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](#_Toc62592120)

[PROBLEMAS ENCONTRADOS Y SOLUCIONES APORTADAS 3](#_Toc62592121)

[IMPLEMENTACIÓN 4](#_Toc62592122)

[Implementación Recursos Humanos 5](#_Toc62592123)

[Mostrar Empleados 5](#_Toc62592124)

[Mostrar Proyectos 9](#_Toc62592125)

[Mostrar Empresas 12](#_Toc62592126)

[Mostrar Peticiones 16](#_Toc62592127)

[Solicitar Informe 18](#_Toc62592128)

[Implementación Empleado 19](#_Toc62592129)

[Fichar 19](#_Toc62592130)

[Calendario 20](#_Toc62592131)

[Solicitar Día Libre 21](#_Toc62592132)

[BIBLIOGRAFÍA 22](#_Toc62592133)

# 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

La mayoría del tiempo lo dedicamos a investigar bastante sobre JSP y servlets ya que no entendíamos bien su funcionamiento. Nos costó unir la parte de Java con la de HTML

En la parte de base de datos algún problema que hemos tenido fue la de hacer la conexión.

Otro de los problemas que hemos tenido con la base de datos ha sido a la hora de hacer el calendario con la clave primaria, no podíamos generar solicitudes porque la clave se repetía, entonces lo que pensamos al final fue poner una doble.

En la parte de fichar empleado el problema que tuvimos es que no funcionaba cuando le metíamos un dato tipo date. Tuvimos que separar la fecha y la hora con el método substring y así poder coger los datos de fecha y hora por separado.

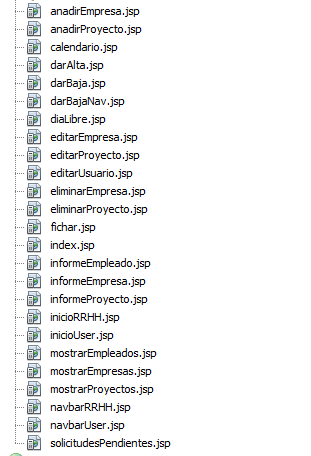
También tuvimos problemas a la hora de los botones de eliminar y editar ya que los dos necesitaban el ID pero en el form solo se podía poner un action

Otro problema encontrado fue que no sabíamos cómo avisar al usuario de que se la solicitud que ha hecho se ha aprobado o no, lo que pensamos fue crear un sistema de alertas que cuando se cargue la página de inicio de usuario coge todas las solicitudes tramitadas y no leídas y genera una alerta.

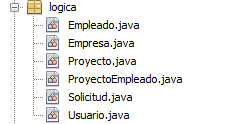
La base de datos se ha realizado en MySQL. Para la realización del diagrama hemos usado Enterprise Architect, porque creemos que es una solución muy eficiente y fácil de utilizar ya que se ha usado en cursos anteriores.

# IMPLEMENTACIÓN

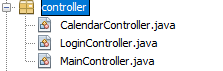
Para la elaboración de este software se ha utilizado el patrón Modelo-Vista-Controlador. En la clase Modelo hemos incluido las funcionalidades del sistema y clases con sus respectivos métodos para la conexión con la base de datos. La vista, por su parte, incluye las páginas HTML con sus formularios con 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*

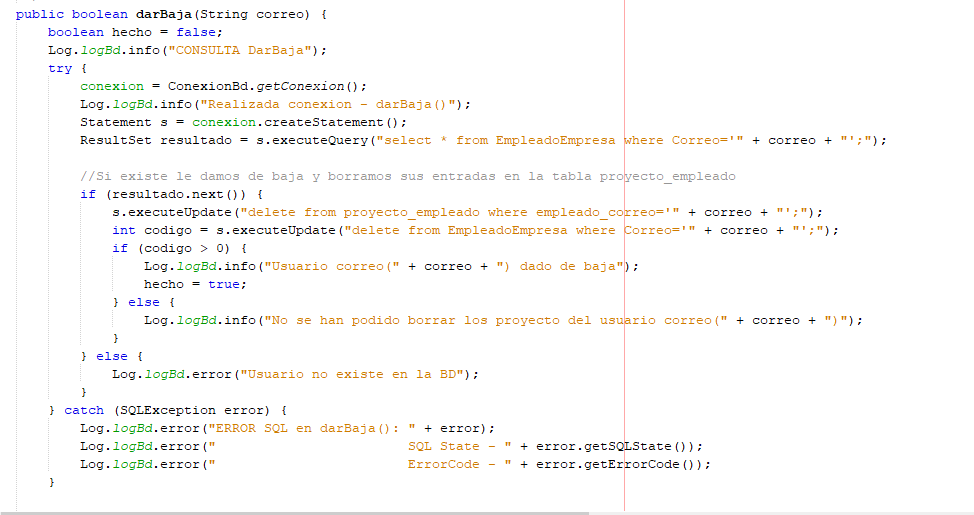
Al presionar el botón añadir desde la vista anterior, el usuario obtiene un formulario donde ha de ingresar los datos del usuario que desea dar de alta. Al presionar el botón Enviar, el controlador recibe la acción y llama al método darAlta pasándole un objeto de tipo usuario con los atributos del formulario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



*Figura 9. Controlador captura la acción altaEmpleado*

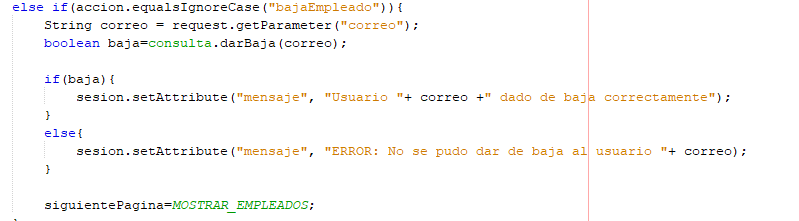
#### Eliminar

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*

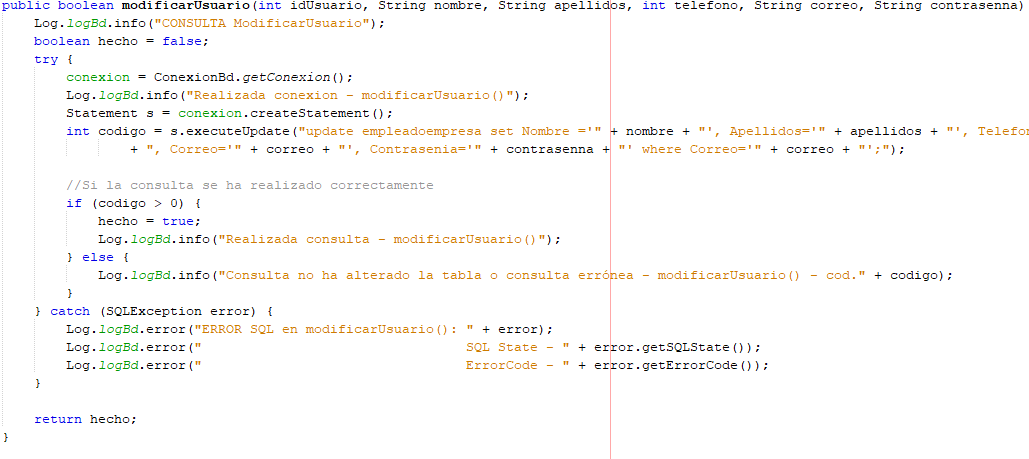
El usuario selecciona un empleado a eliminar desde la vista y, esta acción, se pasa al controlador. El controlador obtiene de la vista el correo del empleado a eliminar y realiza una llamada al método darBaja pasándole como parámetro el correo obtenido.



*Figura 11. Controlador captura acción bajaEmpleado*

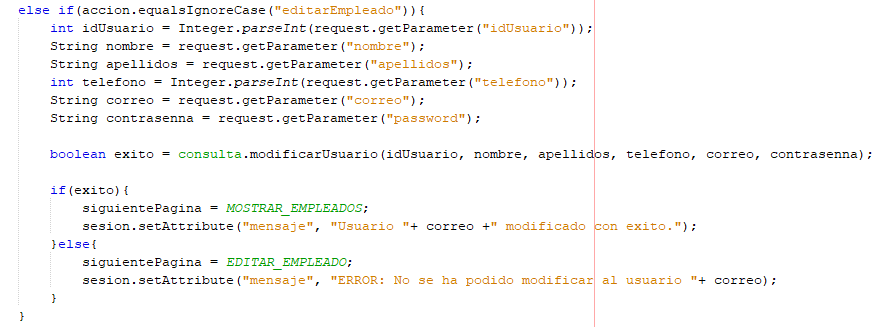
#### Editar

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.



*Figura 13. Controlador captura acción editarEmpleado*

### Mostrar Proyectos

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



*Figura 14. Método mostrar proyectos*

Por otro lado, tenemos una vista con un botón Mostrar Proyectos 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.



*Figura 15. Iterar proyecto en la vista*

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

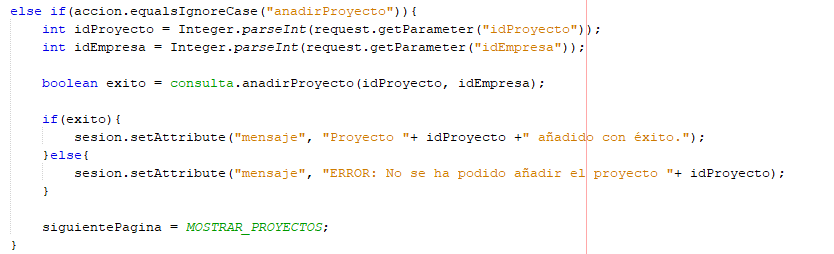
#### Añadir

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.



*Figura 16. Método añadir proyecto*

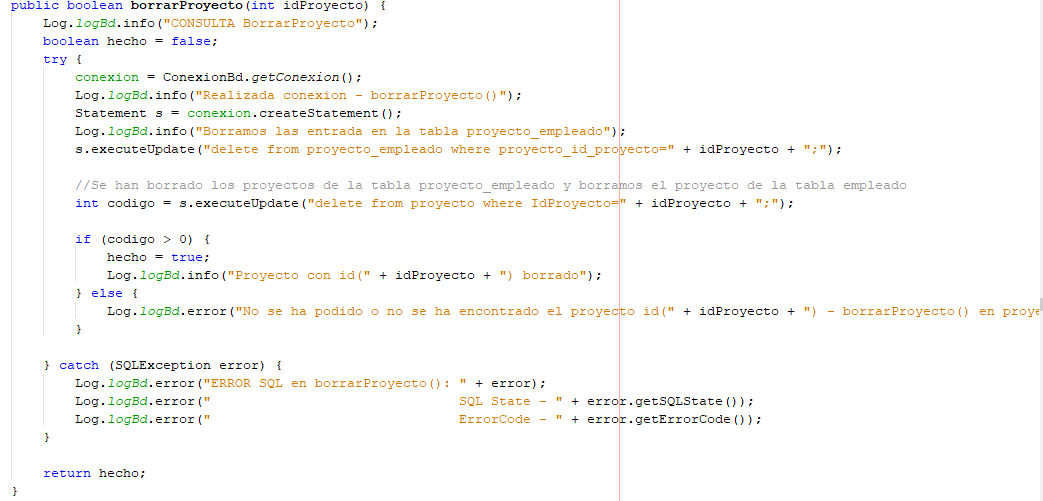
Al presionar el botón añadir desde la vista anterior, el usuario obtiene un formulario donde ha de ingresar los datos del proyecto que desea añadir. Al presionar el botón Enviar, el controlador recibe la acción y llama al método anadirProyecto pasándole los atributos del formulario que nuestro usuario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



*Figura 17. Controlador captura acción de añadir proyecto*

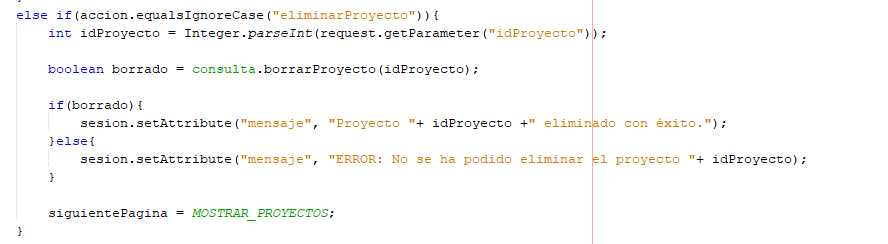
#### Eliminar

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.



*Figura 18. Método borrar proyecto*

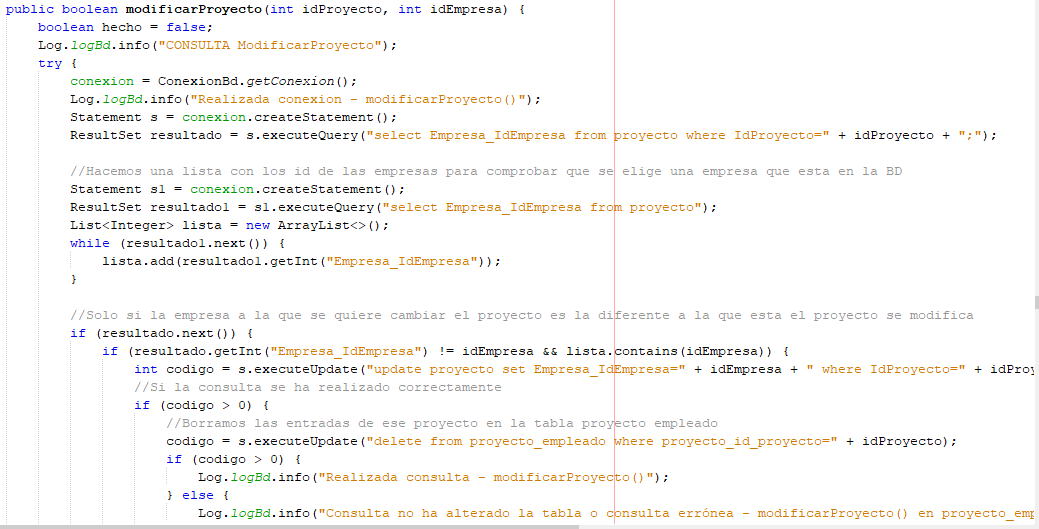
El usuario selecciona un proyecto a eliminar desde la vista y, esta acción, se pasa al controlador. El controlador obtiene de la vista el id del proyecto a eliminar y realiza una llamada al método borrarProyecto pasándole como parámetro los datos obtenidos.



*Figura 19. Controlador captura acción eliminar proyecto*

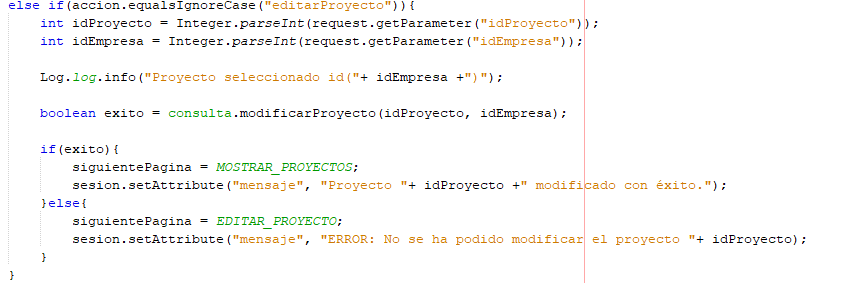
#### Editar

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.



*Figura 20. Método modificar proyecto*

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.



*Figura 21. Controlador captura acción editar proyecto*

### Mostrar Empresas

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



*Figura 22. Método mostrar empresas*

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.

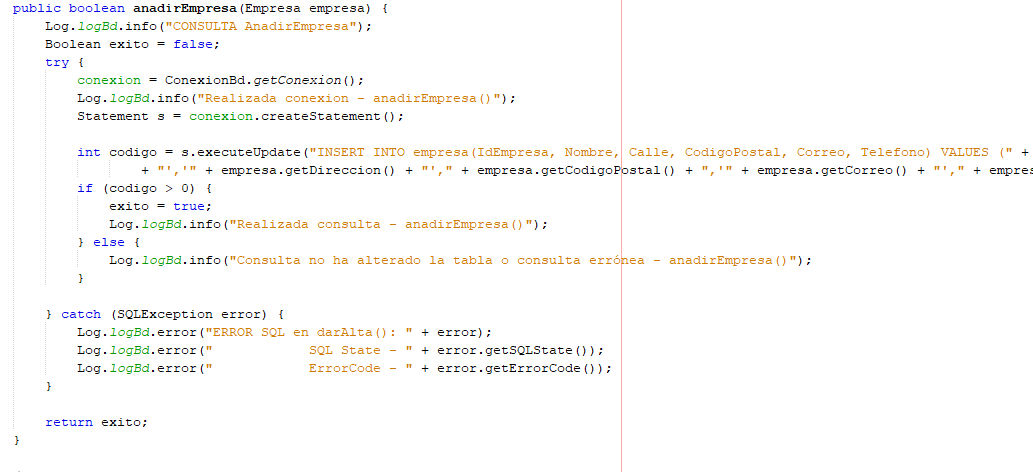


*Figura 23. Mostrar empresas desde la vista*

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

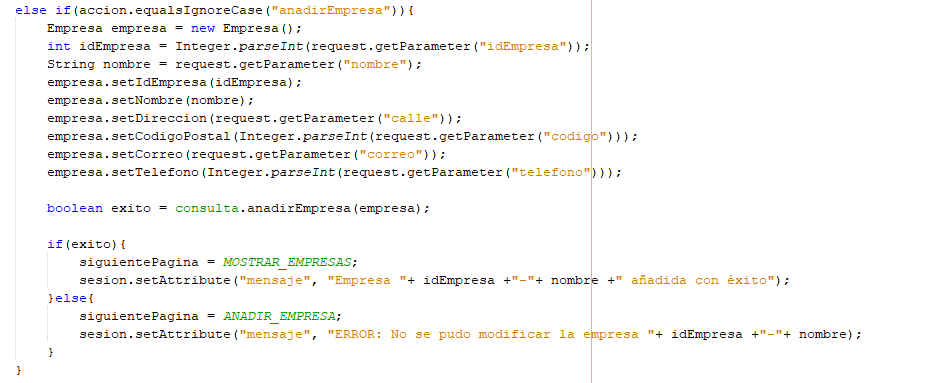
#### Añadir

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.



*Figura 24. Método añadir empresa*

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.



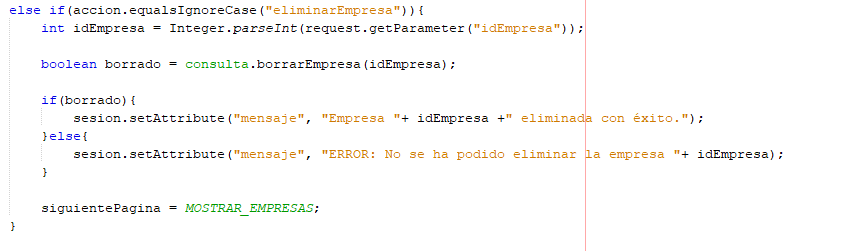
*Figura 25. Controlador captura la acción añadir empresa*

#### 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. 

*Figura 26. Método borrar empresa*

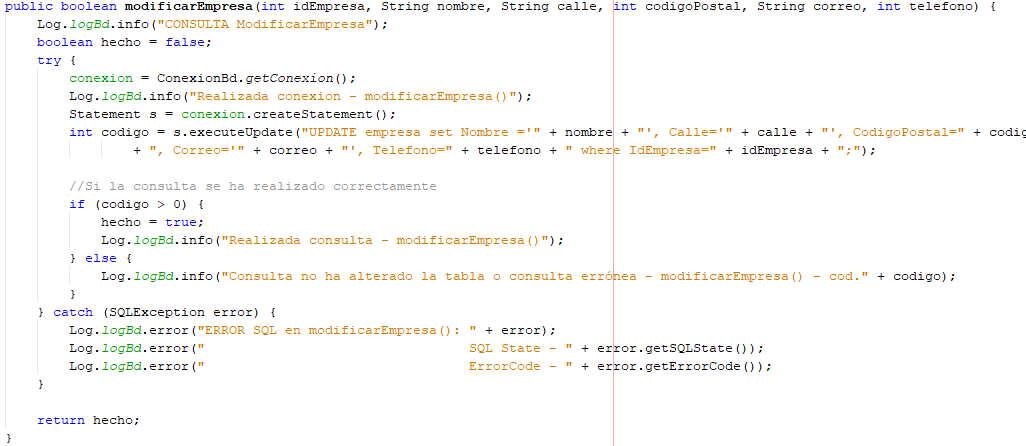
El usuario selecciona una empresa a eliminar desde la vista y, esta acción, se pasa al controlador. El controlador obtiene de la vista el identificador de la empresa a eliminar y realiza una llamada al método borrarEmpresa pasándole como párametro el id obtenido.



*Figura 26. Controlador captura la acción eliminar empresa*

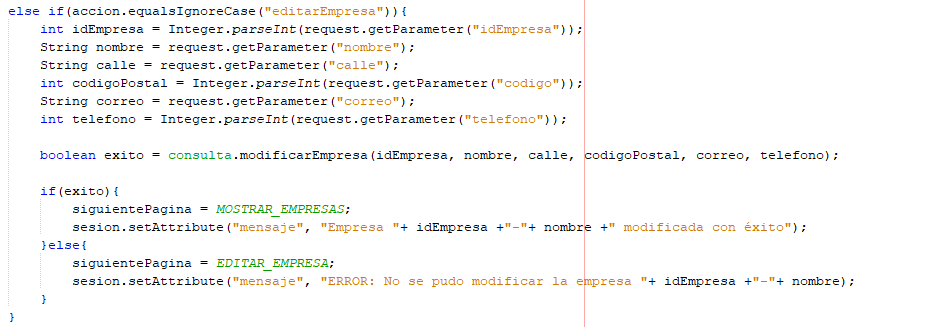
#### Editar

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.



*Figura 27. Método modificar empresa*

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.



*Figura 28. Controlador captura la acción editar empresa*

### Mostrar Peticiones

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



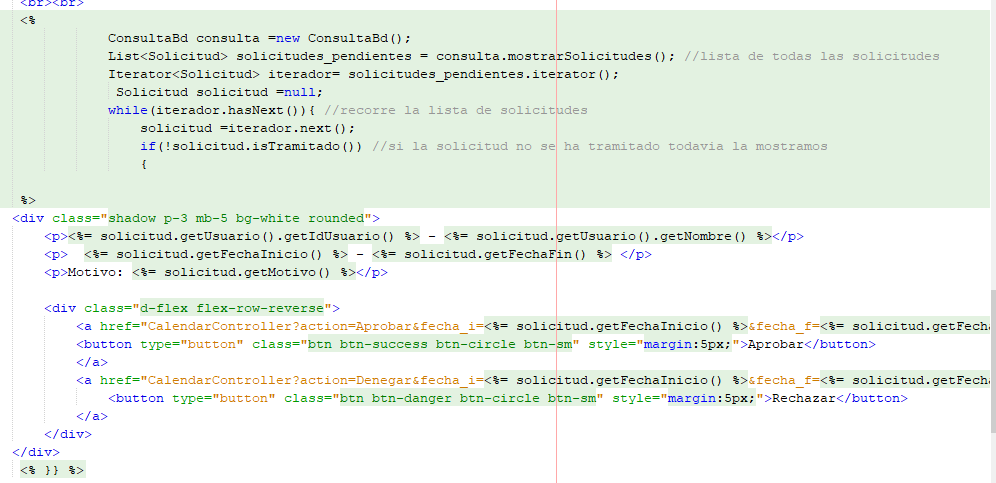
*Figura 29. Método mostrar solicitudes*

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



*Figura 30. Referencia al jsp solicitudes pendientes*

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

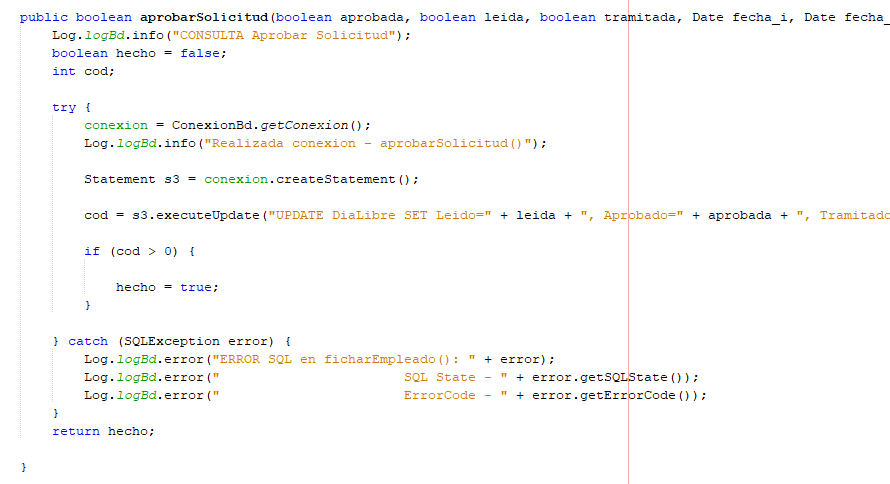


*Figura 31. Mostrar la solicitud desde la vista*

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

#### Aprobar/Denegar

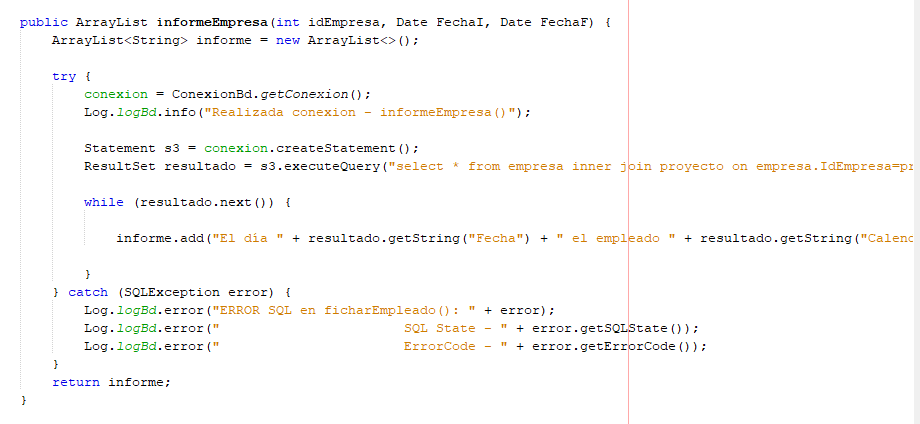
Estas peticiones comentadas anteriormente pueden ser aprobadas o rechazadas por el personal de recursos humanos. Para ello, se ha implementado un método aprobarSolicitud con unos parámetros para indicar si la solicitud se desea aprobar o rechazar. Al llamar al método aprobarSolicitud desde el controlador, la solicitud especificada cambia su estado a aprobado o denegado.



*Figura 32. Método aprobar solicitud*

### Solicitar Informe

Por último, los empleados de recursos humanos tienen la opción de solicitar el informe para un período de tiempo seleccionado. Este informe puede ser por empleado, empresa o proyecto. Para ello, se ha creado un método para cada entidad: InformeEmpresa indicando el id de la empresa, InformeEmpleado indicando el id del empleado y InformeProyecto indicando el id del proyecto.



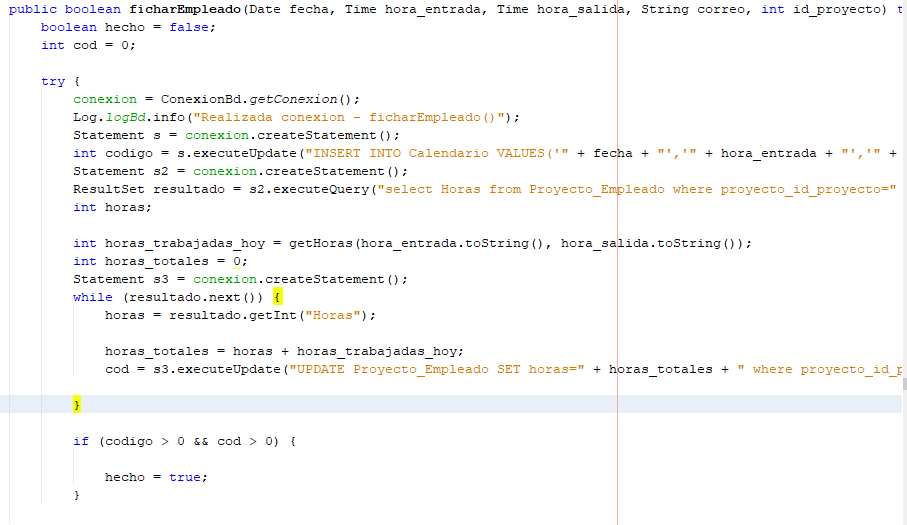
*Figura 33. Método informe empresa*

## Implementación Empleado

Por otro lado, tenemos la implementación de las funciones que pueden realizar los empleados dentro de la aplicación. Entre ellas, podemos destacar principalmente fichar un empleado, ver el calendario del empleado o solicitar un día libre.

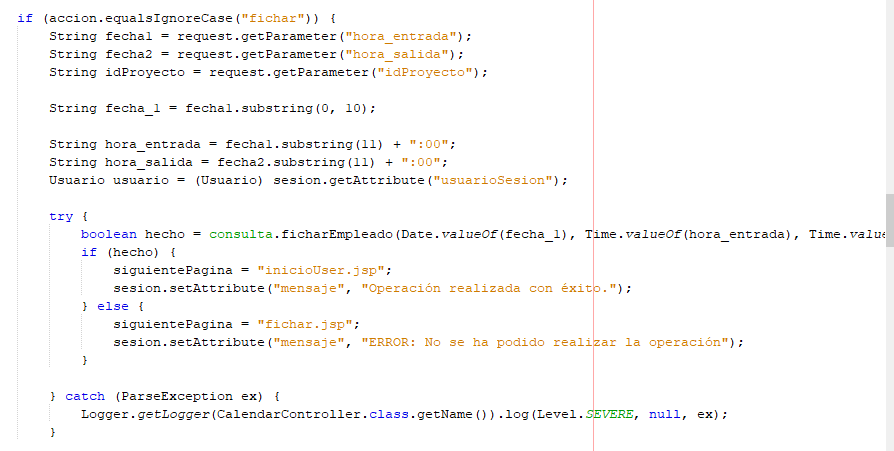
### Fichar

Para fichar un empleado se ha implementado un método ficharEmpleado pasándole como parámetros la fecha en la que se produce la acción, la hora de entrada y la hora de salida, así como el correo del empleado y el proyecto en el que ha trabajado.



*Figura 34. Método fichar empleado*

Tras introducir los datos en el formulario correspondiente, el empleado pulsa el botón enviar y esta acción se pasa al controlador CalendarioController. Desde ahí se obtienen los valores de los parámetros introducidos por el usuario y se llama al método descrito anteriormente para registrar esos valores en la base de datos.



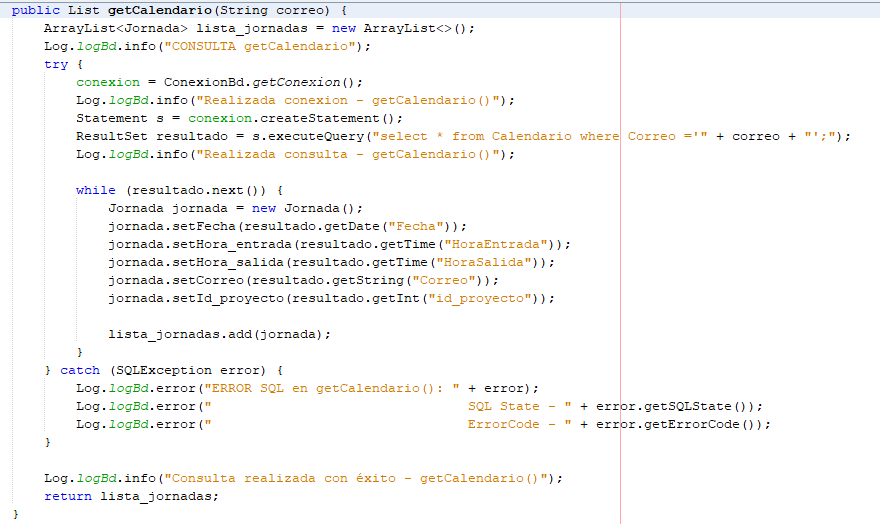
*Figura 35. Controlador captura la acción fichar empleado*

### Calendario

El empleado tiene la opción de poder visualizar el calendario con sus días libres asignados. Esta acción es posible gracias al método getDiasLibres(correo) que devuelve todos los registros de los días libres aprobados que hay en la base de datos del empleado identificado por el correo.

*Figura 36. Método obtener días libres*

Además, también se pueden visualizar los días laborales del empleado a través de un método getCalendario(correo) que contiene el registro de los días que el empleado ha estado trabajando.

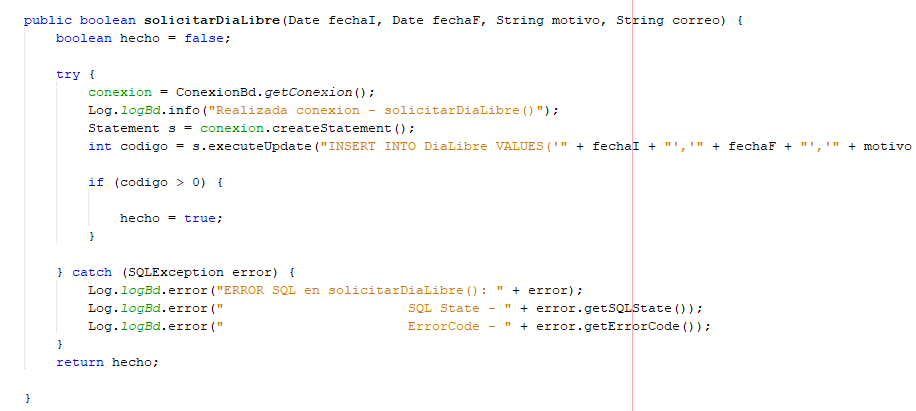
*Figura 37. Método obtener calendario*

Este método se llama posteriormente en la vista para mostrar los registros comentados anteriormente.

### Solicitar Día Libre

Por último, el empleado podrá solicitar un día libre o varios indicando la fecha de inicio y fin, así como el motivo. Si solo solicita un día libre la fecha final es la misma que la fecha inicial.

Para realizar esta acción se ha implementado el método solicitarDíaLibre que se llamará posteriormente desde el controlador CalendarioController.



*Figura 38. Método solicitar día libre*

# BIBLIOGRAFÍA

[1] <https://www.cssscript.com/create-simple-event-calendar-javascript-caleandar-js/>

[2] <https://www.w3schools.com/js/js_errors.asp>

[3] <https://danlaho.wordpress.com/2010/05/04/login-de-usuario-jspservlet/>

[4] <https://www.arquitecturajava.com/usando-java-session-en-aplicaciones-web/>

[5] <https://tomcat.apache.org/tomcat-5.5-doc/servletapi/javax/servlet/http/HttpSession.html>

[6] <https://tutobasico.com/instalar-tomcat-windows/>

[7] <https://support.academicsoftware.eu/hc/es/articles/360007014958-C%C3%B3mo-instalar-MySQL-Workbench#:~:text=Paso%201%3A%20Ve%20a%20la,Next%20para%20iniciar%20la%20instalaci%C3%B3n>.

[8] <https://docs.oracle.com/javase/7/docs/api/java/util/logging/Logger.html>

[9] <https://codigosparadesarrolladores.blogspot.com/2014/04/codigo-java-conectar-nuestra-aplicacion-con-una-base-de-datos.html>

[10] <https://developer.mozilla.org/es/docs/Web/API/Element/getAttribute>