Arquitectura y Diseño de sistemas Web y C/S

Práctica Final

Convocatoria Ordinaria

(Evaluación continua)

Curso 2020/2021

Marina Benito Silvestre

Álvaro Blanco Álvarez

Cristina Elena Dascalu

Alberto González Martínez

Aarón Iglesias Chorro

Contenido

[ANÁLISIS DEL PROBLEMA 3](#_Toc61880420)

[PROBLEMAS ENCONTRADOS Y SOLUCIONES APORTADAS 4](#_Toc61880421)

[IMPLEMENTACIÓN 4](#_Toc61880422)

[BIBLIOGRAFÍA 4](#_Toc61880423)

# ANÁLISIS DEL PROBLEMA

Este proyecto trata sobre una empresa, que tiene una plantilla personal que pone a disposición de distintos clientes, en el cual se necesita desarrollar un sistema informático para la gestión de los costes del personal.

La aplicación está dividida en dos grupos, el personal de recursos humanos y el personal de los clientes.

Cuando se habla del personal de recursos humanos debe tener el control de las empresas, proyectos, trabajadores y calendarios, así como atender todas las peticiones de los trabajadores. En cualquier momento se podrá solicitar a través de la aplicación un informe sobre el cumplimiento de las jornadas de trabajo de la empresa, proyecto o empleado.

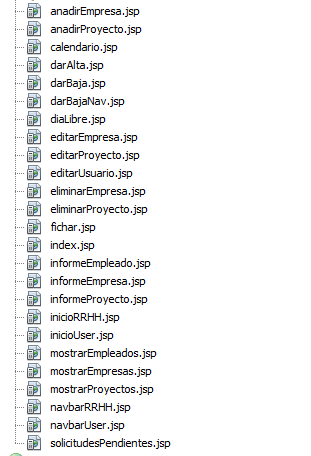
Cuando hablamos de la aplicación, nos referimos a llevar un control diario sobre el trabajo, los empleados harán un marcaje de su entrada y salida, dirán las jornadas realizadas en el día anterior en base al proyecto, indican el numero de horas de la jornada que dedicaran a cada proyecto asignado, también solicitaran vacaciones, horas, días libres en el calendario.

Estos cambios lo tienen que aprobar el personal de recursos humanos, una vez conseguida esta aprobación el empleado recibirá un mensaje con la respuesta.

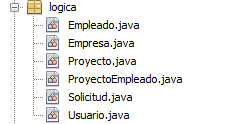
# PROBLEMAS ENCONTRADOS Y SOLUCIONES APORTADAS

# IMPLEMENTACIÓN

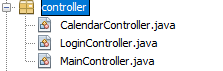
Para la elaboración de este software se ha utilizado el patrón Modelo-Vista-Controlador. En la clase Modelo hemos incluido las funcionalidades del sistema y clases con sus respectivos métodos para la conexión con la base de datos. La vista, por su parte, incluye las páginas HTML con sus formularios con los cuales el cliente puede interactuar. Por último, en el controlador hemos incluido Servlets para capturar las peticiones del cliente y comunicarlas al modelo para ofrecer las vistas adecuadas.



*Figura 1. Vista*



*Figura 2. Modelo*



*Figura 3. Controlador*

La parte del Modelo, como hemos dicho anteriormente, se conecta a la base de datos mediante unas clases encargadas de dicha conexión. Además, el software implementa funciones propias de la parte de recursos humanos y funciones propias de la parte de empleados normales. A continuación, vamos a ver más en detalle estas funcionalidades.

## Implementación Recursos Humanos

### Mostrar Empleados

Se ha implementado un método para mostrar todos los empleados dados de alta en la base de datos.



*Figura 4. mostrarEmpleados*

Por otro lado, tenemos una vista con un botón Mostrar Empleados que incluye una referencia a otra vista mostrarEmpleados.jsp.



*Figura 5. Botón mostrar Empleados*

Cuando el usuario pulsa el botón, el controlador captura la acción y realiza la operación correspondiente que, en este caso, implica mostrar al usuario la vista correspondiente.



*Figura 6. Capturar acción y devolver resultado*.

Posteriormente, se abre una nueva vista con la lista de empleados. En esta vista representada mediante un jsp llamamos al método mostrarEmpleados.



*Figura 7. Vista de los empleados*

En esta misma vista, el usuario tiene opciones de añadir un nuevo empleado, eliminarlo o editarlo.

#### Añadir

Para añadir empleados se ha implementado el método darAlta(Usuario). Este método se basa en hacer un insert en la base de datos de un objeto de tipo usuario con unos atributos definidos previamente.



*Figura 8. Dar Alta*

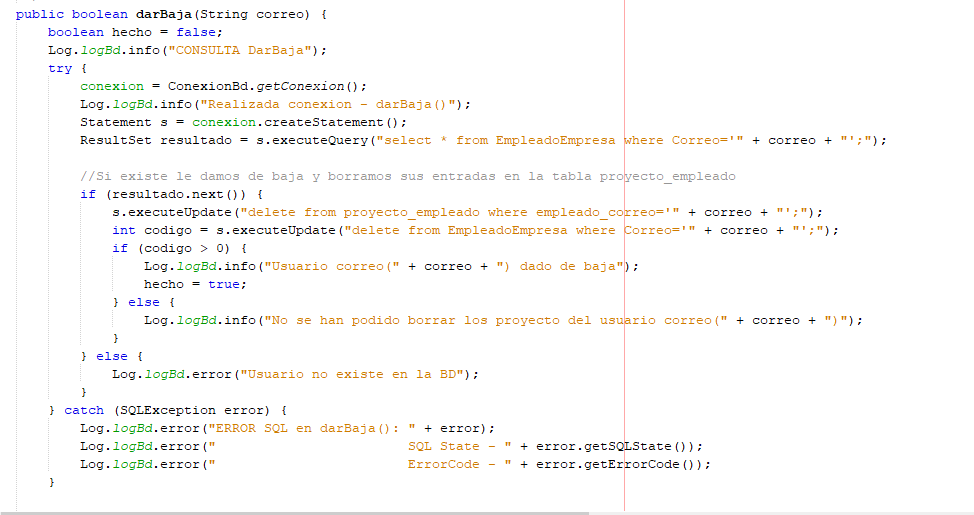
Al presionar el botón añadir desde la vista anterior, el usuario obtiene un formulario donde ha de ingresar los datos del usuario que desea dar de alta. Al presionar el botón Enviar, el controlador recibe la acción y llama al método darAlta pasándole un objeto de tipo usuario con los atributos del formulario que nuestro usuario ha enviado.



*Figura 9. Controlador captura la acción altaEmpleado*

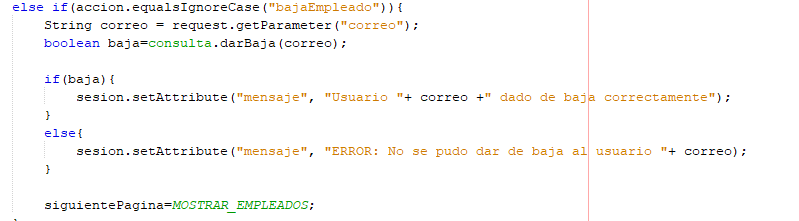
#### Eliminar

Para eliminar un empleado tenemos el método darBaja(correo) que realiza una consulta en la base de datos para buscar el usuario que tiene asignado dicho correo. Si lo encuentra, se realiza una consulta para eliminar el usuario de la base de datos.



*Figura 10. darBaja*

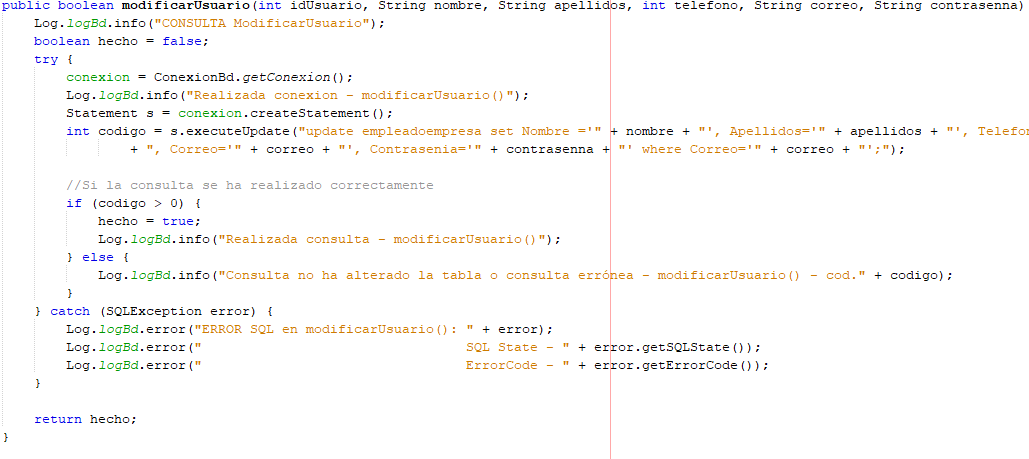
El usuario selecciona un empleado a eliminar desde la vista y, esta acción, se pasa al controlador. El controlador obtiene de la vista el correo del empleado a eliminar y realiza una llamada al método darBaja pasándole como párametro el correo obtenido.



*Figura 11. Controlador captura acción bajaEmpleado*

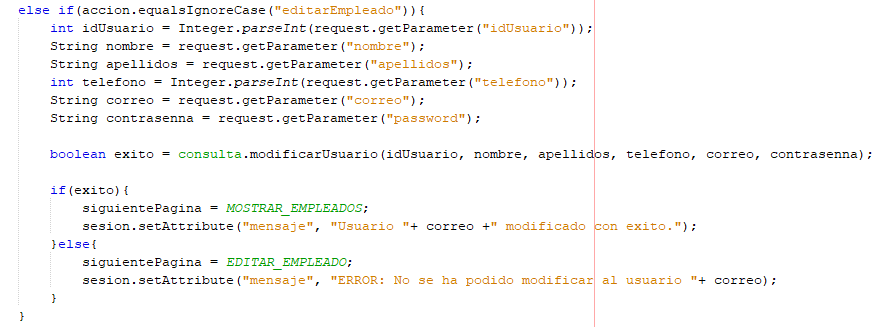
#### Editar

Para editar un empleado se ha utilizado el método modificarUsuario que recibe como parámetros los nuevos valores del usuario a modificar. Se realiza un UPDATE en la base de datos de los valores que se desean modificar del usuario identificado por su correo.



*Figura 12. modificarUsuario*

El usuario selecciona un empleado desde la vista mostrarEmpleados y pulsa el botón Editar. Este botón genera una nueva vista con un formulario para introducir los nuevos datos del usuario. El controlador captura la acción junto con los nuevos valores introducidos en el formulario y llama al método modificarUsuario pasándole dichos valores.



### Mostrar Proyectos

Se ha implementado un método para mostrar todos los proyectos dados de alta en la base de datos.



Por otro lado, tenemos una vista con un botón Mostrar Proyetos que incluye una referencia a otra vista mostrarProyectos.jsp.



Posteriormente, se abre una nueva vista con la lista de proyectos. En esta vista representada mediante un jsp llamamos al método mostrarProyectos.



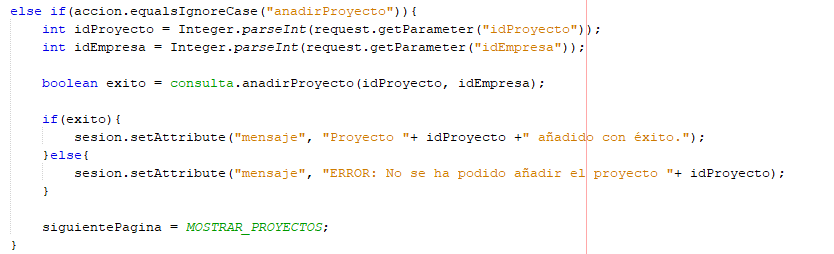
En esta misma vista, el usuario tiene opciones de añadir un nuevo proyecto, eliminarlo o editarlo.

#### Añadir

Para añadir proyectos se ha implementado el método anadirProyecto. Este método se basa en hacer un insert en la base de datos de un nuevo proyecto.

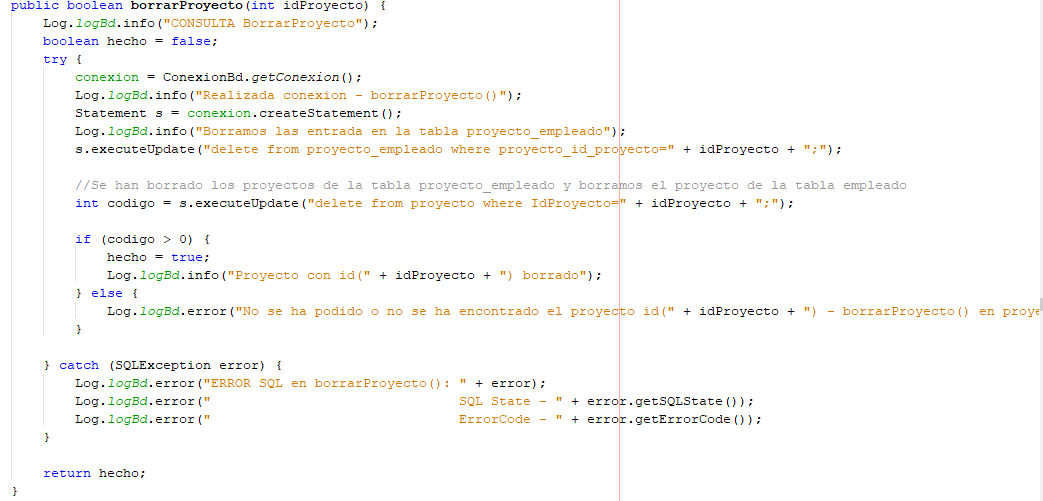


Al presionar el botón añadir desde la vista anterior, el usuario obtiene un formulario donde ha de ingresar los datos del proyecto que desea añadir. Al presionar el botón Enviar, el controlador recibe la acción y llama al método anadirProyecto pasándole los atributos del formulario que nuestro usuario ha enviado.

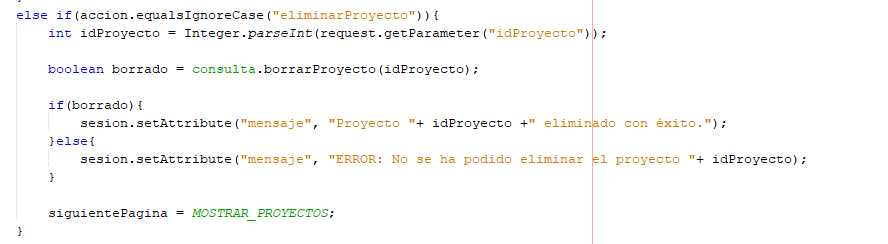


#### Eliminar

Para eliminar un proyecto tenemos el método borrarProyecto(id) que realiza una consulta en la base de datos para eliminar los registros de las tablas que dependen del proyecto identificado por el parámetro id. Una vez realizado este paso, se realiza otra consulta para borrar el proyecto de la base de datos.

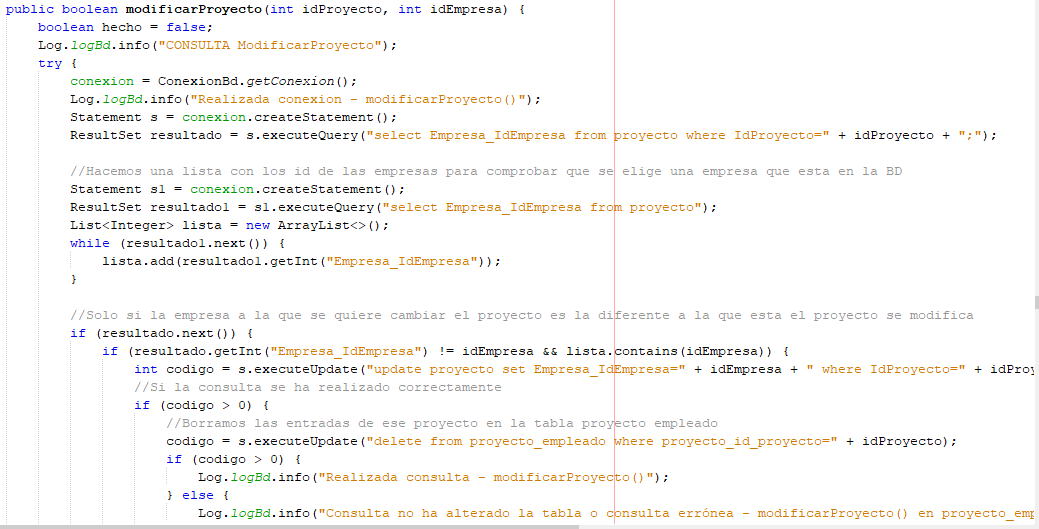


El usuario selecciona un proyecto a eliminar desde la vista y, esta acción, se pasa al controlador. El controlador obtiene de la vista el id del proyecto a eliminar y realiza una llamada al método borrarProyecto pasándole como parámetro los datos obtenidos.

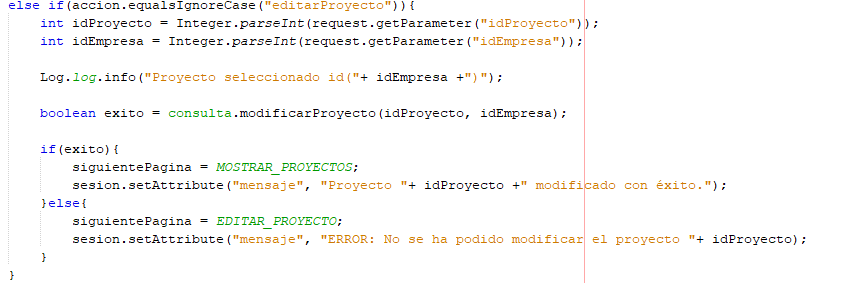


#### Editar

Para editar un proyecto se ha utilizado el método modificarProyecto que recibe como parámetros los nuevos valores del proyecto a modificar. Se realiza un UPDATE en la base de datos de los valores que se desean modificar del proyecto identificado por su id.



El usuario selecciona un proyecto desde la vista mostrarProyectos y pulsa el botón Editar. Este botón genera una nueva vista con un formulario para introducir los nuevos datos del proyecto. El controlador captura la acción junto con los nuevos valores introducidos en el formulario y llama al método modificarProyecto pasándole dichos valores.



### Mostrar Empresas

Se ha implementado un método para mostrar todas las empresas dadas de alta en la base de datos.



Por otro lado, tenemos una vista con un botón Mostrar Empresas que incluye una referencia a otra vista mostrarEmpresas.jsp.



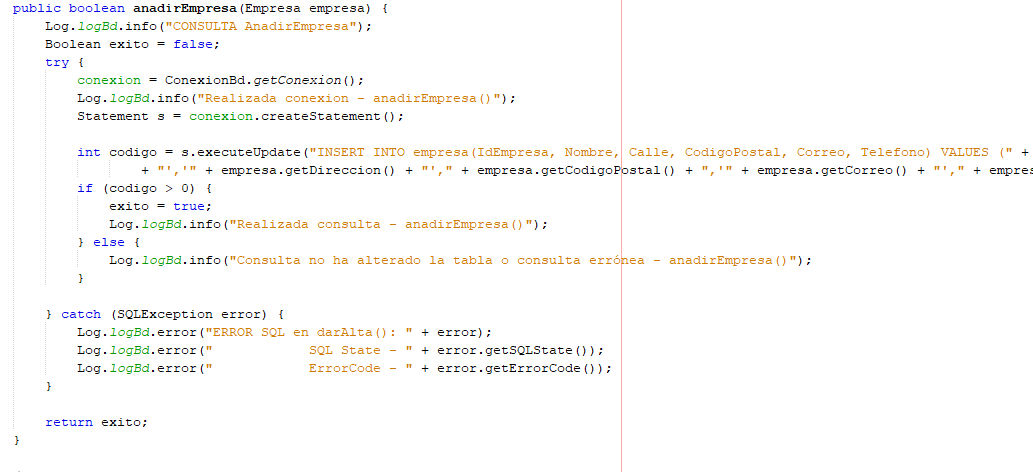
Posteriormente, se abre una nueva vista con la lista de empresas. En esta vista representada mediante un jsp llamamos al método mostrarEmpresas.



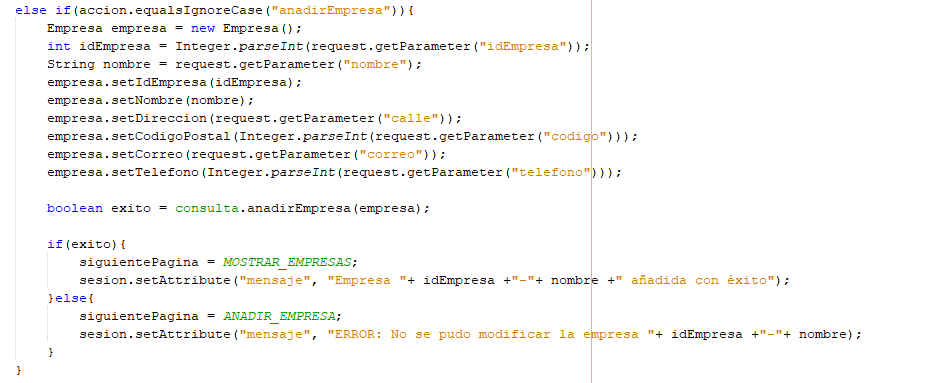
En esta misma vista, el usuario tiene opciones de añadir una nueva empresa, eliminarla o editarla.

#### Añadir

Para añadir empresas se ha implementado el método anadirEmpresa. Este método se basa en hacer un insert en la base de datos de un objeto de tipo empresa con unos atributos definidos previamente.



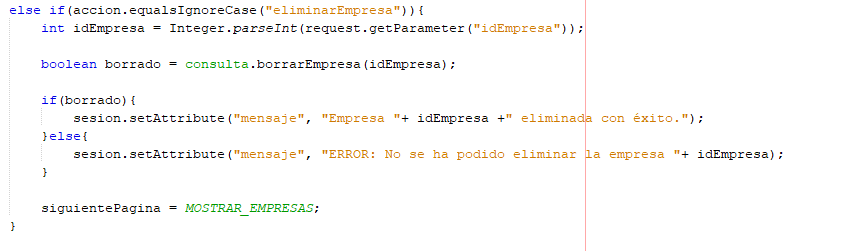
Al presionar el botón añadir desde la vista anterior, el usuario obtiene un formulario donde ha de ingresar los datos de la empresa que desea dar de alta. Al presionar el botón Enviar, el controlador recibe la acción y llama al método anadirEmpresa pasándole un objeto de tipo empresa con los atributos del formulario que nuestro usuario ha enviado.



#### Eliminar

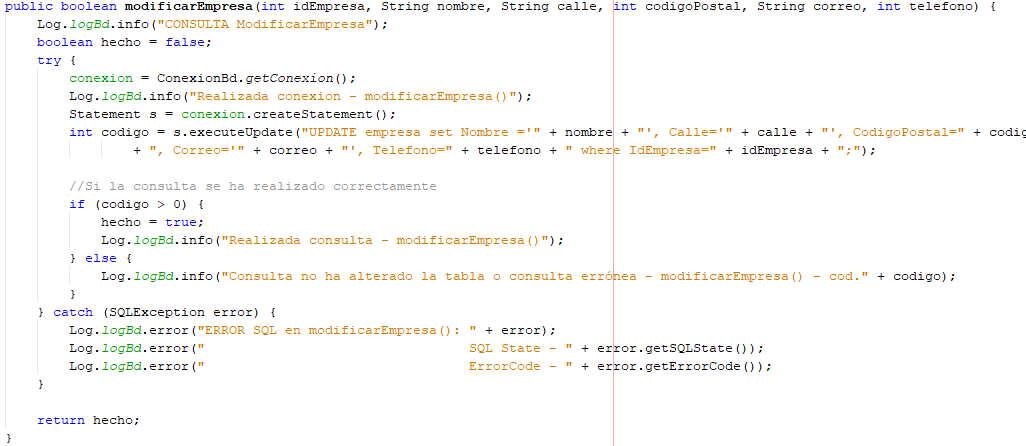
Para eliminar una empresa tenemos el método borrarEmpresa(id) que realiza una consulta en la base de datos para eliminar todos los registros de las tablas que dependen de la empresa identificada por el id. Posteriormente, se realiza otra consulta para eliminar la empresa indicada. 

El usuario selecciona una empresa a eliminar desde la vista y, esta acción, se pasa al controlador. El controlador obtiene de la vista el identificador de la empresa a eliminar y realiza una llamada al método borrarEmpresa pasándole como párametro el id obtenido.

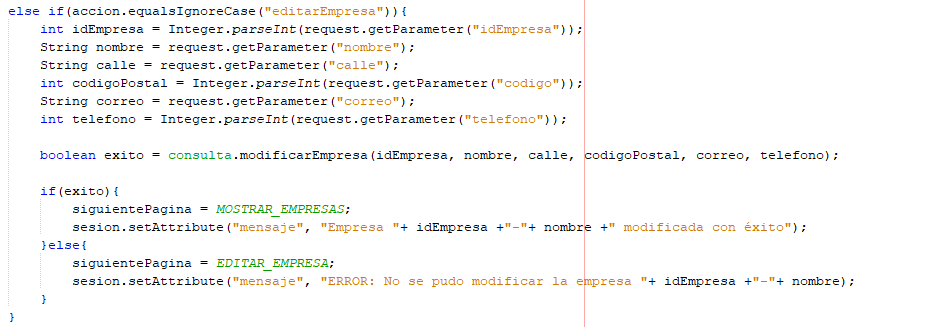


#### Editar

Para editar una empresa se ha utilizado el método modificarEmpresa que recibe como parámetros los nuevos valores de la empresa a modificar. Se realiza un UPDATE en la base de datos de los valores que se desean modificar de la empresa identificada por su id.



El usuario selecciona una empresa desde la vista mostrarEmpresas y pulsa el botón Editar. Este botón genera una nueva vista con un formulario para introducir los nuevos datos de la empresa. El controlador captura la acción junto con los nuevos valores introducidos en el formulario y llama al método modificarEmpresa pasándole dichos valores.



### Mostrar Peticiones

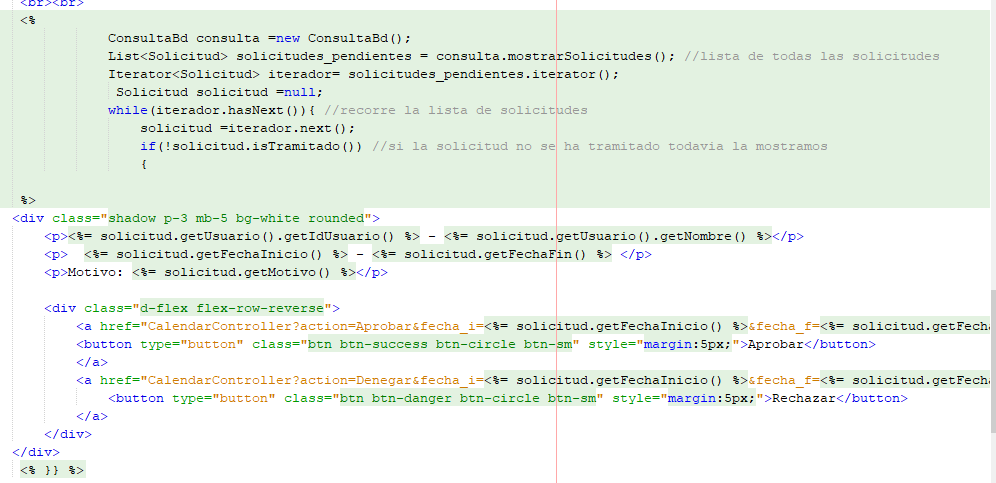
Se ha implementado un método para mostrar todas las peticiones realizadas por los empleados para solicitar días libres.



Por otro lado, tenemos una vista con un botón Peticiones que incluye una referencia a otra vista solicitudesPendientes.jsp.



Posteriormente, se abre una nueva vista con la lista de solicitudes. En esta vista representada mediante un jsp llamamos al método mostrarSolicitudes



En esta misma vista, el usuario tiene opciones de aprobar o denegar solicitudes.

#### Aprobar

#### Denegar

### Solicitar Informe

#### Empresa

#### Proyecto

#### Empleado

## Implementación Empleado

PARTE RRHH

* mantiene info de empresas,proyectos, trabajadores y calendarios de cada uno
* aprobar días libres y vacaciones(peticiones de trabajadores)
* solicitar informe sobre una empresa, proyecto o empleado sobre jornadas de trabajo(semanal, mensual, anual o periodo específico).

PARTE EMPLEADO

* introducir horas trabajadas en un proyecto/s el día anterior(jornada de 8 horas. Ej: 2 horas proyectoA, 5 horas proyectoB, 1 hora proyectoC).
* solicitar días libres, vacaciones, etc
* recibe mensajes de aprobación de vacaciones en Portal del Empleado
* solo trabajan en una empresa pero pueden estar asignados a uno o más proyectos

# BIBLIOGRAFÍA