {
            false, false, false, false, false, false
        };

        public boolean isCellEditable(int rowIndex, int columnIndex) {
            return canEdit [columnIndex];
        }
    });
```

```
        tbHorario.setColumnSelectionAllowed(true);
        tbHorario.setRowSelectionAllowed(false);

tbHorario.setSelectionMode(javax.swing.ListSelectionModel.SINGLE_SELECTION);
        tbHorario.getTableHeader().setReorderingAllowed(false);
        jScrollPane1.setViewportView(tbHorario);

tbHorario.getColumnModel().getSelectionModel().setSelectionMode(javax.swing.ListSel
ectionModel.SINGLE_SELECTION);
        if (tbHorario.getColumnModel().getColumnCount() > 0) {
            tbHorario.getColumnModel().getColumn(0).setResizable(false);
            tbHorario.getColumnModel().getColumn(1).setResizable(false);
            tbHorario.getColumnModel().getColumn(2).setResizable(false);
            tbHorario.getColumnModel().getColumn(3).setResizable(false);
            tbHorario.getColumnModel().getColumn(4).setResizable(false);
            tbHorario.getColumnModel().getColumn(5).setResizable(false);
        }

        v.setColumns(20);
        v.setRows(5);
        v.setText("Oscar Perez\n5toA\nMatematica\n104\n");
        jScrollPane3.setViewportView(v);

        k.setColumns(20);
        k.setRows(5);
        k.setText("Oscar Perez\n6toC\nMatematica\n107\n");
        jScrollPane6.setViewportView(k);

        n.setColumns(20);
        n.setRows(5);
        n.setText("Oscar Perez\n1roA\nMatematica\n201\n");
        jScrollPane7.setViewportView(n);

        jLabel6.setText("Asignatura :");

        jLabel7.setText("Curso :");

        jLabel8.setText("Sala : ");

        cmbAsginaturas.setModel(new javax.swing.DefaultComboBoxModel<>(new
String[] { "Item 1", "Item 2", "Item 3", "Item 4" }));

        cmbCurso.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] {
"Item 1", "Item 2", "Item 3", "Item 4" }));
```

```
        cmbSala.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] {
"Item 1", "Item 2", "Item 3", "Item 4" }));

        btnBorrar.setText("Borrar");
        btnBorrar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnBorrarActionPerformed(evt);
            }
        });

        btnInsertar.setText("Insertar");
        btnInsertar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnInsertarActionPerformed(evt);
            }
        });

        lblError.setBackground(new java.awt.Color(255, 255, 255));
        lblError.setForeground(new java.awt.Color(255, 0, 0));

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jScrollPane1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jScrollPane7, javax.swing.GroupLayout.PREFERRED_SIZE,
0, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(lblError, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jScrollPane3,
javax.swing.GroupLayout.PREFERRED_SIZE, 0,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(180, 180, 180)
                        .addComponent(jScrollPane6,
javax.swing.GroupLayout.PREFERRED_SIZE, 0,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
                .addGroup(layout.createSequentialGroup()
                    .addGap(80, 80, 80)
                    .addComponent(cmbProfes,
javax.swing.GroupLayout.PREFERRED_SIZE, 159,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(layout.createSequentialGroup()
                    .addGap(36, 36, 36)
                    .addComponent(jLabel6, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(cmbAsginaturas, 0, 187, Short.MAX_VALUE)
                    .addGap(22, 22, 22)
                    .addComponent(jLabel7, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGap(18, 18, 18)
                    .addComponent(cmbCurso, 0, 132, Short.MAX_VALUE)
                    .addGap(18, 18, 18)
                    .addComponent(jLabel8, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGap(18, 18, 18)
                    .addComponent(cmbSala, 0, 132, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(btnInsertar,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(btnBorrar,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                    .addGap(14, 14, 14)))
            .addGap(32, 32, 32))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(10, 10, 10)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                .addGroup(layout.createSequentialGroup()
                    .addComponent(jScrollPane7,
javax.swing.GroupLayout.PREFERRED_SIZE, 66,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(199, 199, 199))
```

```
                .addGroup(layout.createSequentialGroup()
                    .addComponent(cmbProfes,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addComponent(jScrollPane3,
javax.swing.GroupLayout.PREFERRED_SIZE, 5, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(jScrollPane6,
javax.swing.GroupLayout.PREFERRED_SIZE, 0,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(btnInsertar)
                    .addGap(22, 22, 22)
                    .addComponent(btnBorrar)
                    .addGap(48, 48, 48))
                .addGroup(layout.createSequentialGroup()
                    .addComponent(lblError, javax.swing.GroupLayout.PREFERRED_SIZE,
22, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE
)
                        .addComponent(jLabel6)
                        .addComponent(cmbAsginaturas,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(cmbCurso,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(jLabel7)
                        .addComponent(cmbSala,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(jLabel8))
                    .addGap(80, 80, 80))))
        );
    }// </editor-fold>

    private void btnInsertarActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        int row = tbHorario.getSelectedRow();
        int col = tbHorario.getSelectedColumn();
        int numBloque = row + 1;
        ProfesoresCmb profe = (ProfesoresCmb) cmbProfes.getSelectedItem();
        CursosCmb curso = (CursosCmb) cmbCurso.getSelectedItem();
        AsignaturasCmb asignatura = (AsignaturasCmb) cmbAsginaturas.getSelectedItem();
        SalasCmb sala = (SalasCmb) cmbSala.getSelectedItem();

        String rut = profe.getRut();
        String codCurso = curso.getCodigoCurso();
        String codAsig = asignatura.getCodigoAsig();
        String codSala = sala.getCodigoSala();
        lblError.setForeground(rojo);
        try {
            if (Funciones.ValidarInsertHoraioRUT(col, numBloque, rut)) {
                if (Funciones.ValidarInsertHoraioCURSO(col, numBloque, codCurso)) {
                    if (Funciones.ValidarInsertHoraioSALA(col, numBloque, codSala)) {
                        if (Funciones.ValidarDisponibilidadHorario(col, numBloque, rut)) {
                            if (Funciones.Validarhoras(rut)) {
                                DefaultListModel<Object> model = new DefaultListModel<>();
                                //DefaultListModel model = new DefaultListModel();
                                model.addElement(profe);
                                model.addElement(curso);
                                model.addElement(sala);
                                model.addElement(asignatura);

                                JList<Object> lista = new JList<>(model);
                                lista.setLayoutOrientation(JList.VERTICAL);

                                MyListModel dlm1 = new MyListModel(new Object[]{profe, curso,
asignatura, sala});
                                tbHorario.getModel().setValueAt(dlm1, row, col);
```

```
                        Conectar bd = new Conectar();
                        bd.IngresarSinPreguntar("INSERT INTO `horario` (`numeroBloque`,
`dia`, `rut`, `codigoCurso`, "
                                + "`codigoAsig`, `codigoSala`, `anno`)"
                                + " VALUES ('" + numBloque + "', '" + col + "', '" + rut + "', '" +
curso.getCodigoCurso() + "', '" + asignatura.getCodigoAsig() + "', '" +
sala.getCodigoSala() + "', '2018');");
                        lblError.setForeground(verde);
                        lblError.setText("INGRESADO CON EXITO");
                    } else {

                        lblError.setText("Profesor : " + rut + " Supera su limite de horas
maximas");
                    }
                } else {
                    lblError.setText("Profesor : " + rut + " No tiene disponibilidad en este
Bloque");
                }
            } else {
                lblError.setText("Sala : " + codSala + " ya esta siendo ocupada  en ese
bloque");
            }
        } else {
            lblError.setText("Curso : " + codCurso + " ya esta siendo ocupado  en ese
bloque");
        }
    } else {
        lblError.setText("Profesor : " + rut + " ya esta siendo ocupado  en ese bloque");
    }
} catch (SQLException ex) {
    Logger.getLogger(JPHorarioProfesores.class.getName()).log(Level.SEVERE,
null, ex);
} catch (ParseException ex) {
    Logger.getLogger(JPHorarioProfesores.class.getName()).log(Level.SEVERE,
null, ex);
}


}

public void F5(String sql, String rut) throws SQLException {

    tbHorario.setModel(Funciones.LlenarHorarioPROFESOR(sql, tbHorario, rut));
}
private void btnBorrarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
```

```
        int row = tbHorario.getSelectedRow();
        int col = tbHorario.getSelectedColumn();
        int numeroBloque = row + 1;
        MyListModel mlmDispo = new MyListModel(new Object[]{"Disponible"});
        tbHorario.getModel().setValueAt(mlmDispo, row, col);
        Conectar bd = new Conectar();
        bd.EliminarSinPreguntar("DELETE FROM horario where rut ='" + rutGlobal + "'
and dia =" + col + " and numeroBloque =" + numeroBloque);

    }


    // Variables declaration - do not modify
    private javax.swing.JButton btnBorrar;
    private javax.swing.JButton btnInsertar;
    private javax.swing.JComboBox<String> cmbAsginaturas;
    private javax.swing.JComboBox<String> cmbCurso;
    private javax.swing.JComboBox<String> cmbProfes;
    private javax.swing.JComboBox<String> cmbSala;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane3;
    private javax.swing.JScrollPane jScrollPane6;
    private javax.swing.JScrollPane jScrollPane7;
    private javax.swing.JTextArea k;
    private javax.swing.JLabel lblError;
    private javax.swing.JTextArea n;
    private javax.swing.JTable tbHorario;
    private javax.swing.JTextArea v;
    // End of variables declaration
}
```

Horario Global

```
import java.sql.SQLException;
import javax.swing.ListSelectionModel;
import javax.swing.table.DefaultTableModel;

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
```

```
/**
 *
 * @author felipe
 */
public class JPHorarioGlobal extends javax.swing.JPanel {

    /**
     * Creates new form JPHorarioGlobal
     */
    public JPHorarioGlobal() throws SQLException {
        initComponents();
        DefaultTableModel dtm = new DefaultTableModel() {
            @Override
            public boolean isCellEditable(int row, int column) {
                //all cells false
                return false;
            }
        };
        dtm.addColumn("Horas");
        tbLunes.setModel(dtm);
        tbLunes.setModel(Funciones.FormatearBloquesHorarioPorDIA(tbLunes));
        tbLunes.setModel(Funciones.AgregarColumnasCursos(tbLunes));
        tbLunes.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
        tbLunes.setShowGrid(true);

        dtm = new DefaultTableModel() {
            @Override
            public boolean isCellEditable(int row, int column) {
                //all cells false
                return false;
            }
        };
        dtm.addColumn("Horas");
        tbMartes.setModel(dtm);
        tbMartes.setModel(Funciones.FormatearBloquesHorarioPorDIA(tbMartes));
        tbMartes.setModel(Funciones.AgregarColumnasCursos(tbMartes));
        tbMartes.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
        tbMartes.setShowGrid(true);

        dtm = new DefaultTableModel() {
            @Override
            public boolean isCellEditable(int row, int column) {
                //all cells false
                return false;
            }
```

```
};
dtm.addColumn("Horas");
tbMiercoles.setModel(dtm);
tbMiercoles.setModel(Funciones.FormatearBloquesHorarioPorDIA(tbMiercoles));
tbMiercoles.setModel(Funciones.AgregarColumnasCursos(tbMiercoles));
tbMiercoles.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
tbMiercoles.setShowGrid(true);

dtm = new DefaultTableModel() {
   @Override
   public boolean isCellEditable(int row, int column) {
      //all cells false
      return false;
   }
};
dtm.addColumn("Horas");
tbJueves.setModel(dtm);
tbJueves.setModel(Funciones.FormatearBloquesHorarioPorDIA(tbJueves));
tbJueves.setModel(Funciones.AgregarColumnasCursos(tbJueves));
tbJueves.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
tbJueves.setShowGrid(true);

dtm = new DefaultTableModel() {
   @Override
   public boolean isCellEditable(int row, int column) {
      //all cells false
      return false;
   }
};
dtm.addColumn("Horas");
tbViernes.setModel(dtm);
tbViernes.setModel(Funciones.FormatearBloquesHorarioPorDIA(tbViernes));
tbViernes.setModel(Funciones.AgregarColumnasCursos(tbViernes));
tbViernes.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
tbViernes.setShowGrid(true);


tbLunes.setRowHeight(20);
tbMartes.setRowHeight(20);
tbMiercoles.setRowHeight(20);
tbJueves.setRowHeight(20);
tbViernes.setRowHeight(20);
tbLunes.setModel(Funciones.PoblarTablasHorarioPorDIA(tbLunes, 1));
tbMartes.setModel(Funciones.PoblarTablasHorarioPorDIA(tbMartes, 2));
tbMiercoles.setModel(Funciones.PoblarTablasHorarioPorDIA(tbMiercoles, 3));
tbJueves.setModel(Funciones.PoblarTablasHorarioPorDIA(tbJueves, 4));
```

```
        tbViernes.setModel(Funciones.PoblarTablasHorarioPorDIA(tbViernes, 5));
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jPanel7 = new javax.swing.JPanel();
        jLabel4 = new javax.swing.JLabel();
        scrollPane2 = new java.awt.ScrollPane();
        jPanel2 = new javax.swing.JPanel();
        jScrollPane1 = new javax.swing.JScrollPane();
        tbMartes = new javax.swing.JTable();
        jScrollPane2 = new javax.swing.JScrollPane();
        tbLunes = new javax.swing.JTable();
        jScrollPane5 = new javax.swing.JScrollPane();
        tbMiercoles = new javax.swing.JTable();
        jScrollPane6 = new javax.swing.JScrollPane();
        tbViernes = new javax.swing.JTable();
        jScrollPane7 = new javax.swing.JScrollPane();
        tbJueves = new javax.swing.JTable();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel6 = new javax.swing.JLabel();

        javax.swing.GroupLayout jPanel7Layout = new javax.swing.GroupLayout(jPanel7);
        jPanel7.setLayout(jPanel7Layout);
        jPanel7Layout.setHorizontalGroup(

jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 100, Short.MAX_VALUE)
        );
        jPanel7Layout.setVerticalGroup(

jPanel7Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 100, Short.MAX_VALUE)
        );

        jLabel4.setText("JUEVES  :");
```

```
        setPreferredSize(new java.awt.Dimension(978, 566));

        scrollPane2.setPreferredSize(new java.awt.Dimension(1000, 566));

        tbMartes.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "Hora"
            }
        ) {
            boolean[] canEdit = new boolean [] {
                false
            };

            public boolean isCellEditable(int rowIndex, int columnIndex) {
                return canEdit [columnIndex];
            }
        });
        tbMartes.setColumnSelectionAllowed(true);
        tbMartes.getTableHeader().setReorderingAllowed(false);
        jScrollPane1.setViewportView(tbMartes);

tbMartes.getColumnModel().getSelectionModel().setSelectionMode(javax.swing.ListSel
ectionModel.SINGLE_SELECTION);
        if (tbMartes.getColumnModel().getColumnCount() > 0) {
            tbMartes.getColumnModel().getColumn(0).setResizable(false);
        }

        tbLunes.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "Hora"
            }
        ) {
            boolean[] canEdit = new boolean [] {
                false
            };

            public boolean isCellEditable(int rowIndex, int columnIndex) {
                return canEdit [columnIndex];
            }
```

```
        });
        tbLunes.getTableHeader().setReorderingAllowed(false);
        jScrollPane2.setViewportView(tbLunes);

tbLunes.getColumnModel().getSelectionModel().setSelectionMode(javax.swing.ListSele
ctionModel.SINGLE_SELECTION);
        if (tbLunes.getColumnModel().getColumnCount() > 0) {
            tbLunes.getColumnModel().getColumn(0).setResizable(false);
        }

        tbMiercoles.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "Hora"
            }
        ) {
            boolean[] canEdit = new boolean [] {
                false
            };

            public boolean isCellEditable(int rowIndex, int columnIndex) {
                return canEdit [columnIndex];
            }
        });
        tbMiercoles.getTableHeader().setReorderingAllowed(false);
        jScrollPane5.setViewportView(tbMiercoles);

tbMiercoles.getColumnModel().getSelectionModel().setSelectionMode(javax.swing.List
SelectionModel.SINGLE_SELECTION);
        if (tbMiercoles.getColumnModel().getColumnCount() > 0) {
            tbMiercoles.getColumnModel().getColumn(0).setResizable(false);
        }

        tbViernes.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "Hora"
            }
        ) {
            boolean[] canEdit = new boolean [] {
                false
            };
```

```
            public boolean isCellEditable(int rowIndex, int columnIndex) {
                return canEdit [columnIndex];
            }
        });
        tbViernes.getTableHeader().setReorderingAllowed(false);
        jScrollPane6.setViewportView(tbViernes);

tbViernes.getColumnModel().getSelectionModel().setSelectionMode(javax.swing.ListSel
ectionModel.SINGLE_SELECTION);
        if (tbViernes.getColumnModel().getColumnCount() > 0) {
            tbViernes.getColumnModel().getColumn(0).setResizable(false);
        }

        tbJueves.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "Hora"
            }
        ) {
            boolean[] canEdit = new boolean [] {
                false
            };

            public boolean isCellEditable(int rowIndex, int columnIndex) {
                return canEdit [columnIndex];
            }
        });
        tbJueves.getTableHeader().setReorderingAllowed(false);
        jScrollPane7.setViewportView(tbJueves);

tbJueves.getColumnModel().getSelectionModel().setSelectionMode(javax.swing.ListSele
ctionModel.SINGLE_SELECTION);
        if (tbJueves.getColumnModel().getColumnCount() > 0) {
            tbJueves.getColumnModel().getColumn(0).setResizable(false);
        }

        jLabel1.setText("LUNES :");

        jLabel2.setText("MARTES  :");

        jLabel3.setText("MIERCOLES  :");

        jLabel5.setText("VIERNES  :");
```

```
        jLabel5.setToolTipText("");

        jLabel6.setText("JUEVES  :");

        javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                    .addComponent(jLabel1)
                    .addComponent(jLabel5)
                    .addComponent(jLabel6)
                    .addComponent(jLabel3)
                    .addComponent(jLabel2))
                .addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                    .addComponent(jScrollPane2)
                    .addComponent(jScrollPane1)
                    .addComponent(jScrollPane7)
                    .addComponent(jScrollPane5,
javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(jScrollPane6))
                .addContainerGap())
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addGap(104, 104, 104)
                        .addComponent(jLabel1))
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE, 220,
javax.swing.GroupLayout.PREFERRED_SIZE)))
```

```
.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addGroup(jPanel2Layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 220,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(jPanel2Layout.createSequentialGroup()
                    .addGap(105, 105, 105)
                    .addComponent(jLabel2)))

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addGroup(jPanel2Layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(jScrollPane5,
javax.swing.GroupLayout.PREFERRED_SIZE, 220,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(jPanel2Layout.createSequentialGroup()
                    .addGap(113, 113, 113)
                    .addComponent(jLabel3)))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addComponent(jScrollPane7,
javax.swing.GroupLayout.PREFERRED_SIZE, 220,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel2Layout.createSequentialGroup()
                    .addComponent(jLabel6)
                    .addGap(95, 95, 95)))

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addGroup(jPanel2Layout.createSequentialGroup()
                    .addGap(120, 120, 120)
                    .addComponent(jLabel5))
                .addComponent(jScrollPane6,
javax.swing.GroupLayout.PREFERRED_SIZE, 220,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
```

```
        );

        scrollPane2.add(jPanel2);

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(scrollPane2, javax.swing.GroupLayout.DEFAULT_SIZE, 978,
Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(scrollPane2, javax.swing.GroupLayout.DEFAULT_SIZE, 1145,
Short.MAX_VALUE)
        );
    }// </editor-fold>


    // Variables declaration - do not modify
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel7;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JScrollPane jScrollPane5;
    private javax.swing.JScrollPane jScrollPane6;
    private javax.swing.JScrollPane jScrollPane7;
    private java.awt.ScrollPane scrollPane2;
    private javax.swing.JTable tbJueves;
    private javax.swing.JTable tbLunes;
    private javax.swing.JTable tbMartes;
    private javax.swing.JTable tbMiercoles;
    private javax.swing.JTable tbViernes;
    // End of variables declaration
}
```

Gestionar Profesores

```
import java.io.IOException;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import java.util.regex.Pattern;
import javax.swing.JFrame;
import javax.swing.ListSelectionModel;
import javax.swing.RowFilter;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;
import javax.swing.table.DefaultTableModel;
import javax.swing.table.TableModel;
import javax.swing.table.TableRowSorter;


/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

/**
 *
 * @author labinf10
 */
public class JPListaProfesores extends javax.swing.JPanel {

    /**
     * Creates new form JPListaProfesores
     */
    public JPListaProfesores() {
        initComponents();

    }

    protected Menu padre;

    public JPListaProfesores(Menu p) throws SQLException {
        try {
            initComponents();
            padre=p;

            lblLupa.setSize(80, 80);
            java.awt.image.BufferedImage img = null;
```

```
            img =
javax.imageio.ImageIO.read(getClass().getResource("/Imagenes/lupa2.png"));
            java.awt.Image dimg = img.getScaledInstance(20,20,
java.awt.Image.SCALE_SMOOTH);
            lblLupa.setIcon(new javax.swing.ImageIcon(dimg));
        } catch (IOException ex) {
            Logger.getLogger(JPListaProfesores.class.getName()).log(Level.SEVERE, null,
ex);
        }
        this.F5();

        tbProfesores.getColumn(tbProfesores.getColumnName(7)).setWidth(30);
        ListSelectionModel model = tbProfesores.getSelectionModel();
        model.addListSelectionListener(new ListSelectionListener() {
            @Override
            public void valueChanged(ListSelectionEvent e) {
                if (model.isSelectionEmpty()) {
                    btnModificar.setEnabled(false);
                    btnEliminar.setEnabled(false);
                    btnVer.setEnabled(false);
                }else{
                    btnModificar.setEnabled(true);
                    btnEliminar.setEnabled(true);
                    btnVer.setEnabled(true);
                }
            }
        });


    }


    public void F5() throws SQLException{
        Funciones.mostrar("Select * FROM profesores", tbProfesores);
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jFrame1 = new javax.swing.JFrame();
```

```
        jScrollPane1 = new javax.swing.JScrollPane();
        tbProfesores = new javax.swing.JTable();
        btnAgregar = new javax.swing.JButton();
        btnModificar = new javax.swing.JButton();
        btnEliminar = new javax.swing.JButton();
        txtFiltro = new javax.swing.JTextField();
        lblLupa = new javax.swing.JLabel();
        btnVer = new javax.swing.JButton();

        javax.swing.GroupLayout jFrame1Layout = new
javax.swing.GroupLayout(jFrame1.getContentPane());
        jFrame1.getContentPane().setLayout(jFrame1Layout);
        jFrame1Layout.setHorizontalGroup(

jFrame1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 400, Short.MAX_VALUE)
        );
        jFrame1Layout.setVerticalGroup(

jFrame1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 300, Short.MAX_VALUE)
        );

        setPreferredSize(new java.awt.Dimension(796, 464));
        addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusGained(java.awt.event.FocusEvent evt) {
                formFocusGained(evt);
            }
            public void focusLost(java.awt.event.FocusEvent evt) {
                formFocusLost(evt);
            }
        });
        addPropertyChangeListener(new java.beans.PropertyChangeListener() {
            public void propertyChange(java.beans.PropertyChangeEvent evt) {
                formPropertyChange(evt);
            }
        });

        tbProfesores.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "Rut", "Nombre", "Ap. Paterno", "Ap. Materno", "Direccion", "Telefono",
"Email", "Horas "
            }
```

```
        ) {
            Class[] types = new Class [] {
                java.lang.String.class, java.lang.String.class, java.lang.String.class,
java.lang.String.class, java.lang.String.class, java.lang.String.class, java.lang.String.class,
java.lang.Byte.class
            };
            boolean[] canEdit = new boolean [] {
                false, false, false, false, false, false, false, false
            };

            public Class getColumnClass(int columnIndex) {
                return types [columnIndex];
            }

            public boolean isCellEditable(int rowIndex, int columnIndex) {
                return canEdit [columnIndex];
            }
        });
        tbProfesores.getTableHeader().setReorderingAllowed(false);
        tbProfesores.setVerifyInputWhenFocusTarget(false);
        tbProfesores.addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusGained(java.awt.event.FocusEvent evt) {
                tbProfesoresFocusGained(evt);
            }
            public void focusLost(java.awt.event.FocusEvent evt) {
                tbProfesoresFocusLost(evt);
            }
        });
        tbProfesores.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                tbProfesoresMouseClicked(evt);
            }
        });
        jScrollPane1.setViewportView(tbProfesores);
        if (tbProfesores.getColumnModel().getColumnCount() > 0) {
            tbProfesores.getColumnModel().getColumn(7).setResizable(false);
        }
        tbProfesores.getAccessibleContext().setAccessibleName("");

        btnAgregar.setText("Agregar");
        btnAgregar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnAgregarActionPerformed(evt);
            }
        });
```

```
        btnModificar.setText("Modificar");
        btnModificar.setEnabled(false);
        btnModificar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnModificarActionPerformed(evt);
            }
        });

        btnEliminar.setText("Eliminar");
        btnEliminar.setEnabled(false);
        btnEliminar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnEliminarActionPerformed(evt);
            }
        });

        txtFiltro.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtFiltroActionPerformed(evt);
            }
        });
        txtFiltro.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyReleased(java.awt.event.KeyEvent evt) {
                txtFiltroKeyReleased(evt);
            }
        });

        lblLupa.setToolTipText("");

        btnVer.setText("Ver/Editar Disponibilidad");
        btnVer.setEnabled(false);
        btnVer.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnVerActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(30, 30, 30)
                .addComponent(lblLupa, javax.swing.GroupLayout.PREFERRED_SIZE, 35,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(27, 27, 27)
```

```
            .addComponent(txtFiltro, javax.swing.GroupLayout.PREFERRED_SIZE, 99,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
               .addContainerGap()
               .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 828, Short.MAX_VALUE))
            .addGroup(layout.createSequentialGroup()
               .addGap(22, 22, 22)
               .addComponent(btnAgregar, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
               .addGap(27, 27, 27)
               .addComponent(btnModificar,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
               .addGap(41, 41, 41)
               .addComponent(btnEliminar,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
               .addGap(81, 81, 81)
               .addComponent(btnVer, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
               .addGap(40, 40, 40)))
            .addContainerGap())
      );
      layout.setVerticalGroup(
         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
         .addGroup(layout.createSequentialGroup()
            .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
            .addComponent(txtFiltro, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(lblLupa, javax.swing.GroupLayout.PREFERRED_SIZE,
29, javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
365, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(18, 18, 18)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE
)
                .addComponent(btnAgregar, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(btnModificar, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(btnEliminar, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(btnVer))
            .addContainerGap())
      );
   }// </editor-fold>

   private void txtFiltroKeyReleased(java.awt.event.KeyEvent evt) {
      // TODO add your handling code here:

      TableRowSorter<TableModel> sorter = new
TableRowSorter<TableModel>(((DefaultTableModel) tbProfesores.getModel()));
      String filtro = txtFiltro.getText();
      //Pattern patron = Pattern.compile("alt",Pattern.CASE_INSENSITIVE);
      sorter.setRowFilter(RowFilter.regexFilter("(?i)" + filtro));
      tbProfesores.setRowSorter(sorter);
      if (tbProfesores.getSelectedRow() == -1){
         btnModificar.setEnabled(false);
         btnEliminar.setEnabled(false);
      };
   }

   private void btnModificarActionPerformed(java.awt.event.ActionEvent evt) {
         // TODO add your handling code here:

      String rut=  tbProfesores.getValueAt(tbProfesores.getSelectedRow(), 0).toString();
      String nom = tbProfesores.getValueAt(tbProfesores.getSelectedRow(), 1).toString();
      String pate = tbProfesores.getValueAt(tbProfesores.getSelectedRow(), 2).toString();
      String mate = tbProfesores.getValueAt(tbProfesores.getSelectedRow(),
3).toString();
      String  dir = tbProfesores.getValueAt(tbProfesores.getSelectedRow(), 4).toString();
      String tel = tbProfesores.getValueAt(tbProfesores.getSelectedRow(), 5).toString();
      String email = tbProfesores.getValueAt(tbProfesores.getSelectedRow(),
6).toString();
      int max = Integer.parseInt(tbProfesores.getValueAt(tbProfesores.getSelectedRow(),
7).toString());
      dProfe mod= new dProfe((Menu)
padre,true,true,false,rut,nom,pate,mate,dir,tel,email,max);
      padre.setEnabled(false);
```

```
    mod.setVisible(true);

}

private void tbProfesoresFocusGained(java.awt.event.FocusEvent evt) {
    // TODO add your handling code here:

}

private void formFocusGained(java.awt.event.FocusEvent evt) {

    if (tbProfesores.getSelectedRow() != -1){
        btnModificar.setEnabled(true);
        btnEliminar.setEnabled(true);
    };
}

private void formFocusLost(java.awt.event.FocusEvent evt) {
    // TODO add your handling code here:
}

private void tbProfesoresFocusLost(java.awt.event.FocusEvent evt) {
    // TODO add your handling code here:
    if (tbProfesores.getSelectedRow() == -1){
        btnModificar.setEnabled(false);
        btnEliminar.setEnabled(false);
    };
}

private void tbProfesoresMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    if (tbProfesores.getSelectedRow() != -1){
        btnModificar.setEnabled(true);
        btnEliminar.setEnabled(true);
    };
}

private void txtFiltroActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void btnAgregarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    dProfe mod= new dProfe(padre,true, false, true);
    padre.setEnabled(false);
    mod.setVisible(true);
```

```
        btnModificar.setEnabled(false);
        btnEliminar.setEnabled(false);

    }

    private void formPropertyChange(java.beans.PropertyChangeEvent evt) {
        // TODO add your handling code here:
    }

    private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:

        String rut = tbProfesores.getValueAt(tbProfesores.getSelectedRow(),0).toString();
        String sql = "DELETE FROM `Wolfo`.`profesores` WHERE `profesores`.`rut` =
'"+rut+"'";
        System.out.println(rut);
        Conectar conexion = new Conectar();
        try {
            if(Funciones.AdvertenciaHoraio("select * from horario where rut ='"+rut+"'")){
                conexion.Eliminar(sql);
            }
        } catch (SQLException ex) {
            Logger.getLogger(JPListaProfesores.class.getName()).log(Level.SEVERE, null,
ex);
        }

        try {
            Funciones.mostrar("SELECT * FROM profesores",tbProfesores);
        } catch (SQLException ex) {
            Logger.getLogger(JPListaProfesores.class.getName()).log(Level.SEVERE, null,
ex);
        }
        if (tbProfesores.getSelectedRow() == -1){
            btnModificar.setEnabled(false);
            btnEliminar.setEnabled(false);
            btnVer.setEnabled(false);
        };
    }

    private void btnVerActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String rut=  tbProfesores.getValueAt(tbProfesores.getSelectedRow(), 0).toString();
        JDHorarioDisponible hDisponible = new JDHorarioDisponible(padre, true,true,rut);
        hDisponible.setVisible(true);
    }
```

```
    // Variables declaration - do not modify
    private javax.swing.JButton btnAgregar;
    private javax.swing.JButton btnEliminar;
    private javax.swing.JButton btnModificar;
    private javax.swing.JButton btnVer;
    private javax.swing.JFrame jFrame1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel lblLupa;
    private javax.swing.JTable tbProfesores;
    private javax.swing.JTextField txtFiltro;
    // End of variables declaration
}
```

Agregar Y Modificar Profesores

```
import java.awt.Color;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.*;


/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
/**
 *
 * @author felipe
 */
public class dProfe extends javax.swing.JDialog {

    /**
     * Creates new form dProfe
     */
    Menu padre;
    Color rojo = new Color(255, 177, 177);
    Color verde = new Color(167, 255, 167);
    Color blanco = new Color(255, 255, 255);
    boolean rutOk = false;
    boolean telOk = false;
```

```
    boolean emailOk = false;

    public dProfe(Menu p) {
        initComponents();
        padre = p;
        this.setLocationRelativeTo(null);
        AsignarLimitesTxt();
    }

    public dProfe(Menu m, boolean modal, boolean visM, boolean visG) {
        super(m, modal);
        initComponents();
        this.setLocationRelativeTo(null);
        AsignarLimitesTxt();
        btnModificar.setVisible(visM);
        btnGuardar.setVisible(visG);
        padre = m;

    }

    public dProfe(Menu m, boolean modal, boolean visM, boolean visG, String rut, String
nombre, String pate, String mate, String direccion, String telefono, String email, int max)
{
        super(m, modal);
        initComponents();

        padre = m;
        AsignarLimitesTxt();
        btnModificar.setVisible(visM);
        btnGuardar.setVisible(visG);
        txtRut.setText(rut);
        txtNombre.setText(nombre);
        txtApellidoP.setText(pate);
        txtApellidoM.setText(mate);
        txtTelefono.setText(telefono);
        txtDireccion.setText(direccion);
        txtEmail.setText(email);
        spnHoras.setValue(max);
        txtRut.setEnabled(false);
        rutOk = true;
        telOk = true;
        emailOk = true;
        this.setLocationRelativeTo(null);
    }

    private dProfe(JFrame jFrame, boolean b) {
```

```
        throw new UnsupportedOperationException("Not supported yet."); //To change
body of generated methods, choose Tools | Templates.
    }

    private void AsignarLimitesTxt() {
        txtApellidoM.setDocument(new JTextFieldLimit(20));
        txtApellidoP.setDocument(new JTextFieldLimit(20));
        txtDireccion.setDocument(new JTextFieldLimit(50));
        txtEmail.setDocument(new JTextFieldLimit(50));
        txtNombre.setDocument(new JTextFieldLimit(30));
        txtRut.setDocument(new JTextFieldLimit(10));
        txtTelefono.setDocument(new JTextFieldLimit(10));
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        lblNombre = new javax.swing.JLabel();
        lblApellidoP = new javax.swing.JLabel();
        lblApellidoM = new javax.swing.JLabel();
        txtApellidoP = new javax.swing.JTextField();
        txtApellidoM = new javax.swing.JTextField();
        txtNombre = new javax.swing.JTextField();
        cmdSalir = new javax.swing.JButton();
        btnGuardar = new javax.swing.JButton();
        lblRut = new javax.swing.JLabel();
        txtRut = new javax.swing.JTextField();
        lblDireccion = new javax.swing.JLabel();
        txtDireccion = new javax.swing.JTextField();
        jLabel2 = new javax.swing.JLabel();
        txtTelefono = new javax.swing.JTextField();
        btnModificar = new javax.swing.JButton();
        jLabel5 = new javax.swing.JLabel();
        txtEmail = new javax.swing.JTextField();
        jLabel6 = new javax.swing.JLabel();
        spnHoras = new javax.swing.JSpinner();
        jLabel3 = new javax.swing.JLabel();
```

```
setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
setLocation(new java.awt.Point(900, 600));
setResizable(false);
addWindowListener(new java.awt.event.WindowAdapter() {
    public void windowClosed(java.awt.event.WindowEvent evt) {
        formWindowClosed(evt);
    }
    public void windowOpened(java.awt.event.WindowEvent evt) {
        formWindowOpened(evt);
    }
});

jPanel1.setBackground(new java.awt.Color(239, 255, 255));
jPanel1.setToolTipText("");
jPanel1.setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));
jPanel1.setMinimumSize(new java.awt.Dimension(700, 500));

jLabel1.setFont(new java.awt.Font("Sitka Display", 2, 24)); // NOI18N
jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
jLabel1.setText("Mantenedor de profesores");

lblNombre.setText("Nombre:");

lblApellidoP.setText("Apellido Paterno:");

lblApellidoM.setText("Apellido Materno:");

txtApellidoP.addFocusListener(new java.awt.event.FocusAdapter() {
    public void focusLost(java.awt.event.FocusEvent evt) {
        txtApellidoPFocusLost(evt);
    }
});
txtApellidoP.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        txtApellidoPActionPerformed(evt);
    }
});
txtApellidoP.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyTyped(java.awt.event.KeyEvent evt) {
        txtApellidoPKeyTyped(evt);
    }
});

txtApellidoM.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyTyped(java.awt.event.KeyEvent evt) {
        txtApellidoMKeyTyped(evt);
```

```
    }
});

txtNombre.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        txtNombreActionPerformed(evt);
    }
});
txtNombre.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyTyped(java.awt.event.KeyEvent evt) {
        txtNombreKeyTyped(evt);
    }
});

cmdSalir.setMnemonic('s');
cmdSalir.setText("Volver");
cmdSalir.setToolTipText("");
cmdSalir.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        cmdSalirActionPerformed(evt);
    }
});

btnGuardar.setText("Guardar");
btnGuardar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnGuardarActionPerformed(evt);
    }
});

lblRut.setText("Rut:");

txtRut.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        txtRutActionPerformed(evt);
    }
});
txtRut.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyReleased(java.awt.event.KeyEvent evt) {
        txtRutKeyReleased(evt);
    }
    public void keyTyped(java.awt.event.KeyEvent evt) {
        txtRutKeyTyped(evt);
    }
});
```

```
lblDireccion.setText("Direccion:");

txtDireccion.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        txtDireccionActionPerformed(evt);
    }
});
txtDireccion.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyTyped(java.awt.event.KeyEvent evt) {
        txtDireccionKeyTyped(evt);
    }
});

jLabel2.setText("Telefono: ");

txtTelefono.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        txtTelefonoActionPerformed(evt);
    }
});
txtTelefono.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyReleased(java.awt.event.KeyEvent evt) {
        txtTelefonoKeyReleased(evt);
    }
    public void keyTyped(java.awt.event.KeyEvent evt) {
        txtTelefonoKeyTyped(evt);
    }
});

btnModificar.setText("Guardar Cambios");
btnModificar.setName(""); // NOI18N
btnModificar.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnModificarActionPerformed(evt);
    }
});

jLabel5.setText("Email");

txtEmail.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        txtEmailActionPerformed(evt);
    }
});
txtEmail.addKeyListener(new java.awt.event.KeyAdapter() {
    public void keyReleased(java.awt.event.KeyEvent evt) {
```

```
            txtEmailKeyReleased(evt);
        }
        public void keyTyped(java.awt.event.KeyEvent evt) {
            txtEmailKeyTyped(evt);
        }
    });

    jLabel6.setText("Horas Maximas:");

    spnHoras.setModel(new javax.swing.SpinnerNumberModel(0, 0, null, 1));

    jLabel3.setText("(Con guion, sin puntos)");

    javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
    jPanel1.setLayout(jPanel1Layout);
    jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGap(30, 30, 30)
                    .addComponent(lblDireccion,
javax.swing.GroupLayout.PREFERRED_SIZE, 90,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(10, 10, 10)
                    .addComponent(txtDireccion,
javax.swing.GroupLayout.PREFERRED_SIZE, 136,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(32, 32, 32)
                    .addComponent(lblApellidoP,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(0, 0, 0)
                    .addComponent(txtApellidoP,
javax.swing.GroupLayout.PREFERRED_SIZE, 136,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGap(28, 28, 28)
                    .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE,
100, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(2, 2, 2)
```

```
                .addComponent(txtTelefono,
javax.swing.GroupLayout.PREFERRED_SIZE, 136,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(30, 30, 30)
                .addComponent(lblApellidoM,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(2, 2, 2)
                .addComponent(txtApellidoM,
javax.swing.GroupLayout.PREFERRED_SIZE, 136,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(28, 28, 28)
                .addComponent(jLabel6)
                .addGap(118, 118, 118)
                .addComponent(spnHoras,
javax.swing.GroupLayout.PREFERRED_SIZE, 46,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(28, 28, 28)
                .addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED_SIZE,
100, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(2, 2, 2)
                .addComponent(txtEmail, javax.swing.GroupLayout.PREFERRED_SIZE,
136, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(104, 104, 104)
                .addComponent(btnModificar,
javax.swing.GroupLayout.PREFERRED_SIZE, 129,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(63, 63, 63)
                .addComponent(btnGuardar,
javax.swing.GroupLayout.PREFERRED_SIZE, 90,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(37, 37, 37)
                .addComponent(cmdSalir,
javax.swing.GroupLayout.PREFERRED_SIZE, 90,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(30, 30, 30)
                .addComponent(lblRut, javax.swing.GroupLayout.PREFERRED_SIZE,
90, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(10, 10, 10)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
```

```
                    .addComponent(jLabel3,
javax.swing.GroupLayout.PREFERRED_SIZE, 183,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(txtRut,
javax.swing.GroupLayout.PREFERRED_SIZE, 136,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(32, 32, 32)
                        .addComponent(lblNombre,
javax.swing.GroupLayout.PREFERRED_SIZE, 100,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(0, 0, 0)
                        .addComponent(txtNombre,
javax.swing.GroupLayout.PREFERRED_SIZE, 136,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGap(168, 168, 168)
                    .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE,
260, javax.swing.GroupLayout.PREFERRED_SIZE)))
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(31, 31, 31)
                .addComponent(jLabel1)
                .addGap(28, 28, 28)
                .addComponent(jLabel3)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addGap(3, 3, 3)
                        .addComponent(lblRut))
                    .addComponent(txtRut, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addGap(6, 6, 6)
                        .addComponent(lblNombre))
                    .addComponent(txtNombre, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
            .addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addComponent(txtDireccion,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(txtApellidoP,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGap(3, 3, 3)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                        .addComponent(lblDireccion)
                        .addComponent(lblApellidoP))))
            .addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addComponent(txtTelefono,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(txtApellidoM,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGap(3, 3, 3)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                        .addComponent(jLabel2)
                        .addComponent(lblApellidoM))))
            .addGap(18, 18, 18)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addComponent(jLabel5)
                .addComponent(txtEmail, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGap(18, 18, 18)
                    .addComponent(jLabel6))
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGap(19, 19, 19)
                    .addComponent(spnHoras,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
            .addGap(41, 41, 41)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                    .addComponent(btnGuardar)
                    .addComponent(btnModificar))
                .addComponent(cmdSalir)))
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE, 612,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(0, 0, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE, 462,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );

        pack();
    }// </editor-fold>

    private void formWindowOpened(java.awt.event.WindowEvent evt) {
```

```
        // TODO add your handling code here:

    }

    private void formWindowClosed(java.awt.event.WindowEvent evt) {
        // TODO add your handling code here:
        padre.setEnabled(true);
        padre.setVisible(true);

    }

    private void txtEmailActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }


    private void btnModificarActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        //UPDATE `Wolfo`.`profesores` SET `nombre` = 'pancha', `apePate` = 'Casas',
`apeMate` = 'Rojas', `direccion` = 'Viña', `telefono` = '44444', `rut` = '19555666-8',
`email` = 'nose@gmail.com', `horas_maximas` = '3' WHERE `profesores`.`rut` =
'11888444-9'
        String rut = txtRut.getText();
        String nom = txtNombre.getText();
        String pate = txtApellidoP.getText();
        String mate = txtApellidoM.getText();
        String dir = txtDireccion.getText();
        String tel = txtTelefono.getText();
        String email = txtEmail.getText();
        int max = (int) spnHoras.getValue();

        if (telOk && emailOk) {
            String consulta = "UPDATE `Wolfo`.`profesores` SET `nombre` = '" + nom + "',
`apePate` = '" + pate + "', `apeMate` = '" + mate + "', `direccion` = '" + dir + "', `telefono`
= '" + tel + "', `email` = '" + email + "', `horas_maximas` = '" + max + "' WHERE
`profesores`.`rut` = '" + rut + "'";
            Conectar conexion = new Conectar();
            conexion.Modificar(consulta);
        } else {
            JOptionPane.showMessageDialog(null, "Telefono o Email Incorrecto",
"ERROR", JOptionPane.WARNING_MESSAGE);
        }


    }
```

```
private void txtTelefonoKeyTyped(java.awt.event.KeyEvent evt) {
    if (txtTelefono.getText().length() == 10) {
        evt.consume();
    }
    // validar solo ingreso de numeros
    char c = evt.getKeyChar();
    if (c < '0' || c > '9') {
        evt.consume();
    }
}

private void txtTelefonoActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void txtDireccionKeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    if (txtDireccion.getText().length() == 50) {
        evt.consume();
    }
    int Chr = evt.getKeyChar();
    if ( Chr == 39) {
        evt.consume();
    }
}

private void txtDireccionActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void txtRutKeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:

    // validar solo ingreso de numeros
    char c = evt.getKeyChar();
    if ((c >= '0' && c <= '9') || (c == '-') || (c == 'k') || (c == 'K')) {
        if (txtRut.getText().length() == 10) {
            evt.consume();
        }
    } else {
        evt.consume();
    }


}
```

```
    private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {

        String rut = txtRut.getText();
        String pate = txtApellidoP.getText();
        String mate = txtApellidoM.getText();
        String dir = txtDireccion.getText();
        String tel = txtTelefono.getText();
        String nom = txtNombre.getText();
        String email = txtEmail.getText();
        String horas = spnHoras.getValue().toString();
        //String cod="";

        if (rut.isEmpty() || nom.isEmpty() || pate.isEmpty() || mate.isEmpty() || dir.isEmpty()
|| tel.isEmpty() || nom.isEmpty() || email.isEmpty()) {
            /*
            dAlerta dale= new dAlerta(a,true);
            dale.setVisible(true);
             */
            JOptionPane.showMessageDialog(null, "Faltan Datos que Ingresar", "Alerta",
JOptionPane.WARNING_MESSAGE);
        } else {
            if (rutOk) {
                if (telOk) {
                    if (emailOk) {
                        Conectar bd = new Conectar();
                        bd.Ingresar("INSERT INTO profesores
(rut,nombre,apePate,apeMate,direccion,telefono,email,horas_maximas ) VALUES ('" +
rut + "', '" + nom + "', '" + pate + "', '" + mate + "', '" + dir + "', '" + tel + "', '" + email + "',
'" + horas + "')");
                        txtApellidoM.setText("");
                        txtApellidoP.setText("");
                        txtDireccion.setText("");
                        txtTelefono.setText("");
                        txtRut.setText("");
                        txtNombre.setText("");
                        txtEmail.setText("");

                        telOk = false;
                        rutOk = false;
                        emailOk = false;

                    } else {
                        JOptionPane.showMessageDialog(null, "Email Incorrecto", "ERROR",
JOptionPane.WARNING_MESSAGE);
                    }
```

```
            } else {
                JOptionPane.showMessageDialog(null, "Telefono Incorrecto", "ERROR",
JOptionPane.WARNING_MESSAGE);
            }

        } else {
            JOptionPane.showMessageDialog(null, "Rut Incorrecto", "ERROR",
JOptionPane.WARNING_MESSAGE);
        }

        //actualizarLista();
    }

}

private void cmdSalirActionPerformed(java.awt.event.ActionEvent evt) {
    padre.setEnabled(true);
    this.setVisible(false);
    try {
        padre.actualizarP("SELECT * FROM profesores");
    } catch (SQLException ex) {
        Logger.getLogger(dProfe.class.getName()).log(Level.SEVERE, null, ex);
    }

}

private void txtNombreKeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    if (txtNombre.getText().length() == 30) {
        evt.consume();
    }
    int Chr = evt.getKeyChar();
    if ( Chr == 39) {
        evt.consume();
    }
}

private void txtNombreActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void txtApellidoMKeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    if (txtApellidoM.getText().length() == 20) {
        evt.consume();
    }
```

```
    int Chr = evt.getKeyChar();
    if ( Chr == 39) {
        evt.consume();
    }
}

private void txtApellidoPKeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    if (txtApellidoP.getText().length() == 20) {
        evt.consume();
    }
    int Chr = evt.getKeyChar();
    if ( Chr == 39) {
        evt.consume();
    }
}

private void txtApellidoPActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void txtApellidoPFocusLost(java.awt.event.FocusEvent evt) {

}

private void txtRutKeyReleased(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:

    if (txtRut.getText().length() >= 9) {
        if (Funciones.validarRutNUEVO(txtRut.getText())) {
            txtRut.setBackground(verde);
            rutOk = true;
        }
    } else {
        rutOk = false;
        txtRut.setBackground(rojo);
        //txtRut.setBorder(javax.swing.BorderFactory.createLineBorder(Color.RED));
    }

}

private void txtRutActionPerformed(java.awt.event.ActionEvent evt) {
}

private void txtTelefonoKeyReleased(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
```

```
        // VALIDAAAR TELEFONO
        //System.out.println((txtTelefono.getText().length()));
        if ((txtTelefono.getText().length()) > 0) {
            telOk = true;
            for (int i = 0; i < txtTelefono.getText().length(); i++) {
                if ((txtTelefono.getText().charAt(i) < '0') || (txtTelefono.getText().charAt(i) >
'9')) {
                    telOk = false;
                }
            }
        }
        if (telOk) {
            txtTelefono.setBackground(blanco);
        } else {
            txtTelefono.setBackground(rojo);
        }
    }

    private void txtEmailKeyReleased(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        System.out.println("VALIDACION = " +
Funciones.validarEmail(txtEmail.getText()));
        if (Funciones.validarEmail(txtEmail.getText())) {
            txtEmail.setBackground(verde);
            emailOk = true;
        } else {
            emailOk = false;
            txtEmail.setBackground(rojo);
            //txtRut.setBorder(javax.swing.BorderFactory.createLineBorder(Color.RED));
        }

    }

    private void txtEmailKeyTyped(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        int Chr = evt.getKeyChar();
        if ( Chr == 39) {
            evt.consume();
        }
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
```

```
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(dProfe.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(dProfe.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(dProfe.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(dProfe.class.getName()).log(java.util.logging.Level.S
EVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the dialog */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                dProfe dialog = new dProfe(new javax.swing.JFrame(), true);
                dialog.addWindowListener(new java.awt.event.WindowAdapter() {
                    @Override
                    public void windowClosing(java.awt.event.WindowEvent e) {
                        System.exit(0);
                    }
                });
                dialog.setVisible(true);
            }
```

```
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnGuardar;
    private javax.swing.JButton btnModificar;
    private javax.swing.JButton cmdSalir;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JLabel lblApellidoM;
    private javax.swing.JLabel lblApellidoP;
    private javax.swing.JLabel lblDireccion;
    private javax.swing.JLabel lblNombre;
    private javax.swing.JLabel lblRut;
    private javax.swing.JSpinner spnHoras;
    private javax.swing.JTextField txtApellidoM;
    private javax.swing.JTextField txtApellidoP;
    private javax.swing.JTextField txtDireccion;
    private javax.swing.JTextField txtEmail;
    private javax.swing.JTextField txtNombre;
    private javax.swing.JTextField txtRut;
    private javax.swing.JTextField txtTelefono;
    // End of variables declaration

}
```

Dispoibilidad Profesor

```
import java.awt.event.ActionListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.ListSelectionModel;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;
```

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
/**
 *
 * @author felipe
 */
public class JDHorarioDisponible extends javax.swing.JDialog {

    String rutGlobal = "";

    /**
     * Creates new form JDHorarioDisponible
     */
    public JDHorarioDisponible(java.awt.Frame parent, boolean modal, boolean editable,
String rut) {
        super(parent, modal);
        initComponents();
        this.setLocationRelativeTo(null);

        rutGlobal = rut;

        Conectar bd = new Conectar();
        ResultSet rs = bd.Consultar("Select nombre ,apePate,apeMate from profesores
where rut = '"+rutGlobal+"'");
        try {
            rs.next();
            lblNombreProfe.setText(rs.getString("nombre")+" "+rs.getString("apePate")+"
"+rs.getString("apeMate"));
        } catch (SQLException ex) {
            Logger.getLogger(JDHorarioDisponible.class.getName()).log(Level.SEVERE,
null, ex);
        }


        tbHorario.setRowHeight(25);
        tbHorario.setShowHorizontalLines(true);
        tbHorario.setShowVerticalLines(true);

        try {
            tbHorario.setModel(Funciones.FormatearBloquesHorario(tbHorario));
            tbHorario.setModel(Funciones.LlenarDisponibilidad(rut, tbHorario));
        } catch (SQLException ex) {
```

```
            Logger.getLogger(JDHorarioDisponible.class.getName()).log(Level.SEVERE,
null, ex);
        }

        tbHorario.addMouseListener(new java.awt.event.MouseAdapter() {
            @Override
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                int row = tbHorario.rowAtPoint(evt.getPoint());
                int col = tbHorario.columnAtPoint(evt.getPoint());

                if (col != 0 && editable == true) {
                    if (tbHorario.getValueAt(row, col) == "Disponible") {
                        tbHorario.setValueAt("", row, col);
                    } else {
                        tbHorario.setValueAt("Disponible", row, col);
                    }
                }
            }
        });

    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jScrollPane1 = new javax.swing.JScrollPane();
        tbHorario = new javax.swing.JTable();
        btnGuardar = new javax.swing.JButton();
        jButton3 = new javax.swing.JButton();
        jLabel1 = new javax.swing.JLabel();
        lblNombreProfe = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);

        tbHorario.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
```

```
            "Hora", "Lunes", "Martes", "Miercoles", "Jueves", "Viernes"
         }
      ) {
         boolean[] canEdit = new boolean [] {
            false, false, false, false, false, false
         };

         public boolean isCellEditable(int rowIndex, int columnIndex) {
            return canEdit [columnIndex];
         }
      });

tbHorario.setSelectionMode(javax.swing.ListSelectionModel.SINGLE_SELECTION);
      tbHorario.getTableHeader().setReorderingAllowed(false);
      jScrollPane1.setViewportView(tbHorario);
      if (tbHorario.getColumnModel().getColumnCount() > 0) {
         tbHorario.getColumnModel().getColumn(0).setResizable(false);
         tbHorario.getColumnModel().getColumn(1).setResizable(false);
         tbHorario.getColumnModel().getColumn(2).setResizable(false);
         tbHorario.getColumnModel().getColumn(3).setResizable(false);
         tbHorario.getColumnModel().getColumn(4).setResizable(false);
         tbHorario.getColumnModel().getColumn(5).setResizable(false);
      }

      btnGuardar.setText("Guardar");
      btnGuardar.addActionListener(new java.awt.event.ActionListener() {
         public void actionPerformed(java.awt.event.ActionEvent evt) {
            btnGuardarActionPerformed(evt);
         }
      });

      jButton3.setText("Volver");
      jButton3.addActionListener(new java.awt.event.ActionListener() {
         public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton3ActionPerformed(evt);
         }
      });

      jLabel1.setText("Horario de Disponiblidad para profesor :");

      javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
      jPanel1.setLayout(jPanel1Layout);
      jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
         .addGroup(jPanel1Layout.createSequentialGroup()
```

```
.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addContainerGap()
                    .addComponent(jScrollPane1))
                .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                        .addGroup(jPanel1Layout.createSequentialGroup()
                            .addGap(143, 143, 143)
                            .addComponent(btnGuardar)
                            .addGap(186, 186, 186)
                            .addComponent(jButton3))
                        .addGroup(jPanel1Layout.createSequentialGroup()
                            .addGap(51, 51, 51)
                            .addComponent(jLabel1)
                            .addGap(18, 18, 18)
                            .addComponent(lblNombreProfe,
javax.swing.GroupLayout.PREFERRED_SIZE, 218,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                    .addGap(0, 374, Short.MAX_VALUE)))
            .addContainerGap())
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(10, 10, 10)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING, false)
                    .addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(lblNombreProfe,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
278, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(78, 78, 78)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
```

```
                .addComponent(btnGuardar)
                .addComponent(jButton3))
            .addContainerGap(19, Short.MAX_VALUE))
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 862, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(0, 0, Short.MAX_VALUE)
                    .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(0, 0, Short.MAX_VALUE)))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 433, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(0, 0, Short.MAX_VALUE)
                    .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(0, 0, Short.MAX_VALUE)))
        );

        pack();
    }// </editor-fold>

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        this.setVisible(false);
    }

    private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        Conectar bd = new Conectar();
        bd.EliminarSinPreguntar("DELETE FROM horario_disponibilidad_profesor where
rut ='" + rutGlobal + "'");
```

```
        for (int i = 1; i < tbHorario.getColumnCount(); i++) {
            for (int j = 0; j < tbHorario.getRowCount(); j++) {
                if (tbHorario.getValueAt(j, i).toString().equals("Disponible")) {
                    int dia = i;
                    int numBloque = j + 1;
                    bd.IngresarSinPreguntar("INSERT INTO horario_disponibilidad_profesor "
                            + "(`dia`, `numero_bloque`, `rut`) VALUES ('" + dia + "','" +
numBloque + "','" + rutGlobal + "')");
                }
            }

        }
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(JDHorarioDisponible.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(JDHorarioDisponible.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(JDHorarioDisponible.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
```

```
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(JDHorarioDisponible.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the dialog */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                JDHorarioDisponible dialog = new JDHorarioDisponible(new
javax.swing.JFrame(), true, true, "");
                dialog.addWindowListener(new java.awt.event.WindowAdapter() {
                    @Override
                    public void windowClosing(java.awt.event.WindowEvent e) {
                        System.exit(0);
                    }
                });
                dialog.setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnGuardar;
    private javax.swing.JButton jButton3;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel lblNombreProfe;
    private javax.swing.JTable tbHorario;
    // End of variables declaration
}
```

Gestionar Bloques de Horario

```
import java.sql.Date;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Time;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.logging.Level;
```

```
import java.util.logging.Logger;
import javax.swing.JSpinner;
import javax.swing.ListSelectionModel;
import javax.swing.SpinnerDateModel;
import javax.swing.event.ChangeEvent;
import javax.swing.event.ChangeListener;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;
import javax.swing.table.DefaultTableModel;
import javax.swing.text.DateFormatter;

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
/**
 *
 * @author felipe
 */
public class JPBloque extends javax.swing.JPanel {

    /**
     * Creates new form JPBloque
     */
    SimpleDateFormat format = new SimpleDateFormat("HH:mm");

    public JPBloque() {
        initComponents();

        try {
            this.F5();
        } catch (SQLException ex) {
            Logger.getLogger(JPBloque.class.getName()).log(Level.SEVERE, null, ex);
        }

        Calendar calendar = Calendar.getInstance();
        calendar.set(Calendar.HOUR_OF_DAY, 24); // 24 == 12 PM == 00:00:00
        calendar.set(Calendar.MINUTE, 0);
        calendar.set(Calendar.SECOND, 0);

        //para setear el formato al  SPNHORAINICIO
        SpinnerDateModel model = new SpinnerDateModel();
        model.setValue(calendar.getTime());
        spnHoraInicio.setModel(model);
        JSpinner.DateEditor editor = new JSpinner.DateEditor(spnHoraInicio, "HH:mm");
```

```
        DateFormatter formatter = (DateFormatter) editor.getTextField().getFormatter();
        formatter.setAllowsInvalid(false); // this makes what you want
        formatter.setOverwriteMode(true);
        spnHoraInicio.setEditor(editor);

        //para setear el formato al  SPNHORATERMINO
        SpinnerDateModel model2 = new SpinnerDateModel();
        model2.setValue(calendar.getTime());
        spnHoraTermino.setModel(model2);
        JSpinner.DateEditor editor2 = new JSpinner.DateEditor(spnHoraTermino,
"HH:mm");
        DateFormatter formatter2 = (DateFormatter) editor2.getTextField().getFormatter();
        formatter2.setAllowsInvalid(false); // this makes what you want
        formatter2.setOverwriteMode(true);
        spnHoraTermino.setEditor(editor2);

        // Tipo de dato para setear el value
        try {
            spnHoraInicio.setValue(format.parseObject("00:00")); // e.g. input 16:45
        } catch (ParseException ex) {
            Logger.getLogger(JPBloque.class.getName()).log(Level.SEVERE, null, ex);
        }

        Calendar t1 = Calendar.getInstance();
        t1.set(t1.HOUR_OF_DAY, 17);
        t1.set(t1.MINUTE, 40);

        Calendar t2 = Calendar.getInstance();
        t2.set(t2.HOUR_OF_DAY, 20);
        t2.set(t2.MINUTE, 30);
        /*
        Calendar t2 = (Calendar) format.parseObject("20:00");

        if(t1.compareTo(t2)>0){
            spnHoras.getModel().setValue(t1.getTime());
        }else{
            spnHoras.getModel().setValue(t2.getTime());
        }
         */
 /*
        spnHoraInicio.addChangeListener(new ChangeListener() {
            @Override
            public void stateChanged(ChangeEvent e) {
                String hora = new
SimpleDateFormat("HH:mm").format(spnHoraInicio.getValue());
                try {
```

```
                spnHoraInicio.setValue(format.parseObject(hora));
            } catch (ParseException ex) {
                Logger.getLogger(JPBloque.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    });
     */
    ListSelectionModel modelTabla = tbBloques.getSelectionModel();
    modelTabla.addListSelectionListener(new ListSelectionListener() {
        @Override
        public void valueChanged(ListSelectionEvent e) {
            try {
                if (tbBloques.getSelectedRow() != -1) {

spnHoraInicio.setValue(format.parseObject(tbBloques.getModel().getValueAt(tbBloques
.getSelectedRow(), 1).toString()));

spnHoraTermino.setValue(format.parseObject(tbBloques.getModel().getValueAt(tbBloq
ues.getSelectedRow(), 2).toString()));
                } else {
                    spnHoraInicio.setValue(format.parseObject("00:00"));
                    spnHoraTermino.setValue(format.parseObject("00:00"));

                }
            } catch (ParseException ex) {
                Logger.getLogger(JPBloque.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
    );
}

/**
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    jScrollPane1 = new javax.swing.JScrollPane();
    tbBloques = new javax.swing.JTable();
    spnHoraInicio = new javax.swing.JSpinner();
    spnHoraTermino = new javax.swing.JSpinner();
    btnAgregar = new javax.swing.JButton();
```

```
        btnEliminar = new javax.swing.JButton();
        lblError = new javax.swing.JLabel();
        btnGuardar = new javax.swing.JButton();
        btnEditar = new javax.swing.JButton();

        tbBloques.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "Numero Bloque", "Hora Inicio", "Hora Término"
            }
        ) {
            boolean[] canEdit = new boolean [] {
                false, false, false
            };

            public boolean isCellEditable(int rowIndex, int columnIndex) {
                return canEdit [columnIndex];
            }
        });
        tbBloques.getTableHeader().setReorderingAllowed(false);
        jScrollPane1.setViewportView(tbBloques);
        if (tbBloques.getColumnModel().getColumnCount() > 0) {
            tbBloques.getColumnModel().getColumn(0).setResizable(false);
            tbBloques.getColumnModel().getColumn(1).setResizable(false);
            tbBloques.getColumnModel().getColumn(2).setResizable(false);
        }

        spnHoraInicio.setModel(new javax.swing.SpinnerDateModel(new
java.util.Date(1538256780000L), new java.util.Date(1538256780000L), new
java.util.Date(1542749603182L), java.util.Calendar.HOUR));

        spnHoraTermino.setModel(new javax.swing.SpinnerDateModel(new
java.util.Date(1538256780000L), new java.util.Date(1538256780000L), new
java.util.Date(1542749603182L), java.util.Calendar.HOUR));

        btnAgregar.setText("Agregar ");
        btnAgregar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnAgregarActionPerformed(evt);
            }
        });

        btnEliminar.setText("Eliminar ");
        btnEliminar.addActionListener(new java.awt.event.ActionListener() {
```

```
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            btnEliminarActionPerformed(evt);
        }
    });

    lblError.setForeground(new java.awt.Color(255, 102, 102));

    btnGuardar.setText("Guardar Cambios");
    btnGuardar.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            btnGuardarActionPerformed(evt);
        }
    });

    btnEditar.setText("Editar");
    btnEditar.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            btnEditarActionPerformed(evt);
        }
    });

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);
    this.setLayout(layout);
    layout.setHorizontalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(61, 61, 61)
                    .addComponent(lblError, javax.swing.GroupLayout.PREFERRED_SIZE,
497, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(layout.createSequentialGroup()
                    .addContainerGap()
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 606, Short.MAX_VALUE)))
            .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(btnEditar, javax.swing.GroupLayout.PREFERRED_SIZE,
115, javax.swing.GroupLayout.PREFERRED_SIZE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(spnHoraInicio,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE,
```

```
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(spnHoraTermino,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addComponent(btnAgregar, javax.swing.GroupLayout.PREFERRED_SIZE,
115, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(btnEliminar,
javax.swing.GroupLayout.PREFERRED_SIZE, 115,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(btnGuardar))
            .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(62, 62, 62)
                    .addComponent(spnHoraInicio,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(25, 25, 25)
                    .addComponent(spnHoraTermino,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(18, 18, 18)
                    .addComponent(btnEditar)
                    .addGap(18, 18, 18)
                    .addComponent(btnAgregar)
                    .addGap(15, 15, 15)
                    .addComponent(btnEliminar)
                    .addGap(18, 18, 18)
                    .addComponent(btnGuardar))
                .addGroup(layout.createSequentialGroup()
                    .addGap(27, 27, 27)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 275, Short.MAX_VALUE)))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(lblError, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
```

```
        );
    }// </editor-fold>

    private void btnAgregarActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            // TODO add your handling code here:
            SimpleDateFormat formato = new SimpleDateFormat("HH:mm");
            DefaultTableModel modelo = (DefaultTableModel) tbBloques.getModel();

            String hInicio = new
SimpleDateFormat("HH:mm").format(spnHoraInicio.getValue());
            String hTermino = new
SimpleDateFormat("HH:mm").format(spnHoraTermino.getValue());

            java.util.Date tini = formato.parse(hInicio);
            java.util.Date ttermi = formato.parse(hTermino);
            boolean sinDatos = false;
            java.util.Date horaTabla = new java.util.Date();
            if (tbBloques.getModel().getRowCount() > 0) {
                horaTabla =
formato.parse(tbBloques.getModel().getValueAt(modelo.getRowCount() - 1,
2).toString());
            } else {
                sinDatos = true;
            }

            if (tini.compareTo(ttermi) == -1) {
                if (!sinDatos) {
                    if (horaTabla.compareTo(tini) <= 0) {
                        int num;
                        num = (int) modelo.getValueAt(modelo.getRowCount() - 1, 0);
                        num++;
                        modelo.addRow(new Object[]{num, hInicio, hTermino});
                        tbBloques.setModel(modelo);
                        lblError.setText("");
                        spnHoraInicio.setValue(format.parseObject(hTermino));

                    } else {
                        lblError.setText("Hora de inicio menor que hora de termino del ultimo
bloque");
                        System.out.println("Hora de inicio menor que hora de termino del ultimo
bloque");
                    }
                } else {
                    int num = 0;
```

```
                num++;
                modelo.addRow(new Object[]{num, hInicio, hTermino});
                tbBloques.setModel(modelo);
                lblError.setText("");
                spnHoraInicio.setValue(format.parseObject(hTermino));

            }
        } else {
            lblError.setText("ERROR hora de inicio mayor o igual que termino");
            System.out.println("ERROR hora de inicio mayor o igual que termino");

        }

        /*
        for (int i = 0; i < modelo.getRowCount(); i++) {
        if (modelo.getValueAt(i, 0).toString().equals(cod)) {
        modelo.removeRow(i);
        }
        }
         */
    } catch (ParseException ex) {
        Logger.getLogger(JPBloque.class
                .getName()).log(Level.SEVERE, null, ex);
    }

}

private void btnEliminarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    if (tbBloques.getSelectedRow() != -1) {
        DefaultTableModel modelo = (DefaultTableModel) tbBloques.getModel();
        for (int i = tbBloques.getSelectedRow() + 1; i <
tbBloques.getModel().getRowCount(); i++) {
            modelo.setValueAt(Integer.parseInt(modelo.getValueAt(i, 0).toString()) - 1, i,
0);
            tbBloques.setModel(modelo);
        }
        modelo.removeRow(tbBloques.getSelectedRow());
        tbBloques.setModel(modelo);
    }


}

private void btnEditarActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        // TODO add your handling code here:
        if (tbBloques.getSelectedRow() != -1) {
            try {

                SimpleDateFormat formato = new SimpleDateFormat("HH:mm");
                DefaultTableModel modelo = (DefaultTableModel) tbBloques.getModel();
                String hInicio = new
SimpleDateFormat("HH:mm").format(spnHoraInicio.getValue());
                String hTermino = new
SimpleDateFormat("HH:mm").format(spnHoraTermino.getValue());

                java.util.Date tini = formato.parse(hInicio);
                java.util.Date ttermi = formato.parse(hTermino);
                // de tabla
                java.util.Date tTerminoAnterio;

                java.util.Date tInicioSiguiente;
                if (tbBloques.getSelectedRow() == 0) {
                    tTerminoAnterio = formato.parse("00:00");
                } else {
                    tTerminoAnterio = formato.parse((String)
tbBloques.getModel().getValueAt(tbBloques.getSelectedRow() - 1, 2));
                }
                if (tbBloques.getSelectedRow() == tbBloques.getRowCount() - 1) {
                    tInicioSiguiente = formato.parse("23:59");
                } else {
                    tInicioSiguiente = formato.parse((String)
tbBloques.getModel().getValueAt(tbBloques.getSelectedRow() + 1, 1));

                }

                if (tini.compareTo(tTerminoAnterio) >= 0 &&
ttermi.compareTo(tInicioSiguiente) <= 0) {
                    if (tini.compareTo(ttermi) == -1) {
                        modelo.setValueAt(hInicio, tbBloques.getSelectedRow(), 1);
                        modelo.setValueAt(hTermino, tbBloques.getSelectedRow(), 2);
                    } else {
                        lblError.setText("ERROR hora de inicio mayor o igual que termino");
                        System.out.println("ERROR hora de inicio mayor o igual que termino");
                    }
                } else {
                    lblError.setText("Error de Horas");
                }
            } catch (ParseException ex) {
                Logger.getLogger(JPBloque.class
                        .getName()).log(Level.SEVERE, null, ex);
```

```
        }

    }
}

private void btnGuardarActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    DefaultTableModel model = (DefaultTableModel) tbBloques.getModel();
    Conectar bd = new Conectar();
    ResultSet rs = bd.Consultar("SELECT * from formato_bloque");
    int cantRS = 0;

    try {
        while (rs.next()) {
            cantRS = cantRS + 1;
        }
        System.out.println("cantrs =" + cantRS);
        rs.beforeFirst();
    } catch (SQLException ex) {
        Logger.getLogger(JPBloque.class.getName()).log(Level.SEVERE, null, ex);
    }

    //bd.EliminarSinPreguntar("delete from formato_bloque");
    if (model.getRowCount() > 0) {
        for (int i = 0; i < model.getRowCount(); i++) {

            if (i < cantRS) {

                String num = model.getValueAt(i, 0).toString();
                String hIni = model.getValueAt(i, 1).toString();
                String hTer = model.getValueAt(i, 2).toString();
                int numBloque = i + 1;
                bd.ModificarSinPregutar("UPDATE `formato_bloque` SET `horaInicio` = '"
+ hIni + "', `horaTermino` = '" + hTer + "' "
                        + "WHERE `formato_bloque`.`numeroBloque` ='" + numBloque + "'");
                //bd.IngresarSinPreguntar("INSERT INTO formato_bloque
(numeroBloque,horaInicio,horaTermino)  "
                //        + "VALUES ('" + num + "','" + hIni + "','" + hTer + "')");
            }
            if (i >= cantRS) {
                //String num = model.getValueAt(i, 0).toString();
                String hIni = model.getValueAt(i, 1).toString();
                String hTer = model.getValueAt(i, 2).toString();
                int numBloque = i + 1;
                bd.IngresarSinPreguntar("INSERT INTO formato_bloque
(numeroBloque,horaInicio,horaTermino)  "
```

```
                    + "VALUES ('" + numBloque + "','" + hIni + "','" + hTer + "')");
            }
          }
          if (cantRS > model.getRowCount()) {
            int diferencia = cantRS - model.getRowCount();
            for (int i = model.getRowCount() + 1; i <= cantRS; i++) {
              bd.ModificarSinPregutar("UPDATE `formato_bloque` SET `horaInicio` =
NULL, `horaTermino` = NULL "
                    + "WHERE `formato_bloque`.`numeroBloque` ='" + i + "'");
            }
          }
        }
        lblError.setText("Guardado Exitoso");
    }

    public void F5() throws SQLException {
        Funciones.mostrar("SELECT
numeroBloque,substring(horaInicio,1,5),substring(horaTermino,1,5) FROM
formato_bloque WHERE horaInicio IS NOT NULL", tbBloques);
        tbBloques.setRowSorter(null);
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnAgregar;
    private javax.swing.JButton btnEditar;
    private javax.swing.JButton btnEliminar;
    private javax.swing.JButton btnGuardar;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel lblError;
    private javax.swing.JSpinner spnHoraInicio;
    private javax.swing.JSpinner spnHoraTermino;
    private javax.swing.JTable tbBloques;
    // End of variables declaration
}
```

Horario por Cursos

```
import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.ParseException;
import java.util.Vector;
```

```
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.DefaultComboBoxModel;
import javax.swing.DefaultListModel;
import javax.swing.JList;
import javax.swing.JTable;
import javax.swing.ListSelectionModel;
import javax.swing.table.DefaultTableModel;

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
/**
 *
 * @author felipe
 */
public class JPHorarioCursos extends javax.swing.JPanel {

    /**
     * Creates new form JPHorarioCursos
     */
    JList lista = new JList();
    JTable tabla = new JTable();
    String codigoCursoGlobal = "";

    Color rojo = new Color(161, 0, 0);
    Color verde = new Color(0, 161, 0);
    Color blanco = new Color(255, 255, 255);
    public JPHorarioCursos() {
        initComponents();
        DefaultTableModel dtm = new DefaultTableModel() {
            @Override
            public boolean isCellEditable(int row, int column) {
                //all cells false
                return false;
            }
        };

        tbHorario.getColumn("Lunes").setCellRenderer(new JListRenderer());
        tbHorario.getColumn("Martes").setCellRenderer(new JListRenderer());
        tbHorario.getColumn("Miercoles").setCellRenderer(new JListRenderer());
        tbHorario.getColumn("Jueves").setCellRenderer(new JListRenderer());
```

```
        tbHorario.getColumn("Viernes").setCellRenderer(new JListRenderer());
        tbHorario.setCellEditor(new JListEditor());
        /*
        final MyListModel dlm2 = new MyListModel(new Object[]{"aaa", "bbb"});

        MyListModel[] datos = {dlm2, dlm2, dlm2, dlm2, dlm2, dlm2}; // Cantidad de
columnas de la tabla
        DefaultTableModel model = (DefaultTableModel) tbHorario.getModel();
        model.addRow(datos);
        tbHorario.setModel(model);
         */
        //ProfesoresCmb profe = (ProfesoresCmb) cmbProfes.getSelectedItem();
        //String rut = profe.getRut();

        try {
            tbHorario.setModel(Funciones.FormatearBloquesHorarioConMLM(tbHorario));

        } catch (SQLException ex) {
            Logger.getLogger(JPHorarioProfesores.class.getName()).log(Level.SEVERE,
null, ex);
        }
        //lista.setModel(model);
        tbHorario.setRowHeight(85);
        tbHorario.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
        tbHorario.setCellSelectionEnabled(true);
        //tbHorario.getModel().setValueAt(jList1.getModel(), 1, 1);
        //tbHorario.getModel().setValueAt( v.getText(), 2, 2);

        v.setVisible(false);
        k.setVisible(false);
        n.setVisible(false);
        tbHorario.setShowGrid(true);

        try {
            llenarComboProfesor();
            llenarComboSala();
            llenarComboCurso();
            llenarComboAsig();
        } catch (SQLException ex) {
            Logger.getLogger(JPHorarioProfesores.class.getName()).log(Level.SEVERE,
null, ex);
        }
        cmbCurso.setSelectedIndex(1);
        CursosCmb curso = (CursosCmb) cmbCurso.getSelectedItem();
```

```
        codigoCursoGlobal = curso.getCodigoCurso();

        try {
            F5("select * from horario where codigoCurso ='" + codigoCursoGlobal + "'");
        } catch (SQLException ex) {
            Logger.getLogger(JPHorarioProfesores.class.getName()).log(Level.SEVERE,
null, ex);
        }

        cmbCurso.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                CursosCmb curso = (CursosCmb) cmbCurso.getSelectedItem();
                codigoCursoGlobal = curso.getCodigoCurso();

                try {
                    F5("select * from horario where codigoCurso ='" + codigoCursoGlobal + "'");
                } catch (SQLException ex) {

Logger.getLogger(JPHorarioProfesores.class.getName()).log(Level.SEVERE, null, ex);
                }
            }
        });
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jScrollPane7 = new javax.swing.JScrollPane();
        n = new javax.swing.JTextArea();
        jScrollPane8 = new javax.swing.JScrollPane();
        k = new javax.swing.JTextArea();
        jScrollPane9 = new javax.swing.JScrollPane();
        v = new javax.swing.JTextArea();
        jLabel6 = new javax.swing.JLabel();
        jLabel7 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        cmbAsginaturas = new javax.swing.JComboBox<>();
        cmbCurso = new javax.swing.JComboBox<>();
```

```
        cmbSala = new javax.swing.JComboBox<>();
        btnBorrar = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        tbHorario = new javax.swing.JTable();
        btnInsertar = new javax.swing.JButton();
        cmbProfes = new javax.swing.JComboBox<>();
        lblError = new javax.swing.JLabel();
        jLabel1 = new javax.swing.JLabel();

        n.setColumns(20);
        n.setRows(5);
        n.setText("Oscar Perez\n1roA\nMatematica\n201\n");
        jScrollPane7.setViewportView(n);

        k.setColumns(20);
        k.setRows(5);
        k.setText("Camila Rojas\n7moC\nMatematica\n101\n");
        jScrollPane8.setViewportView(k);

        v.setColumns(20);
        v.setRows(5);
        v.setText("Ana Zuñiga \n8voA\nMatematica\n106\n");
        jScrollPane9.setViewportView(v);

        jLabel6.setText("Asignatura :");

        jLabel7.setText("Profesor :");

        jLabel8.setText("Sala : ");

        cmbAsginaturas.setModel(new javax.swing.DefaultComboBoxModel<>(new String[]
{ "Item 1", "Item 2", "Item 3", "Item 4" }));

        cmbCurso.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] {
"Item 1", "Item 2", "Item 3", "Item 4" }));

        cmbSala.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "Item
1", "Item 2", "Item 3", "Item 4" }));

        btnBorrar.setText("Borrar");
        btnBorrar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnBorrarActionPerformed(evt);
            }
```

```
        });

        tbHorario.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {

            },
            new String [] {
                "Hora", "Lunes", "Martes", "Miercoles", "Jueves", "Viernes"
            }
        ) {
            boolean[] canEdit = new boolean [] {
                false, false, false, false, false, false
            };

            public boolean isCellEditable(int rowIndex, int columnIndex) {
                return canEdit [columnIndex];
            }
        });
        tbHorario.setColumnSelectionAllowed(true);
        tbHorario.setRowSelectionAllowed(false);

tbHorario.setSelectionMode(javax.swing.ListSelectionModel.SINGLE_SELECTION);
        tbHorario.getTableHeader().setReorderingAllowed(false);
        jScrollPane1.setViewportView(tbHorario);

tbHorario.getColumnModel().getSelectionModel().setSelectionMode(javax.swing.ListSel
ectionModel.SINGLE_SELECTION);
        if (tbHorario.getColumnModel().getColumnCount() > 0) {
            tbHorario.getColumnModel().getColumn(0).setResizable(false);
            tbHorario.getColumnModel().getColumn(1).setResizable(false);
            tbHorario.getColumnModel().getColumn(2).setResizable(false);
            tbHorario.getColumnModel().getColumn(3).setResizable(false);
            tbHorario.getColumnModel().getColumn(4).setResizable(false);
            tbHorario.getColumnModel().getColumn(5).setResizable(false);
        }

        btnInsertar.setText("Insertar");
        btnInsertar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnInsertarActionPerformed(evt);
            }
        });

        lblError.setFont(new java.awt.Font("Tahoma", 1, 18)); // NOI18N
```

```
        lblError.setForeground(new java.awt.Color(255, 0, 0));

        jLabel1.setText("Curso :");

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(0, 0, Short.MAX_VALUE)
                .addComponent(jScrollPane7, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jScrollPane8, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(49, 49, 49)
                .addComponent(jScrollPane9, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(247, 247, 247))
            .addGroup(layout.createSequentialGroup()
                .addGap(35, 35, 35)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel6, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(cmbAsginaturas, 0, 168, Short.MAX_VALUE))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel1)
                        .addGap(27, 27, 27)
                        .addComponent(cmbCurso,
javax.swing.GroupLayout.PREFERRED_SIZE, 124,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(0, 52, Short.MAX_VALUE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                .addComponent(jLabel7, javax.swing.GroupLayout.DEFAULT_SIZE, 50,
Short.MAX_VALUE)
                .addGap(18, 18, 18)
```

```
            .addComponent(cmbProfes, javax.swing.GroupLayout.PREFERRED_SIZE,
167, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(46, 46, 46)
            .addComponent(jLabel8)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(cmbSala, javax.swing.GroupLayout.PREFERRED_SIZE,
137, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(btnInsertar, javax.swing.GroupLayout.PREFERRED_SIZE,
119, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap())
         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
            .addComponent(btnBorrar, javax.swing.GroupLayout.PREFERRED_SIZE,
119, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap())

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
               .addContainerGap()
               .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT_SIZE,
846, Short.MAX_VALUE)
               .addGap(18, 18, 18))
            .addGroup(layout.createSequentialGroup()
               .addGap(31, 31, 31)
               .addComponent(lblError, javax.swing.GroupLayout.PREFERRED_SIZE,
611, javax.swing.GroupLayout.PREFERRED_SIZE)
               .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)))
      );
      layout.setVerticalGroup(
         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
         .addGroup(layout.createSequentialGroup()
            .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
               .addComponent(cmbCurso, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
               .addComponent(jLabel1))
            .addGap(384, 384, 384)
```

```
            .addComponent(jScrollPane9, javax.swing.GroupLayout.PREFERRED_SIZE,
0, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
false)
                .addComponent(jScrollPane8,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, 0, Short.MAX_VALUE)
                .addComponent(jScrollPane7,
javax.swing.GroupLayout.PREFERRED_SIZE, 0,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
66, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                        .addComponent(jLabel6)
                        .addComponent(cmbAsginaturas,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(jLabel7)
                        .addComponent(cmbProfes,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(jLabel8)
                        .addComponent(cmbSala,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGap(48, 48, 48))
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                    .addComponent(btnInsertar)
                    .addGap(22, 22, 22)
                    .addComponent(btnBorrar,
javax.swing.GroupLayout.PREFERRED_SIZE, 23,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addContainerGap())))
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(37, 37, 37)
                .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(lblError, javax.swing.GroupLayout.PREFERRED_SIZE, 22,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(74, Short.MAX_VALUE)))
        );

        getAccessibleContext().setAccessibleDescription("");
    }// </editor-fold>

    public void F5(String sql) throws SQLException {
        tbHorario.setModel(Funciones.LlenarHorario(sql, tbHorario));

    }

    private void llenarComboAsig() throws SQLException {

        Conectar bd = new Conectar();
        ResultSet rs = bd.Consultar("SELECT * FROM asignaturas");
        Vector<AsignaturasCmb> asignaturas = new Vector<AsignaturasCmb>();
        while (rs.next()) {

            AsignaturasCmb asigAux = new AsignaturasCmb(rs.getString("codigoAsig"),
                rs.getString("nombre"));
            asignaturas.addElement(asigAux);
        }
        if (!asignaturas.isEmpty()) {
            DefaultComboBoxModel modelo = new DefaultComboBoxModel();
            for (int i = 0; i < asignaturas.size(); i++) {
                modelo.addElement(asignaturas.get(i));
            }
            cmbAsginaturas.setModel(modelo);
        }
    }

    private void llenarComboSala() throws SQLException {
```

```
    Conectar bd = new Conectar();
    ResultSet rs = bd.Consultar("SELECT * FROM salas");
    Vector<SalasCmb> salas = new Vector<SalasCmb>();
    while (rs.next()) {
        SalasCmb salaAux = new SalasCmb(rs.getString("codigoSala"),
                rs.getString("nombreSala"), rs.getInt("alumnosMax"));
        salas.addElement(salaAux);
    }
    if (!salas.isEmpty()) {
        DefaultComboBoxModel modelo = new DefaultComboBoxModel();
        for (int i = 0; i < salas.size(); i++) {
            modelo.addElement(salas.get(i));
        }
        cmbSala.setModel(modelo);
    }
}

private void llenarComboCurso() throws SQLException {
    Conectar bd = new Conectar();
    ResultSet rs = bd.Consultar("SELECT * FROM cursos");
    Vector<CursosCmb> cursos = new Vector<CursosCmb>();
    while (rs.next()) {
        CursosCmb salaAux = new CursosCmb(rs.getString("codigoCurso"),
                rs.getString("codigoSala"));
        cursos.addElement(salaAux);
    }
    if (!cursos.isEmpty()) {
        DefaultComboBoxModel modelo = new DefaultComboBoxModel();
        for (int i = 0; i < cursos.size(); i++) {
            modelo.addElement(cursos.get(i));
        }
        cmbCurso.setModel(modelo);
    }
}

private void llenarComboProfesor() throws SQLException {
    Conectar bd = new Conectar();
    ResultSet rs = bd.Consultar("SELECT * FROM profesores");
    Vector<ProfesoresCmb> profesores = new Vector<ProfesoresCmb>();
    while (rs.next()) {
        ProfesoresCmb profeAux = new ProfesoresCmb(rs.getString("rut"),
                rs.getString("nombre"), rs.getString("apePate"));
        profesores.addElement(profeAux);
    }
```

```
        if (!profesores.isEmpty()) {
            DefaultComboBoxModel modelo = new DefaultComboBoxModel();
            for (int i = 0; i < profesores.size(); i++) {
                modelo.addElement(profesores.get(i));
            }
            cmbProfes.setModel(modelo);
        }
    }
    private void btnBorrarActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        int row = tbHorario.getSelectedRow();
        int col = tbHorario.getSelectedColumn();
        int numeroBloque = row + 1;
        MyListModel mlmDispo = new MyListModel(new Object[]{""});
        tbHorario.getModel().setValueAt(mlmDispo, row, col);
        Conectar bd = new Conectar();
        bd.EliminarSinPreguntar("DELETE FROM horario where codigoCurso ='" +
codigoCursoGlobal + "' and dia =" + col + " and numeroBloque =" + numeroBloque);

    }

    private void btnInsertarActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        int row = tbHorario.getSelectedRow();
        int col = tbHorario.getSelectedColumn();
        int numBloque = row + 1;
        ProfesoresCmb profe = (ProfesoresCmb) cmbProfes.getSelectedItem();
        CursosCmb curso = (CursosCmb) cmbCurso.getSelectedItem();
        AsignaturasCmb asignatura = (AsignaturasCmb)
cmbAsginaturas.getSelectedItem();
        SalasCmb sala = (SalasCmb) cmbSala.getSelectedItem();

        String rut = profe.getRut();
        String codCurso = curso.getCodigoCurso();
        String codAsig = asignatura.getCodigoAsig();
        String codSala = sala.getCodigoSala();
        lblError.setForeground(rojo);
        try {
            if (Funciones.ValidarInsertHoraioRUT(col, numBloque, rut)) {
                if (Funciones.ValidarInsertHoraioCURSO(col, numBloque, codCurso)) {
                    if (Funciones.ValidarInsertHoraioSALA(col, numBloque, codSala)) {
                        if (Funciones.ValidarDisponibilidadHorario(col, numBloque, rut)) {
                            if (Funciones.Validarhoras(rut)) {
                                DefaultListModel<Object> model = new DefaultListModel<>();
```

```
                        //DefaultListModel model = new DefaultListModel();
                        model.addElement(profe);
                        model.addElement(curso);
                        model.addElement(sala);
                        model.addElement(asignatura);

                        JList<Object> lista = new JList<>(model);
                        lista.setLayoutOrientation(JList.VERTICAL);

                        MyListModel dlm1 = new MyListModel(new Object[]{profe, curso,
asignatura, sala});
                        tbHorario.getModel().setValueAt(dlm1, row, col);
                        Conectar bd = new Conectar();
                        bd.IngresarSinPreguntar("INSERT INTO `horario` (`numeroBloque`,
`dia`, `rut`, `codigoCurso`, "
                                + "`codigoAsig`, `codigoSala`, `anno`)"
                                + " VALUES ('" + numBloque + "', '" + col + "', '" + rut + "', '" +
curso.getCodigoCurso() + "', '" + asignatura.getCodigoAsig() + "', '" +
sala.getCodigoSala() + "', '2018');");
                        lblError.setForeground(verde);
                        lblError.setText("INGRESADO CON EXITO");
                    } else {
                        lblError.setText("Profesor : " + rut + " Supera su limite de horas
maximas");
                    }
                } else {
                    lblError.setText("Profesor : " + rut + " No tiene disponibilidad en este
Bloque");
                }
            } else {
                lblError.setText("Sala : " + codSala + " ya esta siendo ocupada  en ese
bloque");
            }
        } else {
            lblError.setText("Curso : " + codCurso + " ya esta siendo ocupado  en ese
bloque");
        }
    } else {
        lblError.setText("Profesor : " + rut + " ya esta siendo ocupado  en ese
bloque");
    }
} catch (SQLException ex) {
    Logger.getLogger(JPHorarioProfesores.class.getName()).log(Level.SEVERE,
null, ex);
```

```
        } catch (ParseException ex) {
            Logger.getLogger(JPHorarioProfesores.class.getName()).log(Level.SEVERE,
null, ex);
        }

    }


    // Variables declaration - do not modify
    private javax.swing.JButton btnBorrar;
    private javax.swing.JButton btnInsertar;
    private javax.swing.JComboBox<String> cmbAsginaturas;
    private javax.swing.JComboBox<String> cmbCurso;
    private javax.swing.JComboBox<String> cmbProfes;
    private javax.swing.JComboBox<String> cmbSala;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane7;
    private javax.swing.JScrollPane jScrollPane8;
    private javax.swing.JScrollPane jScrollPane9;
    private javax.swing.JTextArea k;
    private javax.swing.JLabel lblError;
    private javax.swing.JTextArea n;
    private javax.swing.JTable tbHorario;
    private javax.swing.JTextArea v;
    // End of variables declaration
}
```