DISEÑO Y PROTOTIPO DE UNA HERRAMIENTA PARA QUE LOS DOCENTES ACTUALICEN LAS NOTAS DE LOS ESTUDIANTES DE FORMA SEGURA.

JUAN JOSE PIZARRO MARQUEZ

FREDYZ NIETO HUERTAS

UNIVERSIDAD TECNOLÓGICA DE BOLIVAR

FACULTAD DE INGENIERIA DE SISTEMAS

CARTAGENA DE INDIAS

2007

DISEÑO DE UN PROTOTIPO DE SOFTWARE DE NOTAS PARA DOCENTES UNIVERSITARIOS

JUAN JOSE PIZARRO MARQUEZ

FREDYZ NIETO HUERTAS

Monografía de grado presentada como requisito para optar el título de Ingeniero de Sistemas

Director

MOISES QUINTANA

Ingeniero de Sistemas

UNIVERSIDAD TECNOLÓGICA DE BOLIVAR

FACULTAD DE INGENIERIA DE SISTEMAS

CARTAGENA DE INDIAS

2007

Nota de aceptación

_____

_____

_____

_____

Presidente del jurado

_____

Jurado

_____

Jurado

Cartagena de Indias, 17 agosto de 2007.

Señores:
COMITÉ DE EVALUACIÓN DE PROYECTOS
Universidad Tecnológica de Bolívar
E.N.S.M

Estimados Señores:

Comedidamente, nos permitimos solicitar a Ustedes, evaluar y aprobar el trabajo **DISEÑO DE UN PROTOTIPO DE SOFTWARE DE NOTAS PARA DOCENTES UNIVERSITARIOS**

Como requisito parcial para aprobar el Minor en Ingeniería de Software.

Atentamente,

JUAN JOSE PIZARRO MARQUEZ
C.C. 73.194.557 de Cartagena

FREDYS NIETO HUERTAS
9.153.265 de María la baja

Cartagena de Indias, 17 de agosto de 2007.

Señores:
COMITÉ DE EVALUACIÓN DE PROYECTOS
Universidad Tecnológica de Bolívar
LC.

Respetados Señores:

Tengo el agrado de presentar a su consideración, estudio y aprobación, la monografía titulada **DISEÑO DE UN PROTOTIPO DE SOFTWARE DE NOTAS PARA DOCENTES UNIVERSITARIOS** desarrollado por los estudiantes de Ingenieria de Sistemas **Juan José Pizarro Márquez y Fredys Nieto Huertas.**

Al respecto me permito comunicar que he dirigido el citado trabajo, el cual considero de gran importancia y utilidad.

Atentamente,

Ing. Moisés Quintana
Director de Proyectos

AUTORIZACIÓN

Cartagena de Indias, D.T. y C., agosto 17 de 2007

Nosotros Juan José Pizarro Márquez y Fredys Nieto Huertas, identificados con Cedula de ciudadanía números 73.194.557 de Cartagena (Bolívar) y 9.153.265 de Maria la baja, autorizamos a la Universidad Tecnológica de Bolívar para hacer uso de nuestro trabajo de grado y publicarlo en el catálogo online de la Biblioteca.

---------- ----------

JUAN JOSE PIZARRO MARQUEZ **FREDYS NIETO HUERTAS**

ARTICULO 105

La UNIVERSIDAD TECNOLOGICA DE BOLIVAR, se reserva el derecho de propiedad intelectual de todos los trabajos de grados aprobados, y no pueden ser explotados comercialmente sin autorización.

DEDICATORIA

Dedico este trabajo a mis padres Maribel Márquez Rivero y Juan José Pizarro Tovar, Por el gran apoyo que me han brindado, a mis familiares por ayudarme a superar los obstáculos presentados en el camino y tenderme la mano en el momento que yo lo necesité.

JUAN JOSÈ PIZARRO MARQUEZ

DEDICATORIA

A DIOS por darme fuerzas, fe y esperanzas; ser la luz que guia toda mi vida.

A mis padres por enseñarme los valores que hoy me hacen un hombre dec bien.

Y en especial a mis hijos (Carlos Daniel, Eliana Rebeca y Fredys Javier) por ser el motivo de todo lo que hago y sueño y y que llenan de felicidad mi alma,

FREDYS NIETO HUERTAS

AGRADECIMIENTOS

Agradecemos principalmente al cuerpo docente y a la institución por darnos las herramientas necesarias en nuestra instancia académica e investigativa.

Especialmente a nuestro Asesor Moisés Quintana, Por la orientación brindada durante la investigación.

TABLA DE CONTENIDO

PAG

INTRODUCCION

El siglo XXI se ha caracterizado por ser la era de la tecnología, suceso que ha con conllevado a las empresas a innovar en todos los procesos que le permitan desarrollar su objeto social, ser competitivas en su entorno, y mantenerse vigente en el mercado.

Las universidades no han sido agena en la aplicación de la tecnología, de hecho ellas son el eje fundamental en construir conocimiento a profesionales que día a día deben enfrentarse con esta realidad.

Sin embargo para que las universidades tengan excelentes resultados en e l logro de sus objetivos, entrega oportuna en el manejo de notas se presenta el siguiente **Diseño de un Prototipo de Software de Notas,** para docentes universitarios, que busca independizar de la coordinación académica los procesos de calificación y entrega de notas de las diversas áreas del saber; este es un método adecuado para lograr soportar datos que puedan consultar directamente tanto alumnos como docentes, sin depender del formato tradicional que congestiona y demora el manejo de la información que necesitan tanto estudiantes, docentes y la universidad.

Las bondades del diseño permiten el uso de la computadora, es de fácil manejo, puede ser de uso cotidiano en diferentes ambientes de estudio y trabajo, puede ser consultado por estudiantes de diferentes niveles de formación.

Su diseño se hizo utilizando herramientas importantes de la tecnología como: JAVA SERVER PAGE (J.S.P) Tecnología que permite mezclar HTML estático con HTML generando dinámicamente; MySQL gestor de base de datos, LENGUAJE DE MOLDEAMIENTO UNIFICADO (UML- unified modeling lenguaje) es un lenguaje gráfico para visualizar especificar y documentar cada una de las partes que comprende desarrollo de software.

1. MARCO TEORICO CONCEPTUAL

El Diseño del Prototipo de Software para Notas se desarrollo con base a herramientas de la tecnología como son:

1.1 LENGUAJE JAVA

Lenguaje desarrollado por Sun Microsystems para la elaboración de aplicaciones exportables a la red y capaces de operar sobre cualquier plataforma, normalmente, de visualizadores WWW. El programa Java se descarga desde el servidor Web y lo interpreta un programa que se ejecuta en el equipo que contiene el explorador Web; y sus ventajas son: uniformidad, comprensión, flexibilidad estabilidad, reusabilidad.

1.2 Tomcat (también llamado **Jakarta Tomcat** o **Apache Tomcat**) funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat implementa las especificaciones de los servlets y de JavaServer Pages (JSP) de Sun Microsystems. Se le considera un servidor de aplicaciones.

Tomcat lo desarrollan y lo mantienen miembros de la Apache Software Foundation y voluntarios independientes. Los usuarios disponen de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la *Apache Software Licence*. Las versiones más recientes son las 5.x, que implementan las especificaciones de Servlet 2.4 y de JSP 2.0.

La jerarquía de directorios de instalación de Tomcat incluye:

Bin - arranque, cierre, y otros scripts y ejecutables, common - clases comunes que pueden utilizar Catalina y las aplicaciones web, conf - ficheros XML y los orrespondientes DTDs para la configuración de Tomcat. logs - logs de Catalina y de las aplicaciones, server - clases utilizadas solamente por Catalina, shared - clases compartidas por todas las aplicaciones web, webapps - directorio que contiene las aplicaciones web, work - almacenamiento temporal de ficheros y directorios

1.3 JAVA SERVER PAGES (JSP)

Es una tecnología orientada a crear páginas web con programación en Java.

Con JSP podemos crear aplicaciones web que se ejecuten en diferentes servidores web, de múltiples plataformas, ya que Java es en esencia un lenguaje multiplataforma. Las páginas JSP están compuestas de código HTML/XML mezclado con etiquetas especiales

para programar scripts de servidor en lenguaje Java. Por tanto, las JSP podremos escribirlas con nuestro editor HTML/XML habitual.

En JSP creamos páginas de manera parecida a como se crean en ASP o PHP. tener archivos con extensión “jsp” que incluyen, dentro de la estructura de etiquetas HTML, las sentencias Java a ejecutar en el servidor.

Las ventajas que ofrece JSP es que se puede crear aplicaciones web que se ejecuten en varios servidores web, de múltiples plataformas, ya que Java es en esencia un lenguaje multiplataforma.

Las páginas JSP están compuestas de código HTML/XML mezclado con etiquetas especiales para programar scripts ejecutables en el servidor en sintaxis Java. Por lo tanto, las JSP podremos escribirlas con nuestro editor HTML/XMLhabitua

El uso de la tecnología JSP ofrece las ventajas como:

Separación entre contenidos dinámico y estático: El modelo JavaServer Pages le permite capturar la lógica de la aplicación en componentes JavaBeans estándar y reutilizables en los que puede definir la presentación utilizando etiquetas especiales JSP y pequeñas secciones de código Java conocido como scriptlets.

Soporte para programación dinámica (scripting): Las Java Server Pages permiten incluir líneas de programación dinámica ejecutadas en el servidor cuando la página es pedida.

Una vez escritas, funcionan en cualquier lugar: Las Javaserver Pages son una extensión de los Java servlets, un estándar 100% Java de JavaSoft. Las Java Server Pages heredan todas las ventajas de

la plataforma Java, Escríbalo una vez y hágalo funcionar en cualquier lugar o funcionalidad multi-plataforma.

Rendimiento avanzado y escalable: Las aplicaciones JSP disfrutan de la misma escalabilidad y rendimiento que los Java servlets ya que se trata de una extensión de la arquitectura Java servlet.

Gran calidad de soporte y documentación: La tecnología JSP es un estándar Java que permite a los usuarios y desarrolladores ordenar toda la documentación en este estándar.

1.4 SERVLET

Son programas en Java que se ejecutan en un servidor HTTP (servidor Web) actúan como capa intermedia entre; petición proveniente de un navegador Web u otro cliente HTTP, bases de datos o aplicaciones en el servidor HTTP

Las tareas encomendadas a un Servlet son:

Leer los datos enviados por un usuario usualmente de formularios en páginas Web, pueden venir de applets de Java o programas cliente http, buscar cualquier otra información sobre la petición que venga incluida en esta, detalles de las capacidades del navegador, cookies, nombre del host del cliente, generar los resultados, puede requerir consultas base de datos, invocar a otras aplicaciones, computar directamente la respuesta, dar formato a los resultados en un documento, Incluir la información en una página HTML, establecer los parámetros de la respuesta http, decirle al navegador el tipo de documento que se va a devolver, establecer las cookies, enviar el documento al cliente.

1.5 BASE DE DATOS

Una base de datos es un conjunto integrado de datos interrelacionados, junto con una serie de aplicaciones para su manejo, accesibles simultáneamente por diferentes usuarios y programas; presenta Las siguientes características: control centralizado de los datos, integridad de los datos, minimización de las repeticiones, independencia de los datos y las aplicaciones, acceso concurrente a los datos, costo mínimo de almacenamiento y mantenimiento, versatilidad para la representación de relaciones, establecimiento de medidas de seguridad, facilidad para el cambio, actualización y optimización.

La estructura y arquitectura de una base de datos

Almacena información de una serie de objetos, datos, entidades, relaciones o dependencias entre entidades, índices, consultas.

Los programas que permiten gestionar las bases de datos se denominan SGBD ó Sistemas de Gestión de Base de Datos. Una característica fundamental de un SGBD es que puede trabajar con diferentes bases de datos, por ejemplo: Almacén, Biblioteca, Agenda, etc.

Las ventajas que presenta una base de datos son:

Seguridad, rapidez y confiabilidad en:

Registros de todos los clientes, proveedores, productos, llistas de tarifas, descuentos, ofertas, impuestos, Listados de stock,

seguimiento de pedidos, albaranes, entregas. multitud de consultas de todos los registros de clientes, inventarios.

1.6 MySQL

Este gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración. Tiene características como:

Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo, soporta gran cantidad de tipos de datos para las columnas, dispone de API's en gran cantidad de lenguajes (C, C++, Java, PHP, gran portabilidad entre sistemas, soporta hasta 32 índices por tabla, gestión de usuarios y passwords, manteniendo un muy buen nivel de seguridad en los datos.

Presenta las siguientes ventajas

Mayor rendimiento, mejores utilidades de Admón. Integracón perfecta con PHP u JSP, sin límites en los tamaños de los registros, mejor control de acceso de usuarios.

Actualmente, se encuentra en fase de estandarización la versión 3, que será un lenguaje por sí mismo, y no necesitará de otros para actuar, nuevos tipos de datos complejos

1.7 MySQL ADMINISTRATOR

Un programa muy útil para administrar, visualmente y de manera sencilla, servidores de bases de datos MySQL.

MySQL Administrador es una herramienta que permite realizar tareas administrativas sobre servidores de MySQL incluyendo: La configuración de las opciones de inicio de los servidores, inicio y detención de servidores, monitorización de conexiones al servidor, administración de usuarios, monitorización del estado del servidor, incluyendo estadísticas de uso, visualización de los logs de servidor, gestión de copias de seguridad y recuperaciones, visualización de catálogos de datos.

1.8 DIAGRAMAS EN UML

Un diagrama es la representación gráfica de un conjunto de elementos o cosas. Esta representación se puede visualizar como un grafo conexo, en el que los vértices son las cosas y las aristas son las relaciones. Los diagramas se dibujan para mostrar una vista del sistema.

1.8.1 DIAGRAMA DE CASOS DE USO

Un diagrama de caso de usos muestra la relación entre un conjunto de casos de uso y actores. Los actores son un tipo especial de clases. Los diagramas de casos de usos abordan la vista estática de casos de usos de un sistema. Son importantes en organizar y modelar los comportamientos de un sistema.

1.8.2 DIAGRAMA DE CLASES

Un diagrama de clases muestra la relación existente entre un conjunto de clases, interfaces y colaboraciones.

Los diagramas de clases se emplean para visualizar los aspectos estáticos de los bloques básicos y sus relaciones y para especificar los detalles para construirlos. Los diagramas de clases contienen, casi siempre, los siguientes elementos, clases, interfaces, colaboraciones y relaciones de dependencia, generalización, asociación.

Usos Comunes

Se utilizan para modelar la vista de diseño estática de un sistema. Esta vista soporta principalmente los requerimientos funcionales de un sistema, los servicios que un sistema debe proporcionar a sus usuarios finales. Bajo este uso, los diagramas de clases son importantes para: Modelar el vocabulario de un sistema, en este caso el modelo solamente incluye, aquellas abstracciones claves del sistema junto con sus responsabilidades, modelar colaboraciones simples.

1.8.3 DIAGRAMA DE COLABORACION

Un diagrama de colaboración es una forma alternativa al diagrama de secuencia de mostrar un escenario. Este tipo de diagrama muestra las interacciones entre objetos organizadas entorno a los objetos y los enlaces entre ellos.

1.8.4 DIAGRAMA DE SECUENCIA

Un diagrama de secuencia muestra las interacciones entre objetos ordenados en secuencia temporal. En el escenario y la secuencia de mensajes intercambiados entre los objetos para llevar a cabo la funcionalidad descrita por el escenario.

Un diagrama de secuencia resulta importante para mantener los enlaces de los mensajes a los métodos apropiados del diagrama de clases.

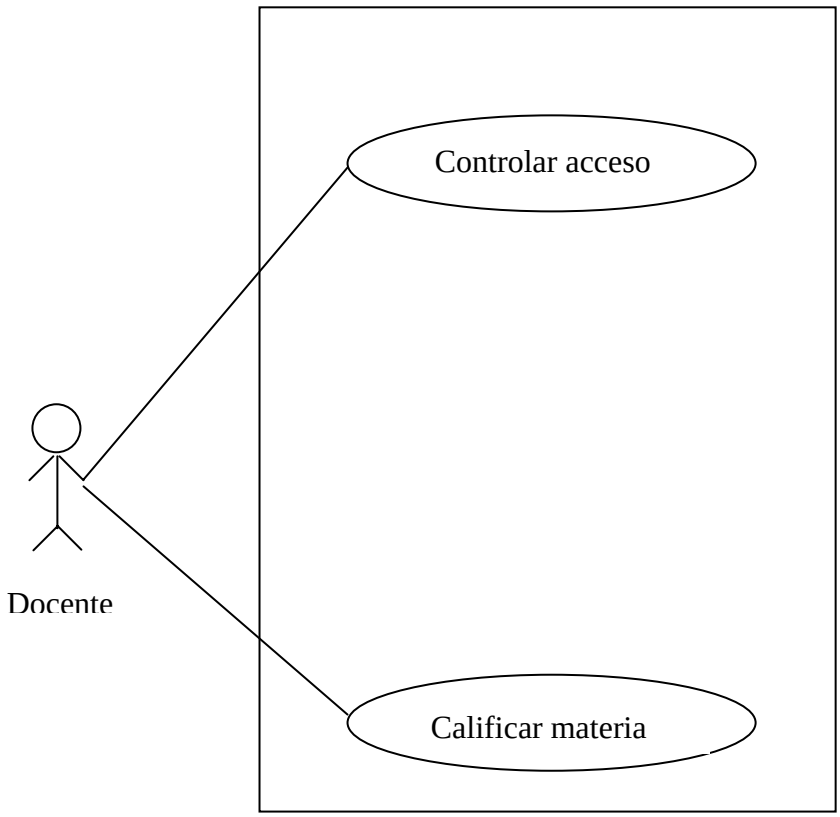
Un diagrama de secuencia un objeto se representa como una línea vertical punteada con un rectángulo de encabezado y con rectángulos a través de la línea principal que denotan la ejecución de métodos (activación). El rectángulo de encabezado contiene el nombre del objeto y el de su clase, en un formato *nombreObjeto: nombreClase*. Por ejemplo, el objeto *m*, instancia de la clase *MaquinaCafe* envía dos mensajes seguidos para dar respuesta a la operación *PedirProducto*: *Servir* al objeto *p* de la clase *Producto* y *DarVueltas* a sí mismo (*self-delegation*).

Activación: Muestra el período de tiempo en el cual el objeto se encuentra desarrollando alguna operación, bien sea por sí mismo o por medio de delegación a alguno de sus atributos. Se denota como un rectángulo delgado sobre la línea de vida del objeto. En el ejemplo anterior el objeto *ingredientes* se encuentra activo mientras ejecuta el método correspondiente al mensaje S*ervir*, el objeto *p* se encuentra activo mientras se ejecuta su método *Servir*, que ejecuta *_ingredientes.Servir*, y el objeto *m* se encuentra activo mientras se ejecuta *p.Servir* y *DarVueltas*.

Mensaje: El envío de mensajes entre objetos se denota mediante una línea sólida dirigida, desde el objeto que emite el mensaje hacia el objeto que lo ejecuta. En el ejemplo anterior el objeto *m* envía el mensaje *Servir* al objeto *p* y un poco más adelante en el tiempo el objeto *m* se envía a sí mismo el mensaje *DarVueltas*.

2. DISEÑO E IMPLEMENTACION DEL PROTOTIPO

2.1 DIAGRAMAS DE CASOS DE USOS



2.1.1 DESCRICION DE CASOS DE USOS

2.1.1.1 CONTROL DE ACCESO.

Nombre: Caso de uso 1.
Autor: Juan José Pizarro M.
Fecha: 23 junio de 2007.
Descripción: El sistema muestra la interfaz para el control de acceso, se debe escribir el código y la contraseña en la interfaz.
Actores: Docente
Precondiciones: El usuario debe haberse logeado en el sistema.
Flujo normal:
1. El sistema muestra 2 cajas de texto para ingresar el código y contraseña respectivamente.
2. El usuario puede hacer clic en los botones reset o aceptar.
3. El sistema comprueba la validez de los datos y los almacena en la base de datos.

Flujo alternativo: Si los datos ingresados no son correctos, se avisa al actor de ello permitiendo que los corrija.
Poscondiciones: El usuario ingresó al sistema.

Identificación de objetos participantes

Frontera: Interfaz index
Control: Botones reset y aceptar.
Entidad: Docente, Estudiante.

2.1.1.2 CALIFICAR MATERIA

Nombre: Caso de uso 2.

Autor: Juan José Pizarro M.

Fecha: 23 junio de 2007.

Descripción: El sistema muestra la interfaz de calificación de asignatura.

Actores: Docente.

Precondiciones: El usuario debe haberse logeado en el sistema.

Flujo normal:
1. El sistema busca las asignaturas del docente.
2. El sistema muestra las asignaturas.
3. El docente elije la asignatura a calificar.
4. El sistema busca los alumnos.
5. El sistema muestra los alumnos.
6. El docente elige el parcial a calificar
7. El docente escoge el alumno a calificar.
8. El docente ingresa la nota del parcial.
9. El docente hace clic en el botón guardar.
10. El sistema guarda la nota del parcial

Flujo alternativo: El sistema comprueba que la nota ingresada sea Numérica y se encuentre en el rango $0 \leq x \leq 5$, donde x es el valor de la nota.

Poscondiciones: El sistema guardo el valor de la nota en base de datos.

Identificación de objetos participantes

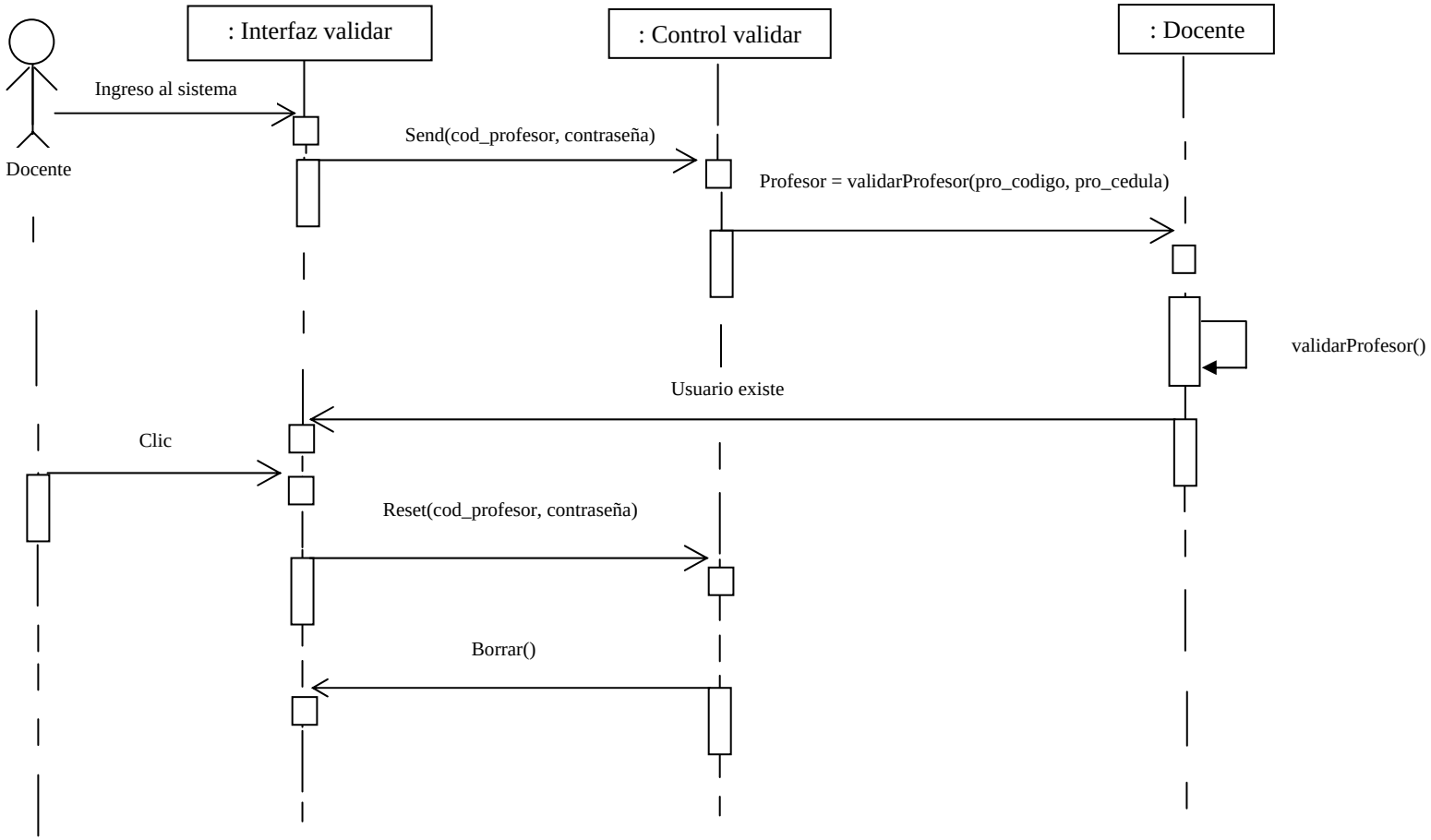
Frontera: Interfaz alumnos

Control: Etiqueta inicio, MyHorario, boton guardar.

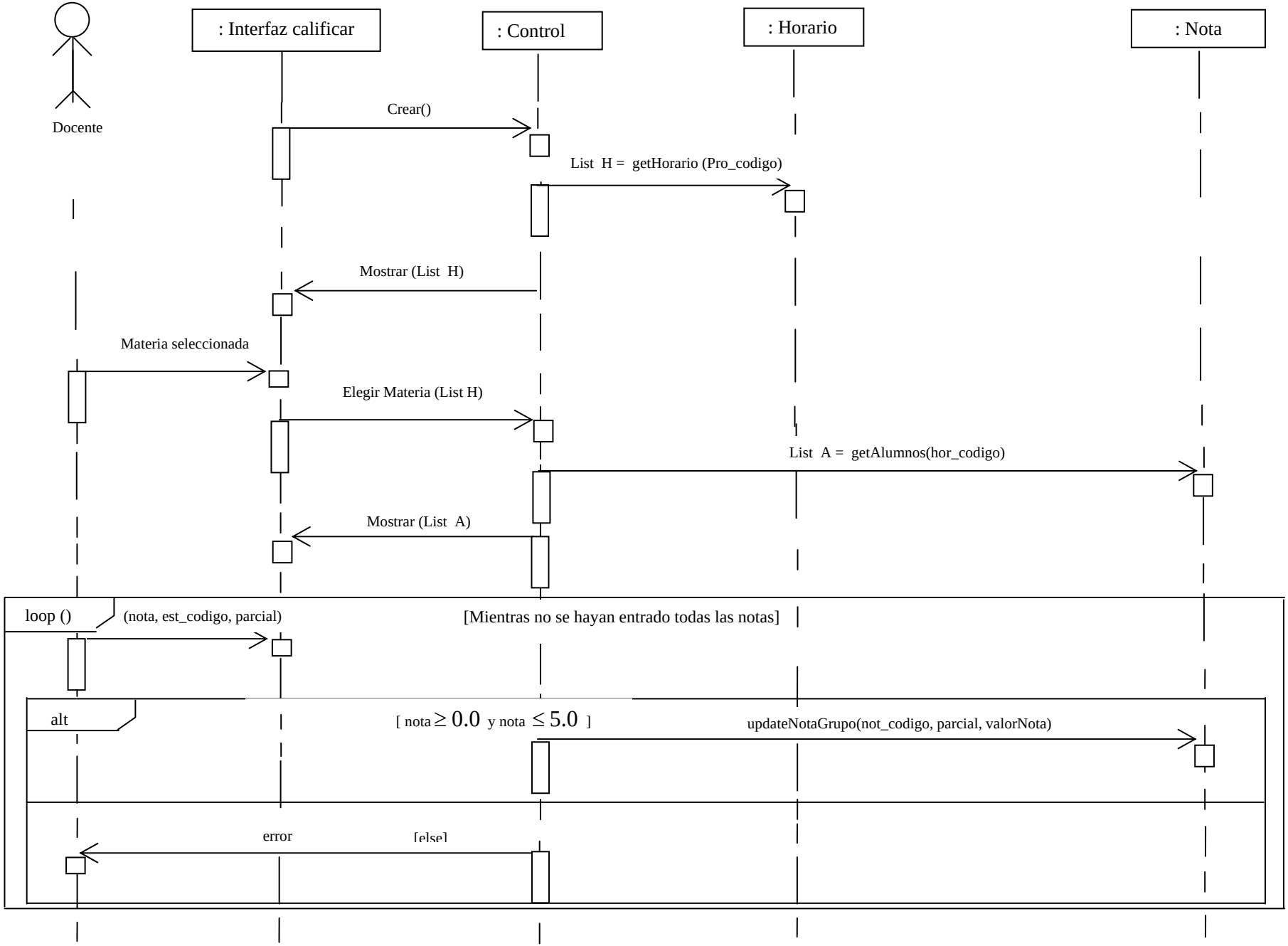
Entidad: Materia, Docente.

2.2 DIAGRAMA DE OBJETOS

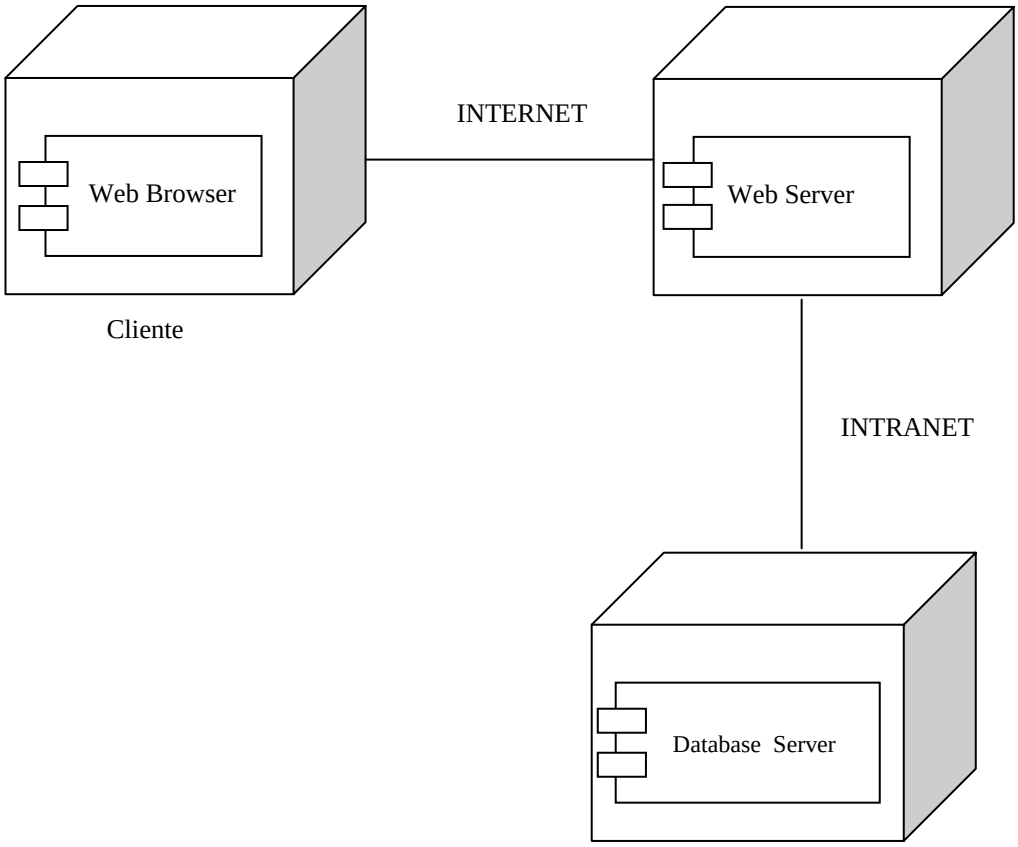
2.3 DIAGRAMA DE SECUENCIA CONTROL DE ACCESO



2.4 DIAGRAMA DE SECUENCIA CALIFICAR MATERIA



2.5 DIAGRAMA DE DESPLIEGUE



Un **servidor** es un tipo de software que realiza ciertas tareas en nombre de los usuarios. El término servidor ahora también se utiliza para referirse al ordenador físico en el cual funciona ese software, una máquina cuyo propósito es proveer datos de modo que otras máquinas puedan utilizar esos datos.

Un Database Server, se refiere a la maquina que aloja las bases de datos , nosotros elegimos el motor de base de datos MySQL, maneja las tablas de las base de datos a través de sentencias SQL como respuesta a las consultas hechas por los usuarios . **1.**

Un servidor web, se refiere a la máquina que almacena y maneja los sitios web, y en este sentido es utilizada por las compañías que ofrecen hosting o hospedaje. Alternativamente, el servidor web podría referirse al software, como el servidor de http de Apache, que funciona en la máquina y maneja la entrega de los componentes de los páginas web como respuesta a peticiones de los navegadores de los clientes. **2.**

Un Web Browser o también llamado cliente es una terminal de un usuario normal, que tiene como particularidad instalado un software de navegación web, dependiendo del sistema operativo usado, en Windows se usa Microsoft Internet Explorer **3.**

1. Database Server: MySQL Server

Username: root
Hostname: localhost
Port: 3306

Version: MySQL 4.0.16-nt via TCP/IP
IP: 127.0.0.1
Operating System: Windows XP
Hardware: Intel(R) Celeron(R) CPU 2.53 GHz, 504 RAM

2. Web Server: Apache Tomcat

Version : 4.1
Port: 8080
Network Name: Localhost
IP: 127.0.0.1
Protocolo: http
Operating System: Windows XP
Hardware: Intel(R) Celeron(R) CPU 2.53 GHz, 504 RAM

3. Cliente: Web Browser

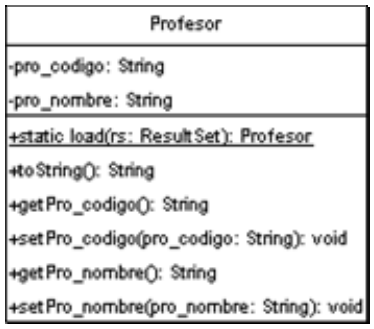
Microsoft Internet Explorer

Network Name: yeisa
Version: 6.0.2800.1106
IP: 164.164.164.1
Operating System: Windows Xp
Hardware: Intel(R) Celeron(R) CPU 3.4 GHz, 512 RAM

2.6 INTERFACES DE LA APLICACIÓN

El docente puede tener acceso al control de notas por medio de un código y una contraseña ya asignadas.

En la siguiente figura se muestra la clase profesor con los atributos y métodos utilizados para este prototipo.



A continuación vamos a ver la interacción que tiene el docente con el sistema:

2.6.1 Verificar docente en el sistema

El docente debe ingresar el código, contraseña y presionar el botón aceptar.

Inmediatamente se activa el método Verificar docente.

```
<% String pro_codigo_jsp;
  String pro_cedula_jsp;
   if((String)session.getAttribute("pro_codigo_jsp")!=null){
     pro_codigo_jsp=(String)session.getAttribute("pro_codigo_jsp");
          pro_cedula_jsp=(String)session.getAttribute("pro_cedula_jsp");
          }else{
     pro_codigo_jsp=request.getParameter("pro_codigo_jsp");
          pro_cedula_jsp=request.getParameter("pro_cedula_jsp");
          }
 %>
```

En el fragmento de código anterior el programa adquiere los datos ingresados: pro_codigo_jsp: Se refiere al código del profesor.

pro_cedula_jsp: Se refiere a la cedula del profesor.

Estos parámetros los envía a la siguiente función.

```
<% Profesor objProfesor=null;
  objProfesor=model.validarProfesor(pro_codigo_jsp,pro_cedula_jsp);
%>
```

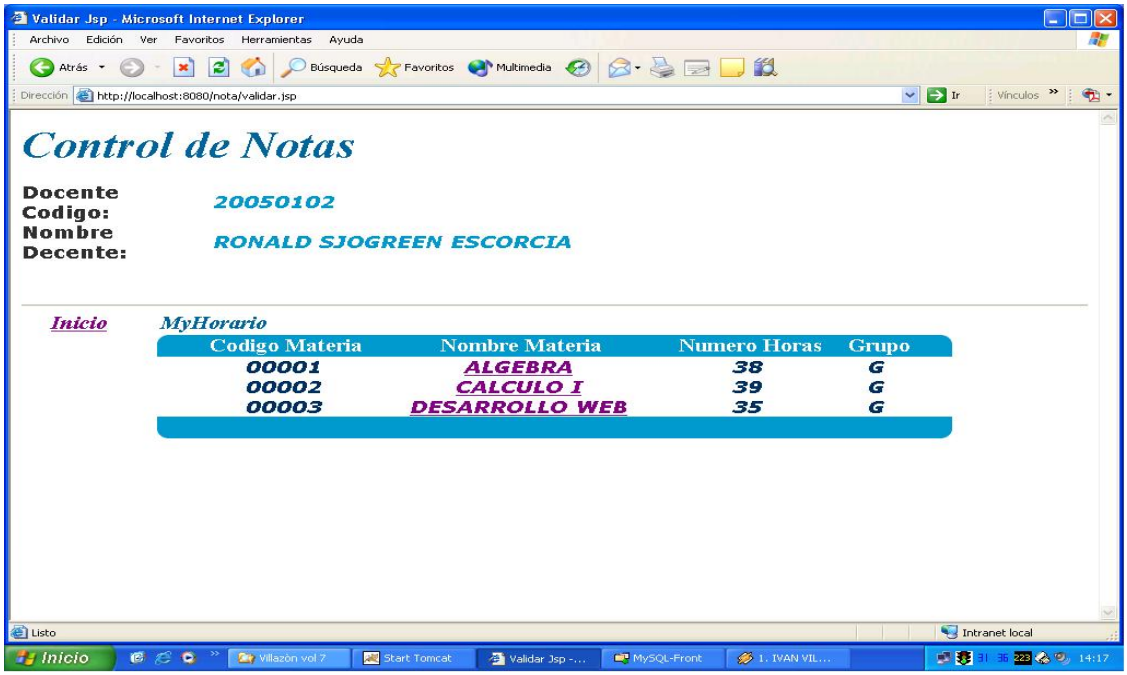
Esta se encarga de hacer la verificación de que realmente el docente exista en la base de datos.

La función validarProfesor(pro_codigo_jsp,pro_cedula_jsp);

Toma los parámetros pro_codigo_jsp,pro_cedula_jsp y devuelve un objeto de tipo profesor si el docente existe y sino devuelve un objeto nulo de tipo Profesor.

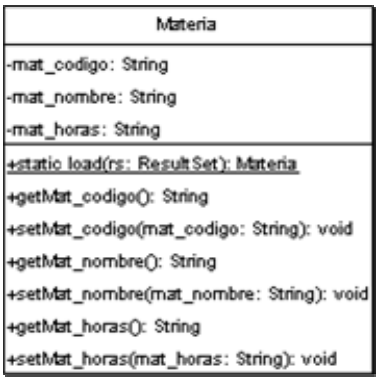
2.6.2 Selección de materias.

Después de que el sistema verifica que el docente exista automáticamente nos muestra esta interfaz.



Esta vista nos muestra que materia tiene asignada el docente que ha ingresado el sistema brindándole la posibilidad de seleccionar la materia.

En la siguiente figura se muestra la clase materia con los atributos y métodos utilizados para este prototipo



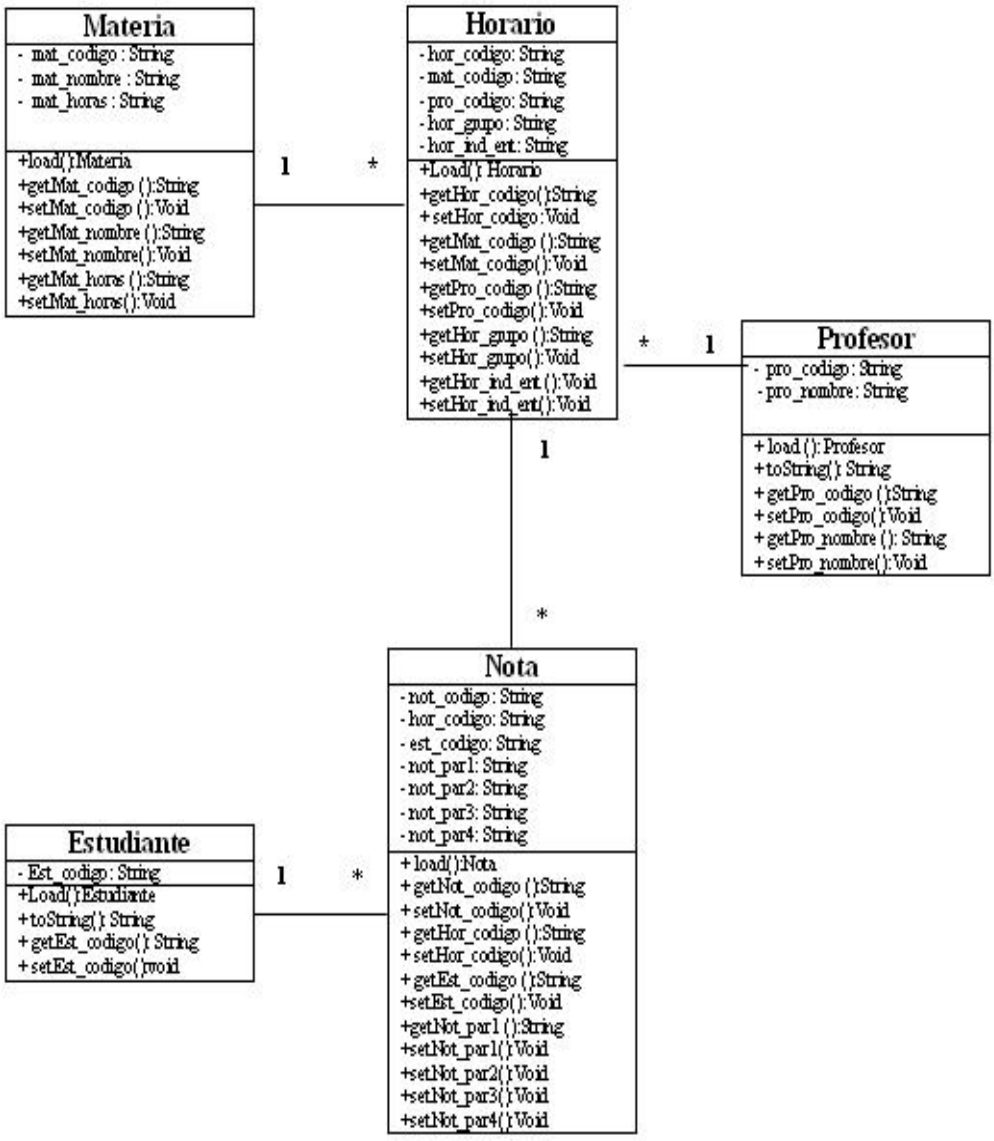
El docente puede elegir que asignatura desea calificar haciendo clic sobre el nombre de la materia.

Con el código del docente el sistema procede a listar las materias que el profesor tiene asignadas y además les agrega el hipervínculo para irse a listar los alumnos de la materia a la cual se le ha dado clic.

```
<%
  List listP=model.getHorario(pro_codigo_jsp);
  if (listP != null){
    Iterator itp= listP.iterator();
    while (itp.hasNext()){
      Horario horario=(Horario) itp.next();
      String hor_codigo=horario.getHor_codigo();// retorna el código del horario
              String mat_codigo=horario.getMat_codigo();//retorna el código de la
materia
              Materia objMateria=model.getMateria(mat_codigo);
              String mat_nombre=objMateria.getMat_nombre();//retorna el
nombre de la materia
              String mat_horas=objMateria.getMat_horas();//retorna el numero
de hrs de la materia
```

```
                String hor_grupo=horario.getHor_grupo(); // retorna el grupo del
horario.
        %>
       <tr bgcolor="#FFFFFF">
        <td> </td>
        <td><div align="center"><font color="#003366" size="2" face="Verdana,
Arial, Helvetica, sans-serif"><em><strong><%= hor_codigo
%></strong></em></font></div></td>
        <td><div align="center"><font color="#003366" size="2" face="Verdana,
Arial, Helvetica, sans-serif"><em><strong><a
href="Alumnos.jsp?hor_codigo=<%= hor_codigo %>&amp;mat_nombre=<%=
mat_nombre %>" ><%= mat_nombre
%></a></strong></em></font></div></td>
        <td><div align="center"><font color="#003366" size="2" face="Verdana,
Arial, Helvetica, sans-serif"><em><strong><%= mat_horas
%></strong></em></font></div></td>
        <td><div align="center"><font color="#003366" size="2" face="Verdana,
Arial, Helvetica, sans-serif"><em><strong><%= hor_grupo
%></strong></em></font></div></td>
        <td><font color="#003366"> </font></td>
       </tr>
       <% }} %>
```

En el fragmento de código anterior hacemos uso de la clase horario y materia. Ver diagrama de clases del sistema.



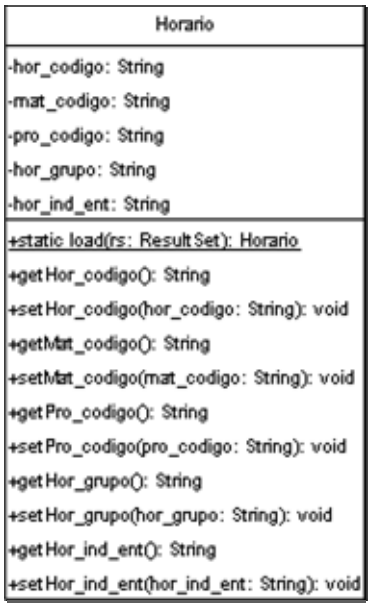
En el fragmento de código anterior se hace la invocación al método

```
model.getHorario(pro_codigo_jsp);
```

Este método utiliza como parámetro el código del profesor pro_codigo_jsp , chequea en la base de datos la tabla horario y lista las materias que el profesor tiene asignadas.

```
public List getHorario(String pro_codigo) throws SQLException {
        Horario objHorario=null;
        List listHorario=null;
        PreparedStatement pst = null;
        ResultSet rs = null;
        String sentencia = " Select * From horario Where pro_codigo = ? ";
        try {
                pst = con.prepareStatement(sentencia);
                pst.setString(1, pro_codigo);
                listHorario = new LinkedList();
                rs = pst.executeQuery();
                while (rs.next()) {
                        listHorario.add(Horario.load(rs));
                }
        } finally {
                if (pst != null)
                        pst.close();
        }
        return (listHorario);
}
```

A continuación se muestra la clase horario con los atributos y métodos correspondientes:



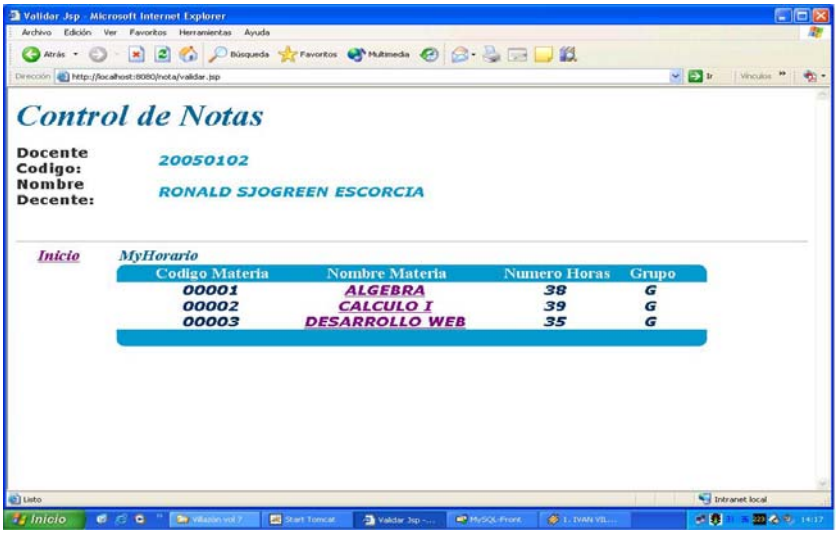
Para poder listar las materias asignadas del docente es necesario invocar al método model.getMateria(mat_codigo);

Con el fin de que nos devuelva un objeto de tipo materia y obtener el número de horas que exige esta. A continuación mostramos el código fuente.

```
public Materia getMateria(String mat_codigo) throws SQLException {
		Materia objMateria=null;
		PreparedStatement pst = null;
		ResultSet rs = null;
		String sentencia = " Select * From materia Where mat_codigo = ? ";
		try {
			pst = con.prepareStatement(sentencia);
			pst.setString(1, mat_codigo);
			rs = pst.executeQuery();
			if (rs.next()) {
				objMateria=Materia.load(rs);
			}
		} finally {
			if (pst != null)
				pst.close();
		}
		return (objMateria);
}
```

Este método utiliza como parámetro el código de la materia: mat_codigo, chequea en la base de datos la tabla materia y obtiene un objeto de tipo materia la cual concuerda con el código de la materia que previamente le hemos pasado como parámetro.

A medida que se va creando la página dinámicamente nos va mostrando los siguientes parámetros: nombre del docente, el código del docente. El código, nombre número de horas grupo de las materias que tiene asignadas el docente como lo vemos en la siguiente ilustración.



2.6.3. Ingresar notas

Para poder avanzar al siguiente nivel de la aplicación se le ha desarrollado un link, que accede a los estudiantes que cursan dicha materia. Este link es generado dinámicamente a través de la siguiente instrucción que se encuentra dentro de validar.jsp

```
<a href="Alumnos.jsp?hor_codigo=<%= hor_codigo %>&amp;mat_nombre=<%= mat_nombre %>" ><%= mat_nombre %></a>
```

Por medio de la anterior instrucción nos muestra la interfaz de alumnos.

Después de haber pulsado clic en la materia deseada por el docente la aplicación hace un llamado a la pagina Alumnos.jsp enviándole como parámetro el código del horario, nombre de la materia.

El siguiente código es el encargado de desarrollar la ventana anterior:

```
<% String hor_codigo=request.getParameter("hor_codigo");// de la pagina
anterior con request
   String mat_nombre=request.getParameter("mat_nombre");
   Horario horario=new Horario();
   int NumeroParcial=0;
    if(hor_codigo!=null){
          session.setAttribute("hor_codigo_jsp",hor_codigo);

horario=model.getHorario_ind((String)session.getAttribute("hor_codigo_jsp"));
          NumeroParcial=Integer.parseInt(horario.getHor_ind_ent())+1;
         }else
         {
         horario=model.getHorario_ind((String)session.getAttribute("hor_codigo_jsp"
));
         NumeroParcial=Integer.parseInt(horario.getHor_ind_ent())+1;
         }
   if(NumeroParcial>3){%>
    <jsp:forward page="NotasListas.jsp" />
 <%}%>
```

El fragmento de código anterior realiza las siguientes acciones:

Toma el código del horario, nombre de la materia y las asigna en dos variables, por medio de las siguientes instrucciones:

```
String hor_codigo=request.getParameter("hor_codigo");
horario=model.getHorario_ind((String)session.getAttribute("hor_codigo_jsp"));
```

Si existe un código de horario diferente de null, crea la variable de sesión: código del horario y guarda su valor en sesión para usarla en cualquier parte del programa, ya que si no la manejamos de esta forma pierde su valor en la interacción de paginas.

```
session.setAttribute("hor_codigo_jsp",hor_codigo);
```

A continuación se invoca al método obtener horario mediante el llamado:

```
horario=model.getHorario_ind((String)session.getAttribute("hor_codigo_jsp"));
```

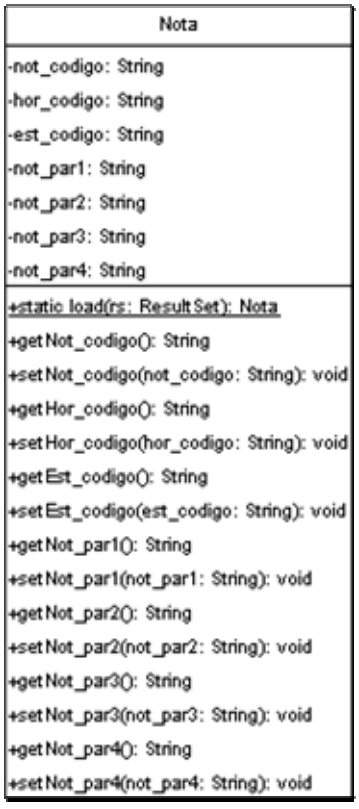
La función getHorario_Ind(String hor_codigo) devuelve un objeto de tipo horario que concuerde con el valor del parámetro que le hemos enviado consultando en la tabla horario de la base de datos.

```
public Horario getHorario_ind(String hor_codigo) throws SQLException {
                PreparedStatement pst = null;
                Horario horario=null;
                ResultSet rs = null;
                String sentencia = " Select * From horario Where hor_codigo = ? ";
                try {
                        pst = con.prepareStatement(sentencia);
                        pst.setString(1, hor_codigo);
                        rs = pst.executeQuery();
                        while (rs.next()) {
                                horario=Horario.load(rs);
                        }
                } finally {
                        if (pst != null)
                                pst.close();
```

```
            }
            return (horario);

}
```

Después de haber sido desplegada la interfaz de ingreso de notas de la materia seleccionada procedemos a introducirlas y presionar el botón guardar, para almacenar los datos en la tabla notas de la base de datos.

A continuación mostramos la clase nota, sus diferentes atributos y métodos.




```
<%
 List listP=model.getAlumnos((String)session.getAttribute("hor_codigo_jsp"));
 int i=0;
    if (listP != null){
    Iterator itp= listP.iterator();
    while (itp.hasNext()){
          i++;
          Nota nota=(Nota)itp.next();
          String not_codigo=nota.getNot_codigo();
```

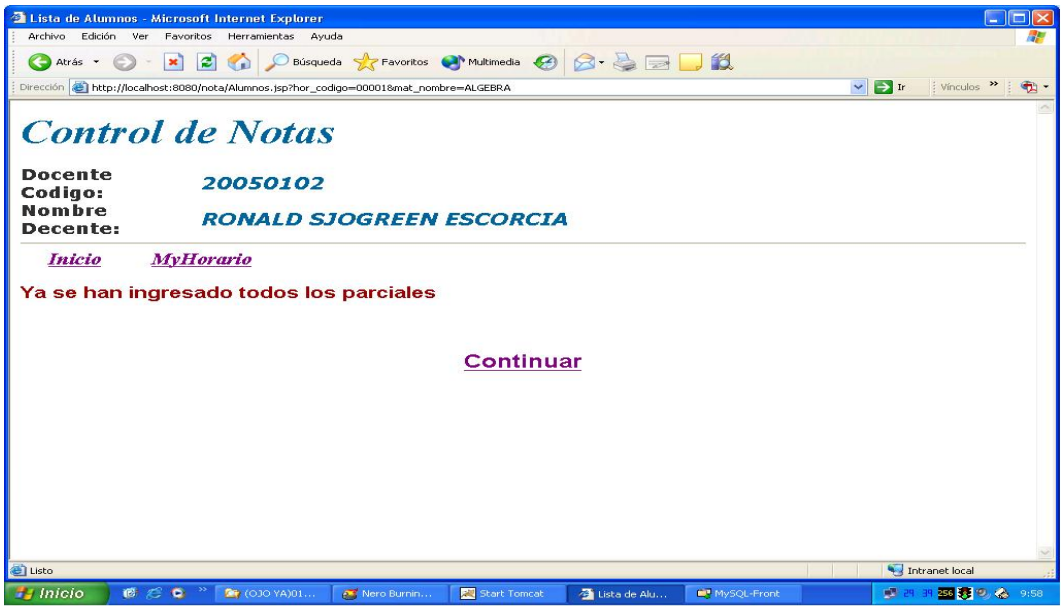
```
        // String hor_codigo=nota.geHor_codigo();
         String est_codigo=nota.getEst_codigo();
         String Nombre_estudiante=model.getNombreEstudiante(est_codigo);
         String not_par1=null;
         if(NumeroParcial==1)
         not_par1=nota.getNot_par1();
         if(NumeroParcial==2)
         not_par1=nota.getNot_par2();
         if(NumeroParcial==3)
         not_par1=nota.getNot_par3();

    %>
```

La siguiente función crea una lista de objetos de tipo notas, que son las tres notas materia seleccionada, teniendo como parámetro el código del horario.

```
public List getAlumnos(String hor_codigo1) throws SQLException {
                List listNota=null;
                PreparedStatement pst = null;
                ResultSet rs = null;
                String sentencia = " Select * From notas Where hor_codigo = ? ";
                try {
                        pst = con.prepareStatement(sentencia);
                        pst.setString(1, hor_codigo1);
                        listNota = new LinkedList();
                        rs = pst.executeQuery();
                        while (rs.next()) {
                                listNota.add(Nota.load(rs));
                        }
                } finally {
                        if (pst != null)
                                pst.close();
                }
                return (listNota);
}
```

```
 if(NumeroParcial>3){%>
   <jsp:forward page="NotasListas.jsp" />
 <%}%>
```

El anterior fragmento de código chequea que si las tres notas de la respectiva materia ya han sido introducidas se despliega la siguiente interfaz



Esta ventana muestra un mensaje anunciando que ya se han calificado todos los parciales, se debe presionar en Continuar para seguir utilizando la aplicación.

CONCLUSIONES

Después de muchas dificultades en el trabajo de investigación, diseño y elaboración del proyecto, donde por partes la mala documentación estaba muy extensa acerca de tecnologías de software, se logro resumir y fundirla en algo conciso y esencial para la comunidad de estudiantes de Ingeniería de Sistemas de la Universidad Tecnológica de Bolívar.

Gracias a la ciencia en la actualidad contamos con herramientas que nos brindan la facilidad a la hora de consultar alguna información de manera eficiente, confiable y segura. Con la ayuda del prototipo de software de notas construido, los docentes van a tener acceso a cada una de las materias que le han sido asignadas, existiendo gran confiabilidad de los datos de usuario y contraseña respectivo, lo cual se vera reflejado en economizar tiempo, disminución de costo, al momento de publicar las notas.

La llegada de la globalización nos hace ser más competitivos para acceder a la información en las diferentes entidades, ya sean estatales o privadas y con estos nuevos programas a menor costo posible, rápido acceso y fácil de utilizar siempre cuando mientras se haga la capacitación concerniente.

Con lo concerniente a la base de datos MySQL le facilita al programador una herramienta como MySQL Administrator. No cabe duda que se trata de un programa extremadamente útil e imprescindible para administrar visualmente servidores MySQL.

La utilización de bases de datos como plataforma para el desarrollo de sistemas de aplicación en las organizaciones se ha incrementado notablemente en los últimos años, se debe a las ventajas que ofrece su utilización, entre estas encontramos: Globalización de la información, Eliminación de información inconsistente, Integridad en la información, Independencia de datos, todo esto ayuda a la rápida proliferación del desarrollo de los sistemas de bases de datos.

RECOMENDACIONES

El objetivo primordial de este proyecto es Diseñar y hacer un prototipo de una herramienta que permita grabar y modificar las notas de los estudiantes por parte de los profesores de forma segura para evitar la doble trascripción y hacer más confiable y ágil este proceso de publicación y corrección de notas. Este estudio hizo énfasis en los detalles técnicos, para la eficiente utilización de la herramientas de Tecnologías de software como son: JAVA, MySQL ADMINISTRATOR, Apache Tomcat, Java Server Pages, Bases de Datos, Diagramas UML, creando una herramienta base, que sirve como apoyo al personal de desarrollo de aplicaciones de la universidad Tecnológica de Bolívar, ahorrando tiempo de investigación y reducir costos en la elaboración de un proyecto de mayor envergadura.

Todos los servicios que presta la aplicación requieren un óptimo desempeño, es por esto que se requiere un estudio del hardware usado en el desarrollo del proyecto, ya que este proyecto fue probado en maquinas con:

En la parte del servidor

Operating System: Windows Xp
Hardware: Intel(R) Celeron(R) CPU 2.5 GHz, 512 RAM

En la parte del cliente

Operating System: Windows Xp
Hardware: Intel(R) Celeron(R) CPU 3.4 GHz, 512 RAM

COMENTARIOS FINALES

La red utilizada para la prueba de este proyecto será de tipo LAN, porque constara de varias terminales que están interconectados en la red, para el acceso remoto y seguro.

En este caso hay que tener control sobre los parámetros de la red, como son : IP, hóstiame, puerto, versiones de programas, protocolo. Para que la comunicación sea exitosa y el envío de datos sea oportuno.

BIBLIOGRAFIA

Korth et al., Korth, H.F. y Silberschatz, A. – “Fundamentos de bases de datos. 2ª edición. Editorial: McGraw-Hill, 1993

J.M. Framiñan Torres: *Java*. Anaya Multimedia (col. Al Día en una Hora), Madrid, 1997. 128p.

DEITEL Y DEITEL, Como Programar en Java. Primera edicion. Editorial: Prentice Hall, 1998.1056p.

Apache Tomcat
Autor: Alejandro Pérez García
http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=tomcatIIS
Fecha de consulta : 21-06-2006

UML
Diagrama de secuencia; http://www-gris.det.uvigo.es/~avilas/UML/
Que es UML; http://www.creangel.com/uml/
Fecha de consulta: 23-06-2006

Modelamiento Visual y UML
Autores: Profesores de Análisis y diseño de sistemas I de CIBERTEC,
Fecha de consulta: 29 -07-2007

MySQL Administrador
Autor : Desarrolloweb.com
http://www.desarrolloweb.com/articulos/1798.php
Fecha de consulta: 23-05-2006

Java

Autor: Programación con JAVA

La tecnología Java http://www.fi-b.unam.mx/pp/profesores/carlos/java/

Fecha de consulta: 30-05-2006

Lenguaje Java

Autor: M.C. Bernabé Ortiz y Herbert

http://www.itlp.edu.mx/posgrado/lengprog/java.html

Fecha de consulta: 02-05-2006

Java Server Pages

Autor: terra.com

http://www.terra.es/tecnologia/glosario/ficha.cfm?id_termino=145

Fecha de consulta: 04-05-2006

Que es JSP

Autor: DesarrolloWeb.com.

http://www.desarrolloweb.com/articulos/831.php

Fecha de consulta: 04-05-2006

ANEXO 1

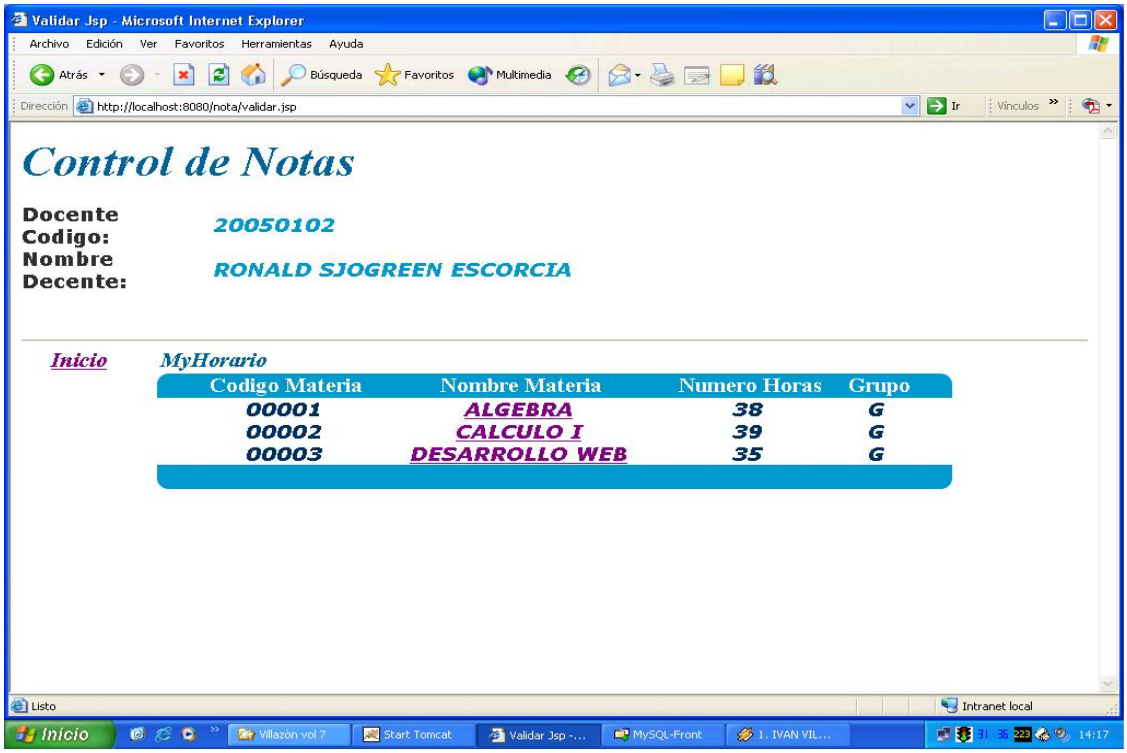
Manual de Usuario

En esta página nos muestra que ingresemos el código del docente y la contraseña del mismo, después de haber ingresado correctamente el código y la contraseña presionamos el botón aceptar (Ver figura 1).

Si por alguna razón el código o contraseña, código y contraseña están escritos incorrectamente puedes presionar el botón reset para borrar ambas cajas de texto (Ver figura 2).

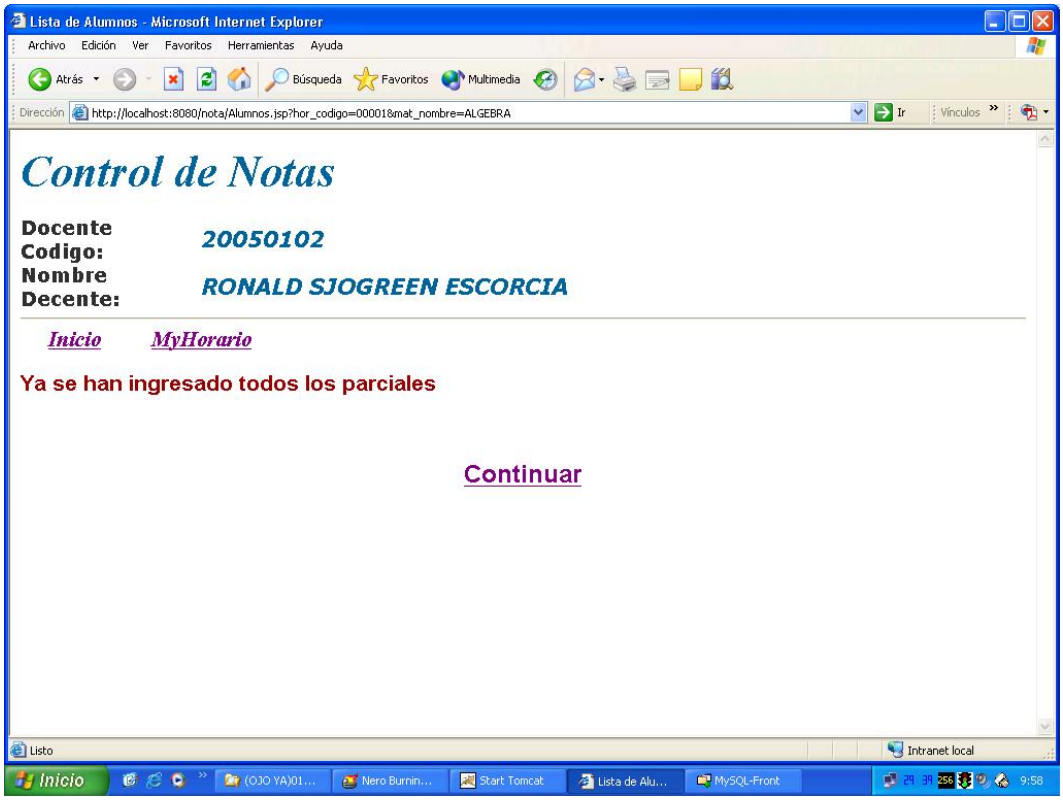
Esta vista nos muestra el Código, Nombre y el parcial a calificar de cada estudiante, en una determinada materia. En el campo Parcial x: donde x =1, x =2, x =3, que representa que parcial va calificar el docente.

Después de haber ingresado la nota del parcial de cada uno de los estudiantes, se presiona el botón guardar para añadir la calificación a la Base de datos.



Esta vista nos muestra que materia tiene asignada cada docente, donde cada asignatura está compuesta del código de materia, nombre de materia, numero de horas de la materia y grupo de la materia.

El docente puede elegir que asignatura va a calificar presionando sobre el nombre de la materia.



Esta venta muestra un mensaje anunciando que ya se han calificado todos los parciales, se debe presionar en Continuar para seguir utilizando la aplicación.

ANEXO 2

CODIGO FUENTE DE CLASES

```
package mvc.model;

import java.io.*;
import java.util.*;
import java.sql.ResultSet;
import java.sql.SQLException;

public class Estudiante implements Serializable{
	private String Est_codigo;

	public static Estudiante load (ResultSet rs) throws
SQLException{
		Estudiante estudiante = new Estudiante ();
		estudiante.setEst_codigo(rs.getString(1)+"");

		return estudiante;
		}

	public String toString (){
		StringBuffer sb = new StringBuffer();
		sb.append(getEst_codigo());

		return sb.toString ();
		}

	public String getEst_codigo (){
		return Est_codigo;
		}
```

```
        public void setEst_codigo (String Est_codigo){
                this.Est_codigo=Est_codigo;
                }


        }
package mvc.model;
import java.io.*;
import java.util.*;
import java.sql.ResultSet;
import java.sql.SQLException;




public class Horario implements Serializable{
        private String hor_codigo;
        private String mat_codigo;
        private String pro_codigo;
        private String hor_grupo;
        private String hor_ind_ent;

        public static Horario load (ResultSet rs) throws SQLException{
                Horario horario = new Horario ();
                horario.setHor_codigo(rs.getString(1)+"");
                horario.setMat_codigo(rs.getString(2)+"");
                horario.setPro_codigo(rs.getString(3)+"");
                horario.setHor_grupo(rs.getString(4)+"");
                horario.setHor_ind_ent(rs.getString(16)+"");
                return horario;
```

```
        }


public String getHor_codigo (){
        return hor_codigo;
        }
public void setHor_codigo (String hor_codigo){
        this.hor_codigo=hor_codigo;
        }

public String getMat_codigo (){
        return mat_codigo;
        }
public void setMat_codigo (String mat_codigo){
        this.mat_codigo=mat_codigo;
        }
public String getPro_codigo (){
        return pro_codigo;
        }
public void setPro_codigo (String pro_codigo){
        this.pro_codigo=pro_codigo;
        }

public String getHor_grupo (){
        return hor_grupo;
        }
public void setHor_grupo (String hor_grupo){
        this.hor_grupo=hor_grupo;
        }
public String getHor_ind_ent (){
```

```
            return hor_ind_ent;
            }
        public void setHor_ind_ent (String hor_ind_ent){
            this.hor_ind_ent=hor_ind_ent;
            }

        }

package mvc.model;

import java.io.*;
import java.util.*;
import java.sql.ResultSet;
import java.sql.SQLException;

public class Materia implements Serializable{
        private String mat_codigo;
        private String mat_nombre;
        private String mat_horas;

        public static Materia load (ResultSet rs) throws SQLException{
            Materia materia = new Materia ();
            materia.setMat_codigo(rs.getString(1)+"");
            materia.setMat_nombre(rs.getString(2)+"");
            materia.setMat_horas(rs.getString(3)+"");
            return materia;
            }

        public String getMat_codigo (){
            return mat_codigo;
```

```
        }
    public void setMat_codigo (String mat_codigo){
        this.mat_codigo=mat_codigo;
        }

    public String getMat_nombre (){
        return mat_nombre;
        }
    public void setMat_nombre (String mat_nombre){
        this.mat_nombre=mat_nombre;
        }
    public String getMat_horas (){
        return mat_horas;
        }
    public void setMat_horas (String mat_horas){
        this.mat_horas=mat_horas;
        }



    }

package mvc.model;

import java.io.*;
import java.util.*;
import java.sql.ResultSet;
import java.sql.SQLException;

public class Nota implements Serializable{
    private String not_codigo;
```

```
private String hor_codigo;
private String est_codigo;
private String not_par1;
private String not_par2;
private String not_par3;
private String not_par4;

public static Nota load (ResultSet rs) throws SQLException{
	Nota nota = new Nota ();
	nota.setNot_codigo(rs.getString(1)+"");
	nota.setHor_codigo(rs.getString(2)+"");
	nota.setEst_codigo(rs.getString(3)+"");
	nota.setNot_par1(rs.getString(9)+"");
	nota.setNot_par2(rs.getString(10)+"");
	nota.setNot_par3(rs.getString(11)+"");
	nota.setNot_par4(rs.getString(12)+"");
	return nota;
	}


public String getNot_codigo (){
	return not_codigo;
	}
public void setNot_codigo (String not_codigo){
	this.not_codigo=not_codigo;
	}

public String getHor_codigo (){
	return hor_codigo;
	}
```

```
public void setHor_codigo (String hor_codigo){
        this.hor_codigo=hor_codigo;
        }
public String getEst_codigo (){
        return est_codigo;
        }
public void setEst_codigo (String est_codigo){
        this.est_codigo=est_codigo;
        }

public String getNot_par1 (){
        return not_par1;
        }
public void setNot_par1 (String not_par1){
        this.not_par1=not_par1;
        }
        public String getNot_par2 (){
        return not_par2;
        }
public void setNot_par2 (String not_par2){
        this.not_par2=not_par2;
        }
        public String getNot_par3 (){
        return not_par3;
        }
public void setNot_par3 (String not_par3){
        this.not_par3=not_par3;
        }
        public String getNot_par4 (){
        return not_par4;
```

```
        }
    public void setNot_par4 (String not_par4){
        this.not_par4=not_par4;
        }
    }

package mvc.model;

import java.io.*;
import java.util.*;
import java.sql.ResultSet;
import java.sql.SQLException;

public class Profesor implements Serializable{
    private String pro_codigo;
    private String pro_nombre;

    public static Profesor load (ResultSet rs) throws SQLException{
        Profesor profesor = new Profesor ();
        profesor.setPro_codigo(rs.getString(1)+"");
        profesor.setPro_nombre(rs.getString(2)+"");
        return profesor;
        }

    public String toString (){
        StringBuffer sb = new StringBuffer();
        sb.append(getPro_codigo());
        sb.append(getPro_nombre());

        return sb.toString ();
```

```
        }

    public String getPro_codigo (){
        return pro_codigo;
        }
    public void setPro_codigo (String pro_codigo){
        this.pro_codigo=pro_codigo;
        }

    public String getPro_nombre (){
        return pro_nombre;
        }
    public void setPro_nombre (String pro_nombre){
        this.pro_nombre=pro_nombre;
        }

    }

package mvc.model;

import java.io.*;
import java.sql.*;
import java.util.*;

public class Model implements Serializable{

//Campos de Configuracion

    private transient Connection con;
    private String jdbcDriver;
```

```
        private String databaseURL;
        private List listaEstudiantes;
        //Campos de Paciente

//Metodos de Configuracion y de bases de datos

        public void connect() throws SQLException{
         if(this.isConnect())
                throw new SQLException("Ya esta conectado");

//Verificar que el controlador(driver) y la URL hayan sido especificado

        if (jdbcDriver==null)
                throw new SQLException("No hay jdbc Driver cargado");

    if (databaseURL==null)
                throw new SQLException("No hay URL cargado en base de
datos");

//Cargar Controlador

        try{
           Class.forName(jdbcDriver).newInstance();
         }catch(Exception e1){
                  throw new SQLException("La clase"+ jdbcDriver+" No se
pudo cargar");
     }

//Abrir la conexion
        con=DriverManager.getConnection(this.databaseURL);
```

```
        }

//Cierra la conexio en curso
        public void disconnect(){
        //Cerrar conexion
    if(con!=null){
        try{
        con.close();
        }catch(SQLException ignore){}
        finally{
                con=null;
                }
        }
   }

//Devuelve verdadero si hay una conexion Activa

        public boolean isConnect(){
                return (con!=null);
        }
        public Connection getConexion(){
                return con;
        }
        public String getJdbcDriver(){
                return jdbcDriver;
        }
    public void setJdbcDriver(String jdbcDriver){
                this.jdbcDriver=jdbcDriver;
        }
    public String getDatabaseURL(){
```

```
            return databaseURL;
      }
      public void setDatabaseURL(String databaseURL){
            this.databaseURL= databaseURL;
      }
//*****************************************************
******
//metodo para traer los nombres de los campos
      public List getListaEstudiantes(){
            return listaEstudiantes;
      }
      public void listEstudiantes() throws SQLException{
            if (!isConnect())
            throw new SQLException("Sin Conexion");
            PreparedStatement pstmt=null;
            ResultSet rs=null;
            try{
                  pstmt=con.prepareStatement("Select * From
Estudiante");
                  rs=pstmt.executeQuery();
                  listaEstudiantes=new LinkedList();
                  while(rs.next())
                  listaEstudiantes.add(Estudiante.load(rs));
            } finally{
                  if(rs!=null)
                  rs.close();
                  if(pstmt !=null)
                  pstmt.close();
            }
      }
```

```
      //ResultSetMetaData
//**********************************************************
******
  public Profesor validarProfesor(String pro_codigo,String
pro_cedula) throws SQLException {
            Profesor objProfesor=null;
            PreparedStatement pst = null;
            ResultSet rs = null;
            String sentencia = " Select * From profesor Where
pro_codigo = ? and cedula = ? ";
            try {
                  pst = con.prepareStatement(sentencia);
                  pst.setString(1, pro_codigo);
                  pst.setString(2, pro_cedula);
                  rs = pst.executeQuery();
                  if (rs.next()) {
                     objProfesor=Profesor.load(rs);

                  }
            } finally {
                  if (pst != null)
                        pst.close();
            }
            return (objProfesor);
}
  public List getHorario(String pro_codigo) throws SQLException {
            Horario objHorario=null;
            List listHorario=null;
            PreparedStatement pst = null;
            ResultSet rs = null;
```

```
            String sentencia = " Select * From horario Where
pro_codigo = ? ";
            try {
                  pst = con.prepareStatement(sentencia);
                  pst.setString(1, pro_codigo);
                  listHorario = new LinkedList();
                  rs = pst.executeQuery();
                  while (rs.next()) {
                        listHorario.add(Horario.load(rs));
                  }
            } finally {
                  if (pst != null)
                        pst.close();
            }
            return (listHorario);
}
  public Materia getMateria(String mat_codigo) throws SQLException
{
            Materia objMateria=null;
            PreparedStatement pst = null;
            ResultSet rs = null;
            String sentencia = " Select * From materia Where
mat_codigo = ? ";
            try {
                  pst = con.prepareStatement(sentencia);
                  pst.setString(1, mat_codigo);
                  rs = pst.executeQuery();
                  if (rs.next()) {
                        objMateria=Materia.load(rs);
                  }
```

```
            } finally {
                  if (pst != null)
                        pst.close();
            }
            return (objMateria);
}
  public String getNombreEstudiante(String est_codigo) throws
SQLException {
            PreparedStatement pst = null;
            String cadenaResul=null;
            ResultSet rs = null;
            String sentencia = " Select * From estudiante Where
est_codigo = ? ";
            try {
                  pst = con.prepareStatement(sentencia);
                  pst.setString(1, est_codigo);
                  rs = pst.executeQuery();
                  if (rs.next()) {
                        cadenaResul=rs.getString(23)+"
"+rs.getString(21)+" "+rs.getString(22);
                  }
            } finally {
                  if (pst != null)
                        pst.close();
            }
            return (cadenaResul);
}

 public List getAlumnos(String hor_codigo1) throws SQLException {
            List listNota=null;
```

```
            PreparedStatement pst = null;
            ResultSet rs = null;
            String sentencia = " Select * From notas Where
hor_codigo = ? ";
            try {
                  pst = con.prepareStatement(sentencia);
                  pst.setString(1, hor_codigo1);
                  listNota = new LinkedList();
                  rs = pst.executeQuery();
                  while (rs.next()) {
                        listNota.add(Nota.load(rs));
                  }
            } finally {
                  if (pst != null)
                        pst.close();
            }
            return (listNota);
}

public void updateNota(Nota nota)throws SQLException{
            PreparedStatement pstmt=null;
            try{
                  pstmt=con.prepareStatement("Update Notas Set "+
                  "  par1=? "+
                  " ,par2=? "+
                  " ,par3=? "+
                  " ,def=? "+
                  " Where not_codigo = ? "
            );
                  pstmt.setString(1,nota.getNot_par1());
```

```
                pstmt.setString(2,nota.getNot_par2());
                pstmt.setString(3,nota.getNot_par3());
                pstmt.setString(4,nota.getNot_par4());
                pstmt.setString(5,nota.getNot_codigo());
                pstmt.executeUpdate();
        }
        finally{
                if (pstmt!=null)
                        pstmt.close();
        }
    }

public Horario getHorario_ind(String hor_codigo) throws
SQLException {
        PreparedStatement pst = null;
        Horario horario=null;
        ResultSet rs = null;
        String sentencia = " Select * From horario Where
hor_codigo = ? ";
        try {
                pst = con.prepareStatement(sentencia);
                pst.setString(1, hor_codigo);
                rs = pst.executeQuery();
                while (rs.next()) {
                        horario=Horario.load(rs);
                }
        } finally {
                if (pst != null)
                        pst.close();
        }
```

```
            return (horario);

}

public void updateNotaGrupo(String not_codigo,int parcial,String
valorNota)throws SQLException{
            PreparedStatement pstmt=null;
            String cadena=null;
            if(parcial==1)
                  cadena="Update Notas Set par1=? Where
not_codigo = ? ";
            if(parcial==2)
                  cadena="Update Notas Set par2=? Where
not_codigo = ? ";
            if(parcial==3)
                  cadena="Update Notas Set par3=? Where
not_codigo = ? ";

            try{
                  pstmt=con.prepareStatement(cadena);
                  pstmt.setString(1,valorNota);
                  pstmt.setString(2,not_codigo);
                  pstmt.executeUpdate();
            }
            finally{
                  if (pstmt!=null)
                        pstmt.close();
            }
      }
```

```
public void updateNotasGeneral(String hor_codigo,Vector vector,int
parcial) throws SQLException {
		List listNota=null;
		PreparedStatement pst = null;
		ResultSet rs = null;
		int i=-1;
		String sentencia = " Select * From notas Where
hor_codigo = ? ";
		try {
			pst = con.prepareStatement(sentencia);
			pst.setString(1, hor_codigo);
			rs = pst.executeQuery();
			while (rs.next()) {
				i++;

updateNotaGrupo(rs.getString(1),parcial,(String)vector.elementAt(i))
;
			}
		} finally {
			if (pst != null)
				pst.close();
		}

}

public void updateInd_ent(int valor,String hor_codigo)throws
SQLException{
		PreparedStatement pstmt=null;
		String cadena=null;
```

```
            cadena="Update horario Set ind_ent=? Where hor_codigo
= ? ";
            try{
                  pstmt=con.prepareStatement(cadena);
                  pstmt.setString(1,String.valueOf(valor));
                  pstmt.setString(2,hor_codigo);
                  pstmt.executeUpdate();
            }
            finally{
                  if (pstmt!=null)
                        pstmt.close();
            }
      }


}
package mvc.model;

import java.io.*;
import java.sql.*;
import java.util.*;

public class Model implements Serializable{

//Campos de Configuracion

      private transient Connection con;
      private String jdbcDriver;
      private String databaseURL;
      private List listaEstudiantes;
```

```
        //Campos de Paciente

//Metodos de Configuracion y de bases de datos

        public void connect() throws SQLException{
         if(this.isConnect())
                throw new SQLException("Ya esta conectado");

//Verificar que el controlador(driver) y la URL hayan sido especificado

        if (jdbcDriver==null)
                throw new SQLException("No hay jdbc Driver cargado");

    if (databaseURL==null)
                throw new SQLException("No hay URL cargado en base de
datos");

//Cargar Controlador

        try{
          Class.forName(jdbcDriver).newInstance();
         }catch(Exception e1){
                  throw new SQLException("La clase"+ jdbcDriver+" No se
pudo cargar");
     }

//Abrir la conexion
        con=DriverManager.getConnection(this.databaseURL);
        }
```

```
//Cierra la conexio en curso
        public void disconnect(){
        //Cerrar conexion
    if(con!=null){
        try{
        con.close();
        }catch(SQLException ignore){}
        finally{
                con=null;
                }
        }
   }

//Devuelve verdadero si hay una conexion Activa

        public boolean isConnect(){
                return (con!=null);
        }
        public Connection getConexion(){
                return con;
        }
        public String getJdbcDriver(){
                return jdbcDriver;
        }
    public void setJdbcDriver(String jdbcDriver){
                this.jdbcDriver=jdbcDriver;
        }
    public String getDatabaseURL(){
                return databaseURL;
        }
```

```
	public void setDatabaseURL(String databaseURL){
		this.databaseURL= databaseURL;
	}
//*********************************************************
******
//metodo para traer los nombres de los campos
	public List getListaEstudiantes(){
		return listaEstudiantes;
	}
	public void listEstudiantes() throws SQLException{
		if (!isConnect())
		throw new SQLException("Sin Conexion");
		PreparedStatement pstmt=null;
		ResultSet rs=null;
		try{
			pstmt=con.prepareStatement("Select * From
Estudiante");
			rs=pstmt.executeQuery();
			listaEstudiantes=new LinkedList();
			while(rs.next())
			listaEstudiantes.add(Estudiante.load(rs));
		} finally{
			if(rs!=null)
			rs.close();
			if(pstmt !=null)
			pstmt.close();
		}
	}
	//ResultSetMetaData
```

```
//*****************************************************
******
  public Profesor validarProfesor(String pro_codigo,String
pro_cedula) throws SQLException {
            Profesor objProfesor=null;
            PreparedStatement pst = null;
            ResultSet rs = null;
            String sentencia = " Select * From profesor Where
pro_codigo = ? and cedula = ? ";
            try {
                  pst = con.prepareStatement(sentencia);
                  pst.setString(1, pro_codigo);
                  pst.setString(2, pro_cedula);
                  rs = pst.executeQuery();
                  if (rs.next()) {
                     objProfesor=Profesor.load(rs);

                  }
            } finally {
                  if (pst != null)
                        pst.close();
            }
            return (objProfesor);
}
  public List getHorario(String pro_codigo) throws SQLException {
            Horario objHorario=null;
            List listHorario=null;
            PreparedStatement pst = null;
            ResultSet rs = null;
```

```
            String sentencia = " Select * From horario Where
pro_codigo = ? ";
            try {
                  pst = con.prepareStatement(sentencia);
                  pst.setString(1, pro_codigo);
                  listHorario = new LinkedList();
                  rs = pst.executeQuery();
                  while (rs.next()) {
                        listHorario.add(Horario.load(rs));
                  }
            } finally {
                  if (pst != null)
                        pst.close();
            }
            return (listHorario);
}
  public Materia getMateria(String mat_codigo) throws SQLException
{
            Materia objMateria=null;
            PreparedStatement pst = null;
            ResultSet rs = null;
            String sentencia = " Select * From materia Where
mat_codigo = ? ";
            try {
                  pst = con.prepareStatement(sentencia);
                  pst.setString(1, mat_codigo);
                  rs = pst.executeQuery();
                  if (rs.next()) {
                        objMateria=Materia.load(rs);
                  }
```

```
            } finally {
                  if (pst != null)
                        pst.close();
            }
            return (objMateria);
}
  public String getNombreEstudiante(String est_codigo) throws
SQLException {
            PreparedStatement pst = null;
            String cadenaResul=null;
            ResultSet rs = null;
            String sentencia = " Select * From estudiante Where
est_codigo = ? ";
            try {
                  pst = con.prepareStatement(sentencia);
                  pst.setString(1, est_codigo);
                  rs = pst.executeQuery();
                  if (rs.next()) {
                        cadenaResul=rs.getString(23)+"
"+rs.getString(21)+" "+rs.getString(22);
                  }
            } finally {
                  if (pst != null)
                        pst.close();
            }
            return (cadenaResul);
}


 public List getAlumnos(String hor_codigo1) throws SQLException {
```

```
            List listNota=null;
            PreparedStatement pst = null;
            ResultSet rs = null;
            String sentencia = " Select * From notas Where
hor_codigo = ? ";
            try {
                  pst = con.prepareStatement(sentencia);
                  pst.setString(1, hor_codigo1);
                  listNota = new LinkedList();
                  rs = pst.executeQuery();
                  while (rs.next()) {
                        listNota.add(Nota.load(rs));
                  }
            } finally {
                  if (pst != null)
                        pst.close();
            }
            return (listNota);
}

public void updateNota(Nota nota)throws SQLException{
            PreparedStatement pstmt=null;
            try{
                  pstmt=con.prepareStatement("Update Notas Set "+
                  "  par1=? "+
                  " ,par2=? "+
                  " ,par3=? "+
                  " ,def=? "+
                  " Where not_codigo = ? "
            );
```

```
                pstmt.setString(1,nota.getNot_par1());
                pstmt.setString(2,nota.getNot_par2());
                pstmt.setString(3,nota.getNot_par3());
                pstmt.setString(4,nota.getNot_par4());
                pstmt.setString(5,nota.getNot_codigo());
                pstmt.executeUpdate();
        }
        finally{
                if (pstmt!=null)
                        pstmt.close();
        }
    }

    public Horario getHorario_ind(String hor_codigo) throws
SQLException {
        PreparedStatement pst = null;
        Horario horario=null;
        ResultSet rs = null;
        String sentencia = " Select * From horario Where
hor_codigo = ? ";
        try {
                pst = con.prepareStatement(sentencia);
                pst.setString(1, hor_codigo);
                rs = pst.executeQuery();
                while (rs.next()) {
                        horario=Horario.load(rs);
                }
        } finally {
                if (pst != null)
                        pst.close();
```

```
            }
            return (horario);

}

public void updateNotaGrupo(String not_codigo,int parcial,String
valorNota)throws SQLException{
            PreparedStatement pstmt=null;
            String cadena=null;
            if(parcial==1)
                  cadena="Update Notas Set par1=? Where
not_codigo = ? ";
            if(parcial==2)
                  cadena="Update Notas Set par2=? Where
not_codigo = ? ";
            if(parcial==3)
                  cadena="Update Notas Set par3=? Where
not_codigo = ? ";

            try{
                  pstmt=con.prepareStatement(cadena);
                  pstmt.setString(1,valorNota);
                  pstmt.setString(2,not_codigo);
                  pstmt.executeUpdate();
            }
            finally{
                  if (pstmt!=null)
                        pstmt.close();
            }
      }
```

```
	public void updateNotasGeneral(String hor_codigo,Vector
vector,int parcial) throws SQLException {
		List listNota=null;
		PreparedStatement pst = null;
		ResultSet rs = null;
		int i=-1;
		String sentencia = " Select * From notas Where
hor_codigo = ? ";
		try {
			pst = con.prepareStatement(sentencia);
			pst.setString(1, hor_codigo);
			rs = pst.executeQuery();
			while (rs.next()) {
				i++;


	updateNotaGrupo(rs.getString(1),parcial,(String)vector.element
At(i));
			}
		} finally {
			if (pst != null)
				pst.close();
		}

}

public void updateInd_ent(int valor,String hor_codigo)throws
SQLException{
		PreparedStatement pstmt=null;
```

```
            String cadena=null;
            cadena="Update horario Set ind_ent=? Where hor_codigo
= ? ";
            try{
                  pstmt=con.prepareStatement(cadena);
                  pstmt.setString(1,String.valueOf(valor));
                  pstmt.setString(2,hor_codigo);
                  pstmt.executeUpdate();
            }
            finally{
                  if (pstmt!=null)
                        pstmt.close();
            }
      }
}
```

ANEXO 3

CODIGO FUENTE DE PÁGINAS

Alumnos.jsp

```
<%@ page session="true" %>
<%@ page errorPage="/vista/ErrorPage.jsp" %>
<%@ page import="java.util.*" %>
<%@ page import="mvc.model.*" %>
<%@ include file="/WEB-INF/InitModel.jsp" %>
<% String hor_codigo=request.getParameter("hor_codigo");
   String mat_nombre=request.getParameter("mat_nombre");
   Horario horario=new Horario();
   int NumeroParcial=0;
    if(hor_codigo!=null){
         session.setAttribute("hor_codigo_jsp",hor_codigo);

horario=model.getHorario_ind((String)session.getAttribute("hor_codi
go_jsp"));
         NumeroParcial=Integer.parseInt(horario.getHor_ind_ent())+1;
        }else{
        horario=model.getHorario_ind((String)session.getAttribute("hor
_codigo_jsp"));
        NumeroParcial=Integer.parseInt(horario.getHor_ind_ent())+1;
        }
   if(NumeroParcial>3){%>
   <jsp:forward page="NotasListas.jsp" />
 <%}%>
<html>
<head>
```

```
  <title>Lista de Alumnos</title>
  <meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
 </head>
<body bgcolor="#FFFFFF">
<p><font color="#006699" size="6" face="Times New Roman,
Times, serif"><strong><em>Control
  de Notas</em></strong></font> </p>

<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="18%"><font color="#333333" size="2"
face="Verdana, Arial, Helvetica, sans-serif"><strong>Docente
      Codigo:</strong> </font></td>
```

```
    <td width="82%"><em><strong><font color="#006699"
size="3" face="Verdana, Arial, Helvetica, sans-serif"><%=
session.getAttribute("pro_codigo_jsp")
%></font></strong></em></td>
  </tr>
  <tr><td><strong><font color="#333333" size="2"
face="Verdana, Arial, Helvetica, sans-serif">Nombre
      Decente:</font></strong></td>
    <td><em><strong><font color="#006699" size="3"
face="Verdana, Arial, Helvetica, sans-serif"><%=
session.getAttribute("pro_nombre_jsp")
%></font></strong></em></td>
  </tr>
</table>
<hr>
<table width="45%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="25%"><div align="center"><em><font
color="#006699"><strong><a
href="index.jsp">Inicio</a></strong></font></em></div></td>
    <td width="31%"><div align="center"><em><strong><font
color="#006699"><a
href="validar.jsp">MyHorario</a></font></strong></em></div><
/td>
    <td width="44%"><div align="center"><em><font
color="#006699"></font></em></div></td>
  </tr>
</table>
<div align="center">
```

```
  <p><strong><font color="#990000" size="4" face="Arial,
Helvetica, sans-serif"><br>
    <%= mat_nombre %> </font></strong></p>
  <font color="#FFFFFF"><strong><font color="#0099CC" size="3"
face="Arial, Helvetica, sans-serif">Ingreso
  de Notas del Parcial #<font color="#990000"> </font></font>
<font color="#990000"><%= NumeroParcial
%></font></strong></font><br>
</div>
<div align="center"></div>
<div align="center"></div>
<form name="form1" method="get" action="GuardarNotas.jsp">
  <table width="70%" border="1" align="center" cellpadding="0"
cellspacing="0" bordercolor="#FFFFFF">
    <tr>
      <td><table width="100%" border="0" align="left"
cellpadding="0" cellspacing="0">
        <tr bgcolor="#0099CC">
          <td width="5%" valign="top"><div align="left"><img
src="Vista/Imagenes/Esquinas/Esq%20Sup%20Izq.gif" width="25"
height="25"></div></td>
          <td width="20%"><div align="center"><font
color="#FFFFFF"><strong>Nota
              Codigo </strong></font></div></td>
          <td width="53%"><div align="center"><font
color="#FFFFFF"><strong>Nombre
              Estudiante</strong></font></div></td>
          <td width="16%"><div align="center"><font
color="#FFFFFF"><strong>Parcial
              <%= NumeroParcial %></strong></font></div></td>
```

```
          <td width="6%" valign="top"><div align="right"><font
color="#FFFFFF"><img
src="Vista/Imagenes/Esquinas/Esq%20Sup%20Der.gif" width="25"
height="25"></font></div></td>
        </tr>
        <%
  List
listP=model.getAlumnos((String)session.getAttribute("hor_codigo_jsp
"));
  int i=0;

  if (listP != null){
     Iterator itp= listP.iterator();
     while (itp.hasNext()){
        i++;
        Nota nota=(Nota)itp.next();
        String not_codigo=nota.getNot_codigo();
       // String hor_codigo=nota.geHor_codigo();
        String est_codigo=nota.getEst_codigo();
        String
Nombre_estudiante=model.getNombreEstudiante(est_codigo);
        String not_par1=null;
        if(NumeroParcial==1)
        not_par1=nota.getNot_par1();
        if(NumeroParcial==2)
        not_par1=nota.getNot_par2();
        if(NumeroParcial==3)
        not_par1=nota.getNot_par3();
```

```
        %>
          <tr bgcolor="#FFFFFF">
            <td> </td>
            <td><div align="center"><font
color="#0099CC"><strong><font color="#0099CC"><strong><font
size="1" face="Verdana, Arial, Helvetica, sans-serif"><em><%=
not_codigo%></em></font></strong></font><font size="1"
face="Verdana, Arial, Helvetica, sans-serif">
                </font></strong></font></div></td>
            <td><div align="center"><font
color="#0099CC"><strong><font size="1" face="Verdana, Arial,
Helvetica, sans-serif"><em><%= Nombre_estudiante %> </em>
</font></strong></font></div></td>
            <td><div align="center"><font
color="#0099CC"><strong><font size="1" face="Verdana, Arial,
Helvetica, sans-serif"><em>
                <input name="parcial<%=i%>" type="text" value="<%=
not_par1 %>" size="5" maxlength="3">
                </em> </font></strong></font></div></td>
            <td><font color="#003366"> </font></td>
          </tr>
          <%  }} %>
          <tr bgcolor="#0099CC">
            <td><img
src="Vista/Imagenes/Esquinas/Esq%20Inf%20Izq.gif" width="25"
height="25"></td>
            <td valign="bottom"> </td>
            <td> </td>
            <td bgcolor="#0099CC"> </td>
```

```
          <td valign="bottom"><div align="right"><img
src="Vista/Imagenes/Esquinas/Esq%20Inf%20Der.gif" width="25"
height="25"></div></td>
        </tr>
      </table>
      <p> </p></td>
    </tr>
  </table>
  <div align="center"></div>
  <p align="center">
  <% if(i>0){%>
    <input name="Guardar" type="submit" id="Guardar"
value="Guardar<%= NumeroParcial %>">
      <%}%>
  </p>
</form>
<div align="center"><br>
  <% if(i==0){%>
  <font color="#0099CC" size="4" face="Arial, Helvetica, sans-
serif"><strong><a
href="validar.jsp">Continuar</a></strong></font>
  <%}%>
</div>
</body>
</html>
```

Confirmar.jsp

```
<%@ page session="true" %>
<%@ page errorPage="/vista/ErrorPage.jsp" %>
<%@ page import="java.util.*" %>
<%@ page import="mvc.model.*" %>
<%@ include file="/WEB-INF/InitModel.jsp" %>
<%
   String est_nombre=(String)session.getAttribute("est_nombre");
   String not_codigo_jsp=request.getParameter("not_codigo_jsp");
      String not_par1=request.getParameter("not_par1");
      String not_par2=request.getParameter("not_par2");
      String not_par3=request.getParameter("not_par3");
      String not_par4=request.getParameter("not_par4");
      Nota nota=new Nota();
      nota.setNot_par1(not_par1);
      nota.setNot_par2(not_par2);
      nota.setNot_par3(not_par3);
      nota.setNot_par4(not_par4);
      nota.setNot_codigo((String)session.getAttribute("not_codigo_js
p"));
      model.updateNota(nota);
%>
<html>
<head>
  <title>Confirmar</title>
  <meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
 </head>
<body bgcolor="#FFFFFF" text="ffffff">
```

```
<p><font color="#006699" size="6" face="Times New Roman,
Times, serif"><strong><em>Control
  de Notas</em></strong></font> </p>

<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="18%"><font color="#333333" size="2"
face="Verdana, Arial, Helvetica, sans-serif"><strong>Docente
      Codigo:</strong> </font></td>
    <td width="82%"><strong><font color="#009999" size="2"
face="Arial, Helvetica, sans-serif"><%=
session.getAttribute("pro_codigo_jsp")  %></font></strong></td>
  </tr>
  <tr>
    <td><strong><font color="#333333" size="2" face="Verdana,
Arial, Helvetica, sans-serif">Nombre
      Decente:</font></strong></td>
    <td><strong><font color="#009999" size="2" face="Arial,
Helvetica, sans-serif"><%=  session.getAttribute("pro_nombre_jsp")
%></font></strong></td>
  </tr>
</table>
<hr>
<br>
  <table width="50%" border="1" align="center" cellpadding="0"
cellspacing="0" bordercolor="#FFFFFF">
    <tr>
      <td><table width="100%" border="0" align="left"
cellpadding="0" cellspacing="0">
        <tr bgcolor="#0099CC">
```

```
        <td width="16%" valign="top"><div align="left"><img
src="Vista/Imagenes/Esquinas/Esq%20Sup%20Izq.gif" width="25"
height="25"></div></td>
        <td width="12%"><div align="center"><font
color="#FFFFFF"></font></div></td>
        <td width="40%"><div align="center"><font
color="#FFFFFF"></font></div></td>
        <td width="32%" valign="top"><div align="right"><font
color="#FFFFFF"><img
src="Vista/Imagenes/Esquinas/Esq%20Sup%20Der.gif" width="25"
height="25"></font></div></td>
      </tr>
      <tr bgcolor="#FFFFFF">
        <td colspan="2"><div align="center"><em><strong><font
color="#006699" size="2" face="Arial, Helvetica, sans-
serif">Nombre
            Etudiante</font></strong></em></div></td>
        <td colspan="2"><div align="center"><font color="#990000"
size="2" face="Verdana, Arial, Helvetica, sans-serif"><strong><%=
est_nombre %> </strong></font></div></td>
      </tr>
      <tr bgcolor="#FFFFFF">
        <td colspan="2"><div align="center"><em><strong><font
color="#006699"><font size="2"><font size="3">Parcia
            1 </font></font></font></strong></em></div></td>
        <td colspan="2"><div align="center"><font color="#990000"
size="2" face="Verdana, Arial, Helvetica, sans-serif"><strong><%=
not_par1 %></strong></font></div></td>
      </tr>
      <tr bgcolor="#FFFFFF">
```

```
        <td colspan="2"><div align="center"><em><strong><font
color="#006699"><font size="2"><font size="3">Parcia
            2 </font></font></font></strong></em></div></td>
        <td colspan="2"><div align="center"><font color="#990000"
size="2" face="Verdana, Arial, Helvetica, sans-serif"><strong><%=
not_par2 %></strong></font></div></td>
      </tr>
      <tr bgcolor="#FFFFFF">
        <td colspan="2"><div align="center"><em><strong><font
color="#006699"><font size="2"><font size="3">Parcia
            3 </font></font></font></strong></em></div></td>
        <td colspan="2"><div align="center"><font color="#990000"
size="2" face="Verdana, Arial, Helvetica, sans-serif"><strong><%=
not_par3 %></strong></font></div></td>
      </tr>
      <tr bgcolor="#FFFFFF">
        <td colspan="2"><div align="center"><em><strong><font
color="#006699"
size="3">Def</font></strong></em></div></td>
        <td colspan="2"><div align="center"><font
size="2"><strong><font color="#006699"><em>
            </em> </font></strong></font><font color="#990000"
size="2" face="Verdana, Arial, Helvetica, sans-serif"><strong><%=
not_par4 %></strong></font></div></td>
      </tr>
      <tr bgcolor="#FFFFFF">
        <td colspan="2"><div align="right"> </div></td>
        <td><div align="center"><font
size="4"><em><strong></strong></em></font></div></td>
```

```
        <td valign="bottom"><div align="center"><font
size="4"><em><strong><font color="#990000"><a
href="Alumnos.jsp">Continuar</a></font></strong></em></font
></div></td>
      </tr>
      <tr bgcolor="#0099CC">
        <td><img
src="Vista/Imagenes/Esquinas/Esq%20Inf%20Izq.gif" width="25"
height="25"></td>
        <td valign="bottom"> </td>
        <td> </td>
        <td valign="bottom"><div align="right"><img
src="Vista/Imagenes/Esquinas/Esq%20Inf%20Der.gif" width="25"
height="25"></div></td>
      </tr>
    </table>
      <p> </p></td>
  </tr>
 </table>
<p> </p>
<p> </p>
</body>
</html>
```

GuardarNotas.jsp

```
<%@ page session="true" %>
<%@ page errorPage="/vista/ErrorPage.jsp" %>
<%@ page import="java.util.*" %>
<%@ page import="mvc.model.*" %>
<%@ include file="/WEB-INF/InitModel.jsp" %>
<%

 Vector vector=new Vector();
 for (int i=1;i<=3;i++){
 vector.add(request.getParameter("parcial"+String.valueOf(i)));
 }
 if(request.getParameter("Guardar").equals("Guardar1")){

model.updateNotasGeneral((String)session.getAttribute("hor_codigo_
jsp"),vector,1);

model.updateInd_ent(1,(String)session.getAttribute("hor_codigo_jsp"
));
        }
 if(request.getParameter("Guardar").equals("Guardar2")){

model.updateNotasGeneral((String)session.getAttribute("hor_codigo_
jsp"),vector,2);

model.updateInd_ent(2,(String)session.getAttribute("hor_codigo_jsp"
));
        }
 if(request.getParameter("Guardar").equals("Guardar3")){
```

```
model.updateNotasGeneral((String)session.getAttribute("hor_codigo_
jsp"),vector,3);

model.updateInd_ent(3,(String)session.getAttribute("hor_codigo_jsp"
));
        }

 %>
<html>
<head>
  <title>Lista de Alumnos</title>
  <meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
 </head>
<body bgcolor="#FFFFFF">
<p><font color="#006699" size="6" face="Times New Roman,
Times, serif"><strong><em>Control
  de Notas</em></strong></font> </p>

<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="18%"><font color="#333333" size="2"
face="Verdana, Arial, Helvetica, sans-serif"><strong>Docente
      Codigo:</strong> </font></td>
    <td width="82%"><em><strong><font color="#006699"
size="3" face="Verdana, Arial, Helvetica, sans-serif"><%=
session.getAttribute("pro_codigo_jsp")
%></font></strong></em></td>
  </tr>
```

```
  <tr>
    <td><strong><font color="#333333" size="2" face="Verdana,
Arial, Helvetica, sans-serif">Nombre
      Decente:</font></strong></td>
    <td><em><strong><font color="#006699" size="3"
face="Verdana, Arial, Helvetica, sans-serif"><%=
session.getAttribute("pro_nombre_jsp")
%></font></strong></em></td>
  </tr>
</table>
<hr>
<table width="45%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="25%"><div align="center"><em><font
color="#006699"><strong><a
href="index.jsp">Inicio</a></strong></font></em></div></td>
    <td width="31%"><div align="center"><em><strong><font
color="#006699"><a
href="validar.jsp">MyHorario</a></font></strong></em></div><
/td>
    <td width="44%"><div align="center"><em><font
color="#006699"></font></em></div></td>
  </tr>
</table>
<p><%  for (int i=0;i<3;i++){%>
        <%= vector.elementAt(i) %>
  <%} %> </p></body>
</html>
```

Index.jsp

```
<%@ page session="true" %>
<%@ page errorPage="/vista/ErrorPage.jsp" %>
<%@ page import="java.util.*" %>
<%@ page import="mvc.model.*" %>
<%@ include file="/WEB-INF/InitModel.jsp" %>
<% String estado=request.getParameter("estado");
   session.removeAttribute("pro_codigo_jsp");
%>

<html>
<head>
  <title>CONTROL DE NOTAS DE ESTUDIANTES</title>
  <meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
 </head>
<body bgcolor="#FFFFFF" text="ffffff" link=ffff11 vlink=ffff11
leftmargin="0" topmargin="0" marginwidth="0" marginheight="0">
<table width="100%" border="0" cellpadding="0" cellspacing="0"
bgcolor="#993333">
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
</table>
<p><font color="#006699" size="6" face="Times New Roman,
Times, serif"><strong><em>Control
  de Notas</em></strong></font> </p>
```

```
<table width="75%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="38%" rowspan="2"><img
src="Vista/Imagenes/IFRS.png" width="306" height="260"></td>
    <td width="62%"><table width="75%" border="0" align="center"
cellpadding="0" cellspacing="0">
      <tr>
        <td><div align="center">
            <form name="form1" method="post" action="validar.jsp">
              <table width="75%" border="0" cellspacing="3"
cellpadding="3">
                <tr>
                  <td width="45%"><div align="right"><font
color="#006699" size="2" face="Verdana, Arial, Helvetica, sans-
serif"><strong>Codigo</strong></font></div></td>
```

```
<td width="55%"><div align="left">
   <input type="text" name="pro_codigo_jsp">
```

```
                </div></td>
              </tr>
              <tr>
                <td><div align="right"><font color="#006699"
size="2" face="Verdana, Arial, Helvetica, sans-
serif"><strong>Contrase&ntilde;a</strong></font></div></td>
                <td><div align="left">
                  <input type="password" name="pro_cedula_jsp">
                </div></td>
              </tr>
              <tr>
                <td colspan="2"> <div align="center">
                  <input type="reset" name="Reset" value="Reset">
                  <input name="Aceptar" type="submit" id="Aceptar"
value="Aceptar" >
                </div></td>
              </tr>
              <tr>
                <td> </td>
                <td> </td>
              </tr>
            </table>
          </form>
        </div></td>
      </tr>
    </table></td>
  </tr>
  <tr>
    <td><div align="center"><em><strong><font color="#993300"
size="2" face="Arial, Helvetica, sans-serif">
```

```
        <%if(estado!=null){%>
        <%= estado %>
        <% } %>
        </font></strong></em></div></td>
  </tr>
</table>
<p align="center"><em><strong><font color="#993300" size="2"
face="Arial, Helvetica, sans-serif">
  </font></strong></em></p>
<table width="100%" border="0" cellpadding="0" cellspacing="0"
bgcolor="#993333">
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
</table>
<p align="center"> </p>
</body> </html>
```

Modificar.jsp

```
<%@ page session="true" %>
<%@ page errorPage="/vista/ErrorPage.jsp" %>
<%@ page import="java.util.*" %>
<%@ page import="mvc.model.*" %>
<%@ include file="/WEB-INF/InitModel.jsp" %>

<% String est_nombre=request.getParameter("est_nombre");
   String not_codigo_jsp=request.getParameter("not_codigo_jsp");
      String not_par1=request.getParameter("not_par1");
      String not_par2=request.getParameter("not_par2");
      String not_par3=request.getParameter("not_par3");
      String not_par4=request.getParameter("not_par4");
      session.setAttribute("est_nombre",est_nombre);
      session.setAttribute("not_codigo_jsp",not_codigo_jsp);
%>
<html>

<head>
  <title>Lista de Alumnos</title>
  <meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
  <script language="JavaScript1.4" >
 function Validar(){
// form1.action
 form1.submit();
 return void;
 }
</script>
```

```
 </head>
<body bgcolor="#FFFFFF" text="ffffff">
<p><font color="#006699" size="6" face="Times New Roman,
Times, serif"><strong><em>Control
  de Notas</em></strong></font> </p>

<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="18%"><font color="#333333" size="2"
face="Verdana, Arial, Helvetica, sans-serif"><strong>Docente
      Codigo:</strong> </font></td>
    <td width="82%"><strong><font color="#009999" size="2"
face="Arial, Helvetica, sans-serif"><%=
session.getAttribute("pro_codigo_jsp")  %></font></strong></td>
  </tr>
  <tr>
<td><strong><font color="#333333" size="2" face="Verdana, Arial,
Helvetica, sans-serif">Nombre
      Decente:</font></strong></td>
    <td><strong><font color="#009999" size="2" face="Arial,
Helvetica, sans-serif"><%=  session.getAttribute("pro_nombre_jsp")
%></font></strong></td>
  </tr>
</table>

<hr>
<form name="form1" method="post" action="Confirmar.jsp">
  <table width="60%" border="1" align="center" cellpadding="0"
cellspacing="0" bordercolor="#FFFFFF">
    <tr>
```

```
    <td><table width="100%" border="0" align="left"
cellpadding="0" cellspacing="0">
      <tr bgcolor="#0099CC">
        <td width="4%" valign="top"><div align="left"><img
src="Vista/Imagenes/Esquinas/Esq%20Sup%20Izq.gif" width="25"
height="25"></div></td>
        <td width="20%"><div align="center"><font
color="#FFFFFF"></font></div></td>
        <td width="27%"><div align="center"><font
color="#FFFFFF"></font></div></td>
        <td width="5%" valign="top"><div align="right"><font
color="#FFFFFF"><img
src="Vista/Imagenes/Esquinas/Esq%20Sup%20Der.gif" width="25"
height="25"></font></div></td>
      </tr>
      <tr bgcolor="#FFFFFF">
        <td> </td>
        <td><div align="center"><em><strong><font
color="#006699" size="2" face="Arial, Helvetica, sans-
serif">Nombre
            Etudiante</font></strong></em></div></td>
        <td><font color="#990000" size="2" face="Verdana, Arial,
Helvetica, sans-serif"><strong><%= est_nombre
%> </strong></font></td>
        <td> </td>
      </tr>
      <tr bgcolor="#FFFFFF">
        <td> </td>
        <td><div align="center"><em><strong><font
color="#006699"><font size="2"><font size="3">Parcia
```

```
            1 </font></font></font></strong></em></div></td>
          <td><input type="text" name="not_par1" <% if
(not_par1!=null) { %> value="<%= not_par1 %>" <%}%>></td>
          <td> </td>
        </tr>
        <tr bgcolor="#FFFFFF">
          <td> </td>
          <td><div align="center"><em><strong><font
color="#006699"><font size="2"><font size="3">Parcia
            2 </font></font></font></strong></em></div></td>
          <td><input type="text" name="not_par2" <% if
(not_par2!=null) { %> value="<%= not_par2 %>" <%}%>></td>
          <td> </td>
        </tr>
        <tr bgcolor="#FFFFFF">
          <td> </td>
          <td><div align="center"><em><strong><font
color="#006699"><font size="2"><font size="3">Parcia
            3 </font></font></font></strong></em></div></td>
          <td><input type="text" name="not_par3" <% if
(not_par3!=null) { %> value="<%= not_par3 %>" <%}%>></td>
          <td> </td>
        </tr>
        <tr bgcolor="#FFFFFF">
          <td> </td>
          <td><div align="center"><em><strong><font
color="#006699"
size="3">Def</font></strong></em></div></td>
          <td><div align="left"><font size="2"><strong><font
color="#006699"><em>
```

```
            <input type="text" name="not_par4" <% if
(not_par4!=null) { %> value="<%= not_par4 %>" <%}%>>
            </em> </font></strong></font></div></td>
          <td><font color="#003366"> </font></td>
        </tr>
        <tr bgcolor="#FFFFFF">
          <td> </td>
          <td valign="bottom"><div align="right">
            <input name="Cancelar" type="button" id="Cancelar"
value="Cancelar" onClick="Validar();" >
            </div></td>
          <td><input type="submit" name="Submit"
value="Guardar"></td>
          <td valign="bottom"> </td>
        </tr>
        <tr bgcolor="#0099CC">
          <td><img
src="Vista/Imagenes/Esquinas/Esq%20Inf%20Izq.gif" width="25"
height="25"></td>
          <td valign="bottom"> </td>
          <td> </td>
          <td valign="bottom"><div align="right"><img
src="Vista/Imagenes/Esquinas/Esq%20Inf%20Der.gif" width="25"
height="25"></div></td>
        </tr>
      </table>
      <p> </p></td>
    </tr>
  </table>
</form>
```

```
<p> </p>
<p> </p>
</body>
</html>
```

NotasListas.jsp

```
<%@ page session="true" %>
<%@ page errorPage="/vista/ErrorPage.jsp" %>
<%@ page import="java.util.*" %>
<%@ page import="mvc.model.*" %>
<%@ include file="/WEB-INF/InitModel.jsp" %>
<html>
<head>
  <title>Lista de Alumnos</title>
  <meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
 </head>
<body bgcolor="#FFFFFF">
<p><font color="#006699" size="6" face="Times New Roman,
Times, serif"><strong><em>Control
  de Notas</em></strong></font> </p>

<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="18%"><font color="#333333" size="2"
face="Verdana, Arial, Helvetica, sans-serif"><strong>Docente
      Codigo:</strong> </font></td>
    <td width="82%"><em><strong><font color="#006699"
size="3" face="Verdana, Arial, Helvetica, sans-serif"><%=
session.getAttribute("pro_codigo_jsp")
%></font></strong></em></td>
  </tr>
  <tr>
```

```
    <td><strong><font color="#333333" size="2" face="Verdana,
Arial, Helvetica, sans-serif">Nombre
      Decente:</font></strong></td>
    <td><em><strong><font color="#006699" size="3"
face="Verdana, Arial, Helvetica, sans-serif"><%=
session.getAttribute("pro_nombre_jsp")
%></font></strong></em></td>
  </tr>
</table>
<hr>
<table width="45%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="25%"><div align="center"><em><font
color="#006699"><strong><a
href="index.jsp">Inicio</a></strong></font></em></div></td>
    <td width="31%"><div align="center"><em><strong><font
color="#006699"><ahref="validar.jsp">MyHorario</a></font></str
ong></em></div></td>
```

```
    <td width="44%"><div align="center"><em><font
color="#006699"></font></em></div></td>
  </tr>
</table>
<p><strong><font color="#990000" face="Arial, Helvetica, sans-
serif">Ya se han ingresado todos
  los parciales</font></strong></p>
<p> </p>
<p align="center"> <font color="#0099CC" size="4" face="Arial,
Helvetica, sans-serif"><strong><a
href="validar.jsp">Continuar</a></strong></font>
</p>
</body>
</html>
```

Validar.jsp

```
<%@ page session="true" %>
<%@ page errorPage="/vista/ErrorPage.jsp" %>
<%@ page import="java.util.*" %>
<%@ page import="mvc.model.*" %>
<%@ include file="/WEB-INF/InitModel.jsp" %>
<% String pro_codigo_jsp;
   String pro_cedula_jsp;
        if((String)session.getAttribute("pro_codigo_jsp")!=null){
       pro_codigo_jsp=(String)session.getAttribute("pro_codigo_jsp");

       pro_cedula_jsp=(String)session.getAttribute("pro_cedula_jsp");
              }else{
       pro_codigo_jsp=request.getParameter("pro_codigo_jsp");
              pro_cedula_jsp=request.getParameter("pro_cedula_jsp");
              }
  %>
<html>
<head>
  <title>Validar Jsp</title>
  <meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
 </head>
<body bgcolor="#FFFFFF" text="ffffff">
<p><font color="#006699" size="6" face="Times New Roman,
Times, serif"><strong><em>Control
  de Notas</em></strong></font> </p>
<p>
  <% Profesor objProfesor=null;
```

```
objProfesor=model.validarProfesor(pro_codigo_jsp,pro_cedula_jsp);
if(objProfesor==null){
%>
</p>
<jsp:forward page="index.jsp?estado=Usuario no existe" />
<%}else{ %>
<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="18%"><font color="#333333" size="2"
face="Verdana, Arial, Helvetica, sans-serif"><strong>Docente
      Codigo:</strong> </font></td>
    <td width="82%"><em><strong><font color="#0099CC"
size="2" face="Verdana, Arial, Helvetica, sans-serif"><%=
objProfesor.getPro_codigo()  %></font></strong></em></td>
  </tr><tr> <td><strong><font color="#333333" size="2"
face="Verdana, Arial, Helvetica, sans-serif">Nombre
      Decente:</font></strong></td>
    <td><em><strong><font color="#0099CC" size="2"
face="Verdana, Arial, Helvetica, sans-serif"><%=
objProfesor.getPro_nombre() %></font></strong></em></td>
  </tr>
</table>
<p>
  <%
  session.setAttribute("pro_codigo_jsp",pro_codigo_jsp);

session.setAttribute("pro_nombre_jsp",objProfesor.getPro_nombre())
;
  session.setAttribute("pro_cedula_jsp",pro_cedula_jsp);
```

```
  } %>
  <br>
<hr>
<table width="45%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="25%"><div align="center"><em><font
color="#006699"><strong><a
href="index.jsp">Inicio</a></strong></font></em></div></td>
    <td width="31%"><div align="center"><em><strong><font
color="#006699">MyHorario</font></strong></em></div></td>
    <td width="44%"><div align="center"><em><font
color="#006699"></font></em></div></td>
  </tr>
</table>
<table width="75%" border="1" align="center" cellpadding="0"
cellspacing="0" bordercolor="#FFFFFF">
  <tr>
    <td><table width="100%" border="0" align="left"
cellpadding="0" cellspacing="0">
        <tr bgcolor="#0099CC">
          <td><div align="left"><img
src="Vista/Imagenes/Esquinas/Esq%20Sup%20Izq.gif" width="25"
height="25"></div></td>
          <td><div align="center"><font
color="#FFFFFF"><strong>Codigo
Materia</strong></font></div></td>
          <td><div align="center"><font
color="#FFFFFF"><strong>Nombre
Materia</strong></font></div></td>
```

```
        <td><div align="center"><font
color="#FFFFFF"><strong>Numero
Horas</strong></font></div></td>
        <td><div align="center"><font
color="#FFFFFF"><strong>Grupo</strong></font></div></td>
        <td><div align="right"><font color="#FFFFFF"><img
src="Vista/Imagenes/Esquinas/Esq%20Sup%20Der.gif" width="25"
height="25"></font></div></td>
      </tr>
      <%
  List listP=model.getHorario(pro_codigo_jsp);
  if (listP != null){
    Iterator itp= listP.iterator();
    while (itp.hasNext()){
      Horario horario=(Horario) itp.next();
      String hor_codigo=horario.getHor_codigo();
            String mat_codigo=horario.getMat_codigo();
            Materia objMateria=model.getMateria(mat_codigo);
            String mat_nombre=objMateria.getMat_nombre();
            String mat_horas=objMateria.getMat_horas();
            String hor_grupo=horario.getHor_grupo();
      %>
      <tr bgcolor="#FFFFFF">
        <td> </td>
        <td><div align="center"><font color="#003366" size="2"
face="Verdana, Arial, Helvetica, sans-serif"><em><strong><%=
hor_codigo %></strong></em></font></div></td>
        <td><div align="center"><font color="#003366" size="2"
face="Verdana, Arial, Helvetica, sans-serif"><em><strong><a
href="Alumnos.jsp?hor_codigo=<%= hor_codigo
```

```
%>&amp;mat_nombre=<%= mat_nombre %>" ><%=
mat_nombre %></a></strong></em></font></div></td>
        <td><div align="center"><font color="#003366" size="2"
face="Verdana, Arial, Helvetica, sans-serif"><em><strong><%=
mat_horas %></strong></em></font></div></td>
        <td><div align="center"><font color="#003366" size="2"
face="Verdana, Arial, Helvetica, sans-serif"><em><strong><%=
hor_grupo %></strong></em></font></div></td>
        <td><font color="#003366"> </font></td>
      </tr>
      <% }} %>
      <tr bgcolor="#0099CC">
        <td><img
src="Vista/Imagenes/Esquinas/Esq%20Inf%20Izq.gif" width="25"
height="25"></td>
        <td> </td>
        <td> </td>
        <td bgcolor="#0099CC"> </td>
        <td> </td>
        <td><div align="right"><img
src="Vista/Imagenes/Esquinas/Esq%20Inf%20Der.gif" width="25"
height="25"></div></td>
      </tr>
    </table>
    <p> </p>
</td>
  </tr>
</table>
<p> </p>
<p> </p></body></html>
```