

MEMORIA DE LA PRÁCTICA FINAL

APLICACIÓN WEB "INGENIEROS AL PESO S.A."

Arquitectura y Diseño de Sistemas Web y C/S

Grupo 11

Daniel Hernández Arcos, Jaime Jimeno Martín, Guillermo Vergel Gómez,
Carlos Martínez Álvarez, Sergio Moreno Solera y Raquel Morales Ambrós.

Índice

Introducción.....	3
Análisis del problema.....	4
Patrón MVC.....	5
Organización de la Aplicación e implementación	8
Base de Datos	8
Páginas Web	9
Java	13
Diagrama de Despliegue	14
Requisitos de la aplicación	15
Problemas encontrados y soluciones aportadas	16
Bibliografía.....	17

Introducción

El **objetivo** de la práctica es crear una aplicación utilizando el patrón de diseño Modelo Vista Controlador e incorporar todas las herramientas y tecnologías utilizadas a lo largo de la asignatura.

Consiste en desarrollar una aplicación web para la empresa "Ingenieros al peso S.A." la cual necesita desarrollar un sistema informático para la gestión de la imputación de costes del personal en las empresas que desarrollan su trabajo.

Análisis del problema

Como hemos mencionado en la introducción debemos desarrollar una aplicación web para la empresa "Ingenieros al peso S.A."

Después de haber leído el enunciado con la explicación del funcionamiento de la aplicación y determinados requisitos hemos planteado el problema de la siguiente manera:

Los trabajadores de la empresa se pueden clasificar en dos grupos: el personal de Recursos Humanos (RRHH) y el personal en contacto con los clientes (empleados).

El inicio de sesión de todos los usuarios se va a realizar mediante un usuario y contraseña y dependiendo del tipo de trabajador que solicite acceder al sistema tendrá unas funciones disponibles u otras.

Un trabajador de recursos humanos (RRHH) debe:

- Mantener la información de empresas, proyectos, trabajadores y sus respectivos calendarios.
- Atender las peticiones de los trabajadores realizadas desde la aplicación.
- Solicitar un informe de una empresa, proyecto o empleado específico.

Un empleado de la empresa debe:

- Realizar un marcaje diario a su entrada y salida de la empresa cliente.
- Indicar las horas realizadas en base a los proyectos asignados.
- Poder solicitar días libres, vacaciones u horas libres que se reflejarán en el calendario.

En cuanto a la implementación de la aplicación es necesario que la práctica sea programada en JAVA implementando el patrón MVC. Hay que usar HTML5, CSS3 y JavaScript y almacenar los datos en MySQL.

Patrón MVC

Como hemos mencionado en la introducción es necesario que la aplicación web se cree utilizando el patrón de Modelo Vista Controlador.

El patrón Modelo Vista Controlador se trata de un patrón de arquitectura de las aplicaciones software que separa la lógica de negocio de la interfaz de usuario por dos motivos principales:

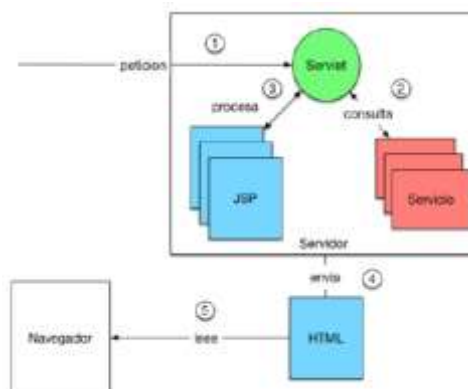
- Para facilitar por separado la evolución de ambos aspectos
- Para incrementar la reutilización y la flexibilidad de la aplicación

Dicho modelo se compone de varias vistas y varios controladores. Cada controlador trata los distintos eventos que se pueden producir en la interfaz gráfica de la aplicación (las vistas, html).

Los elementos que componen el patrón son:

- Modelo: conjunto de clases que representan la información del mundo real que el sistema debe reflejar.
En la aplicación, nuestros modelos se encuentran en la carpeta "model" dentro de "Source Packages".
- Vistas: encargadas de representar los datos al usuario. En nuestra aplicación corresponden a todas las páginas que se encuentran en los .jsp en la carpeta "Web Pages".
- Controlador: encargado de interpretar y dar sentido a las instrucciones que realiza el usuario, realizando actuaciones sobre el modelo.
Los controladores, dentro de nuestra aplicación, los podemos encontrar en la carpeta "controller" dentro de "Source Packages".

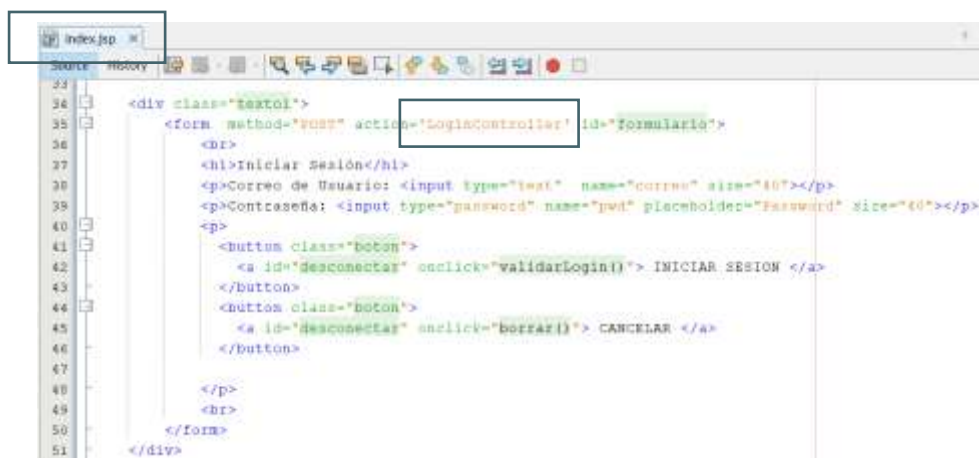
Un buen esquema del patrón en el Servidor sería el siguiente.



En nuestra aplicación web, los .jsp se conectan mediante los servlets a la lógica de la aplicación en los controladores que a su vez llaman a los dao para realizar operaciones en la base de datos. Para ello hemos instanciado los controllers y los dao necesarios para cubrir toda la funcionalidad de la aplicación.

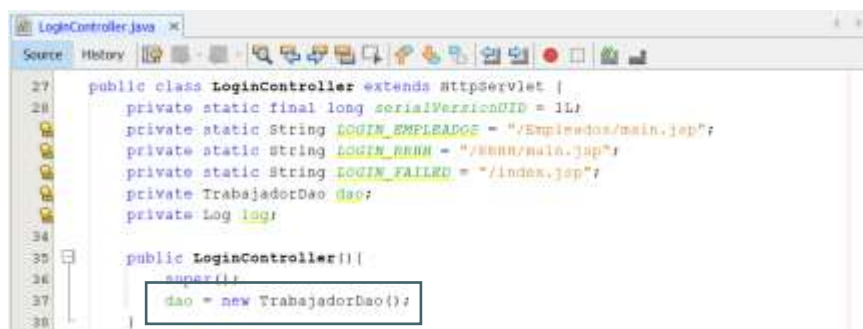
A continuación, vamos a mostrar un ejemplo de uso de todos los elementos juntos en el inicio de la sesión.

En primer lugar, tenemos el "index.jsp" desde el cual los usuarios inician sesión para poder acceder a la aplicación activando el "LoginController".



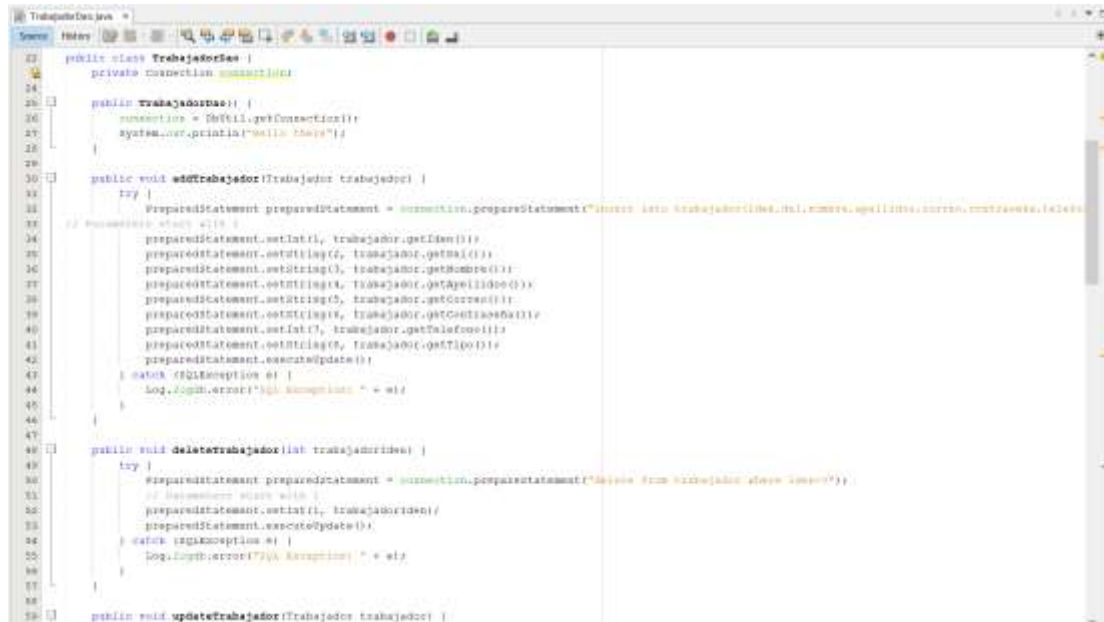
```
23
24
25 <div class="textol">
26
27 <form method="post" action="LoginController" id="Formulario">
28
29 <hr>
30 <h1>Iniciar Sesión</h1>
31
32 <p>Correo de Usuario: <input type="text" name="correo" size="40"></p>
33
34 <p>Contraseña: <input type="password" name="pwd" placeholder="Password" size="40"></p>
35
36 <p>
37 <button class="boton">
38 <a id="desconectar" onclick="validarLogin()"> INICIAR SESION </a>
39 </button>
40
41 <button class="boton">
42 <a id="desconectar" onclick="borrar()"> CANCELAR </a>
43 </button>
44
45 </p>
46
47 </p>
48
49 <hr>
50
51 </form>
52
53 </div>
```

En el "LoginController" se llama a los Dao para realizar operaciones en la base de datos, en este caso comprobar que el correo y la contraseña de un usuario están registrados.



```
27 public class LoginController extends HttpServlet {
28
29     private static final long serialVersionUID = 1L;
30     private static String LOGIN_EMPLEADO = "/Empleado/main.jsp";
31     private static String LOGIN_RHNN = "/RHNN/main.jsp";
32     private static String LOGIN_FAILED = "/index.jsp";
33     private TrabajadorDao dao;
34     private Log log;
35
36     public LoginController() {
37         super();
38         dao = new TrabajadorDao();
39     }
40 }
```

Todos los Dao se encuentran en la carpeta "util" dentro de "Source Packages". En este caso nos centramos en TrabajadorDao.java que es el que nos va a proporcionar la información necesaria de los trabajadores. Podemos encontrar los métodos: addTrabajador, deleteTrabajador, updateTrabajador, getAllTrabajadores y getTrabajadorByIden.



```
23 public class TrabajadorDao {
24     private Connection connection;
25
26     public TrabajadorDao() {
27         connection = Default.getConnection();
28         System.out.println("Hello Dao");
29     }
30
31     public void addTrabajador(Trabajador trabajador) {
32         try {
33             PreparedStatement preparedStatement = connection.prepareStatement("insert into trabajador (id, nombre, apellido, correo, contraseña, idden)
34             // PreparedStatement pstmt = null;
35             preparedStatement.setInt(1, trabajador.getId());
36             preparedStatement.setString(2, trabajador.getNombre());
37             preparedStatement.setString(3, trabajador.getApellido());
38             preparedStatement.setString(4, trabajador.getCorreo());
39             preparedStatement.setString(5, trabajador.getIdden());
40             preparedStatement.setInt(6, trabajador.getIdden());
41             preparedStatement.setString(7, trabajador.getIdden());
42             preparedStatement.executeUpdate();
43         } catch (SQLException e) {
44             Log.getLogger().log(Level.SEVERE, "Error al agregar trabajador");
45         }
46     }
47
48     public void deleteTrabajador(int trabajadorIden) {
49         try {
50             PreparedStatement preparedStatement = connection.prepareStatement("delete from trabajador where idden=?");
51             // PreparedStatement pstmt = null;
52             preparedStatement.setInt(1, trabajadorIden);
53             preparedStatement.executeUpdate();
54         } catch (SQLException e) {
55             Log.getLogger().log(Level.SEVERE, "Error al eliminar trabajador");
56         }
57     }
58
59     public void updateTrabajador(Trabajador trabajador) {
60         try {
61             PreparedStatement preparedStatement = connection.prepareStatement("update trabajador set nombre=?, apellido=?, correo=?, idden=? where id=?");
62             // PreparedStatement pstmt = null;
63             preparedStatement.setString(1, trabajador.getNombre());
64             preparedStatement.setString(2, trabajador.getApellido());
65             preparedStatement.setString(3, trabajador.getCorreo());
66             preparedStatement.setInt(4, trabajador.getIdden());
67             preparedStatement.setInt(5, trabajador.getId());
68             preparedStatement.executeUpdate();
69         } catch (SQLException e) {
70             Log.getLogger().log(Level.SEVERE, "Error al actualizar trabajador");
71         }
72     }
73 }
```

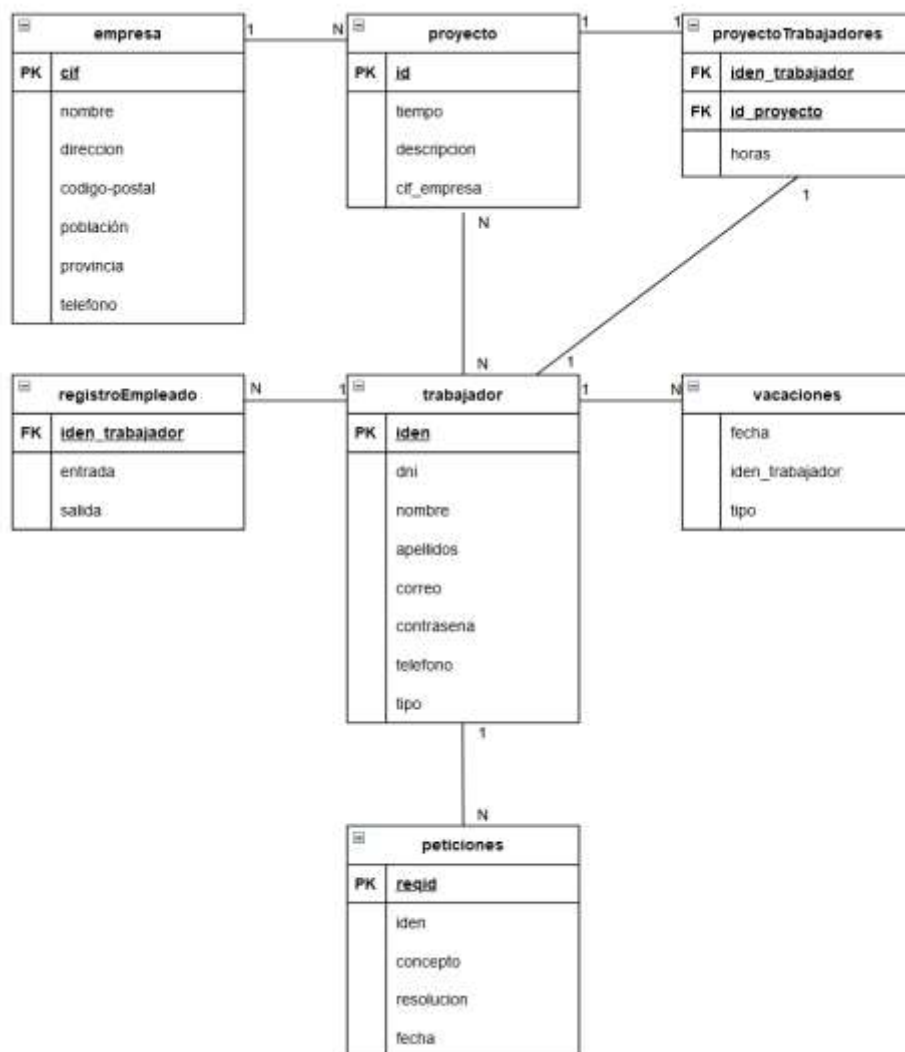
Organización de la Aplicación e implementación

Una vez planteado el problema lo hemos subdividido en las siguientes partes trabajando 2 miembros del grupo en cada una de ellas:

- I. Base de Datos en MySQL
- II. Páginas web en HTML5, CSS3 Y JavaScript
- III. Java

Base de Datos

Diagrama de Entidad Relación de la Base de Datos que compone nuestra aplicación:



Páginas Web

En cuanto a las páginas web, hemos dividido la aplicación web en dos partes diferenciadas ya que no todos los usuarios deben tener acceso a los mismos elementos. Tras iniciar sesión en una página común, los empleados tendrán acceso a determinadas páginas web y los trabajadores de RRHH a otras.

A pesar de ello, la aplicación mantiene un archivo "style_sheet.css" común que define el estilo de las páginas manteniendo toda la aplicación con una estética y colores uniforme independientemente del tipo de usuario que acceda. La aplicación al completo se ha desarrollado haciendo uso de html5, css y JavaScript.

Además, todas las páginas tienen una barra menú debajo del título de la página a través de la cuál pueden navegar por las distintas páginas de la aplicación web.



A continuación, procedemos a explicar cada página que compone la aplicación y su correspondiente funcionalidad.

Página de inicio de sesión **común**:

<i>index.jsp</i>	<p>Se trata de la página de inicio de sesión común para los trabajadores de RRHH y los empleados.</p> <p>En esta página solicita la dirección de correo y la contraseña necesarios para poder acceder a la parte de la aplicación a la que tiene acceso cada uno de los trabajadores.</p> <p>Con la función "validarLogin()" que se ejecuta al hacer click sobre el botón "INICIAR SESIÓN" comprobamos con una función si los campos introducidos son correctos y corresponden con un usuario registrado o no. La función está implementada en JavaScript y la podemos encontrar en "funciones.js". En caso de que los datos no correspondan la página muestra un mensaje de error y no permite el acceso al usuario.</p>
------------------	---

Páginas web de la parte de **Empleado**:

<i>main.jsp</i>	<p>Es la página principal que se le muestra a un empleado cuando inicia sesión correctamente en la aplicación.</p> <p>En ella se muestra un mensaje de bienvenida y un botón para desconectarse de la sesión.</p>
<i>registro_diario.jsp</i>	<p>En esta página los empleados deben realizar el marcaje de entrada al iniciar sesión y de salida al acabar la jornada laboral mediante los botones "MARCAR ENTRADA" y "MARCAR SALIDA"</p>
<i>proyectos.jsp</i>	<p>En esta página los empleados ven todos los proyectos en los que están involucrados y añaden horas en aquellos que hayan avanzado.</p> <p>Esto, al igual que en la mayoría de las páginas se hace mediante el controller específico.</p>
<i>calendario.jsp</i>	<p>En esta página los empleados pueden ver un calendario en el que se reflejan las vacaciones, días libres solicitados, etc.</p> <p>Todas las funciones necesarias para que el calendario funcione correctamente se recoge en el fichero de JavaScript: "scripts.js"</p>
<i>solicitudes.jsp</i>	<p>En la página de solicitudes los usuarios seleccionan qué tipo de solicitud desean hacer y son redirigidos a diferentes páginas: solicitud de día libre, de horas libres o vacaciones.</p> <p>Cada botón abre una nueva página en la que se solicitan los datos necesarios para hacer la solicitud.</p>
<i>días_libres_sol.jsp</i>	<p>En esta página se solicita que los empleados introduzcan el día libre que quieren solicitar junto con un breve contexto del motivo.</p> <p>Para enviar la solicitud basta con hacer click en el botón "SOLICITAR" y se mandara toda la información necesaria al controlador específico.</p>

<i>horas_sol.jsp</i>	<p>En este caso, se selecciona la hora de salida y de entrada y una breve explicación del motivo por el que el empleado solicita dichas horas libres.</p> <p>Al igual que en el caso anterior, la solicitud se registra al hacer click sobre el botón "REGISTRAR".</p> <p>A lo hora de seleccionar la hora de entrada y salida hay marcados que el valor mínimo que puede seleccionar sean las 08:30h de la mañana y el valor máximo las 18:00h. Este horario corresponde con el horario de la oficina.</p>
<i>vacaciones_sol.jsp</i>	<p>Esta página funciona de manera similar a la anterior, pero con la diferencia de que en vez de introducir una hora de salida y de entrada hay que seleccionar la fecha del primer y del último día de las vacaciones.</p>

Páginas web de la parte del trabajador de **Recursos Humanos**:

<i>main.jsp</i>	<p>Es la página de inicio que se le muestra al trabajador de RRHH de la empresa.</p> <p>Tiene un mensaje de bienvenida y una breve explicación de las funcionalidades que la aplicación le ofrece.</p> <p>Además, tiene también un botón de "DESCONECTAR" que usará para cerrar la sesión.</p>
<i>informacion.jsp</i>	<p>En esta página el trabajador de RRHH puede seleccionar si desea ver información de los empleados, los proyectos o las empresas relacionadas con Ingenieros al Peso.</p> <p>Tiene 3 enlaces distintos con los que será redirigido a otra página dependiendo del contenido que quiere ver.</p>
<i>empresasRRHH.jsp</i>	<p>En esta página se mostrará en forma de tabla la información de todas las empresas que tienen asociadas.</p> <p>Los datos que se muestran son: CIF, Nombre, Dirección, Código Postal, Población, Provincia, Teléfono y Acción.</p>

<i>trabajadoresRRHH.jsp</i>	<p>Al igual que en la página anterior se mostrará una tabla, pero esta vez con la información de los empleados de la empresa.</p> <p>Los datos que se muestran son: Identificador, DNI, Nombre, Apellidos, Teléfono y Acción.</p>
<i>proyectosRRHH.jsp</i>	<p>En esta página se muestra una tabla con toda la información relacionada a los proyectos que tiene la empresa.</p> <p>Los datos que se muestran de cada proyecto son: ID, Descripción, CIF Empresa y Acción.</p>
<i>peticiones.jsp</i>	<p>En esta página, los trabajadores de RRHH pueden visualizar en una tabla todas las peticiones que han solicitado los empleados de la empresa mostrando el ID de la petición e información del empleado junto con la resolución de la misma.</p> <p>El trabajador puede decir "Aceptar" o "Denegar" las peticiones que se encuentren pendientes de resolución.</p>
<i>informe.jsp</i>	<p>Esta es la página en la que los trabajadores de recursos humanos podrán solicitar informes de los empleados, proyectos o empresas en un rango semanal, mensual o anual.</p> <p>Cuando seleccionan los parámetros de los cuales quieren realizar el informe pulsan el botón "Aceptar" y se envía la información necesaria al controlador específico.</p>
<i>editEmpresa.jsp</i> <i>editProyecto.jsp</i> <i>editEmpleado.jsp</i>	<p>En estas páginas, los trabajadores de RRHH pueden modificar los valores asociados a una empresa, un proyecto o un empleado.</p>

En cuanto a la accesibilidad y usabilidad de la aplicación web, la implementación se ha pensado de tal manera que los usuarios no tengan ningún problema en el momento de hacer uso de ella y sea muy intuitiva, fácil de usar, con un diseño limpio y claro y coherente.

Java

La aplicación de Java se comunica a través de la página web mediante las llamadas a los controladores. En adición tenemos la implementación de jsp.

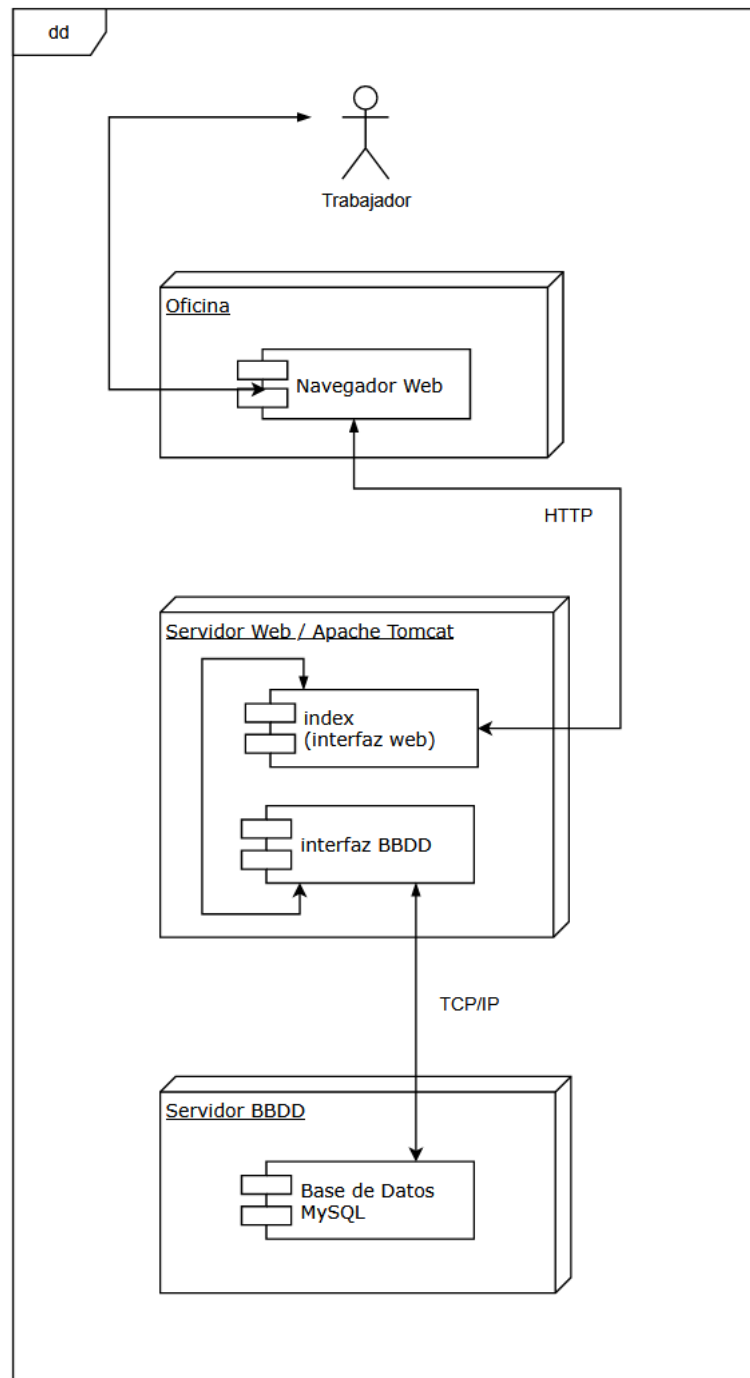
El jsp es un lenguaje de programación similar a php pero con lenguaje java, esto aumenta las posibilidades de la parte de la página web y facilita la implementación de las funciones.

El programa consta de tres paquetes: controller, donde se recoge se encuentran todos los controladores; en el modelo están las clases que se almacenarán en la base de datos y por último tenemos el paquete útil, que consta del controlador de error, del log, de la clase que da inicio a la conexión la base de datos y los de los DAO(Data access object). Estas últimas clases nos permiten acceder a los objetos y atributos serializados en la base de datos y realizar operaciones.

El funcionamiento de todos los componentes unidos funciona de la siguiente forma: hay un evento en la página web, jsp, este lanza una petición a los servlets que a su vez interactúa con los controladores. Dentro la parte java, se convierte la petición del servlet a los tipos que tenemos en la base de datos y se opera a través de ella mediante los dao.

Diagrama de Despliegue

A continuación, adjuntamos el diagrama de despliegue que corresponde con nuestra aplicación web:



Requisitos de la aplicación

A continuación, mostramos la lista de requisitos que se especifican en el enunciado de la práctica indicando si se han cumplido o no:

	Uso de HTML5, CSS3 Y JavaScript
	La práctica será programada en JAVA implementando el patrón MVC
	El código deberá estar comentado de acuerdo con los estándares de comentarios JAVA y generar los JAVADOC correspondientes
	Se hará uso de la autenticación, perfiles de usuario y sesiones
	El almacenamiento de datos se realizará en MySQL (El diseño de las tablas quedará a criterio de cada grupo)
	Debe validarse cada página con la generación de código HTML5 estándar
	Se valorará la usabilidad y accesibilidad de la aplicación

Problemas encontrados y soluciones aportadas

Durante el desarrollo de la práctica nos hemos encontrado los siguientes problemas:

- En la parte de desarrollo web al validar nos ha ido dando distintos errores que hemos ido solucionando los que no suponían un gran cambio para el resto de la práctica, que conllevaba a una edición general de todo el proyecto, también tuvimos que rehacer la parte del CSS ya que la creamos al principio como parte nativa de nuestros HTML5.
- En la parte de java tuvimos que idear distintos tipos de métodos para poder configurar nuestro proyecto de forma que todo estuviera entrelazado entre sí.

Bibliografía

- Apuntes de teoría de la asignatura.
- Ejemplo de aplicación web publicado en la página web de la asignatura.
- Para la realización del calendario usamos esta base:
<https://www.youtube.com/watch?v=rcbYjraGxDo>
- Para la validación de nuestros HTML5 usamos la siguiente página:
<https://validator.w3.org/nu/>
- Ejemplo de aplicación web publicado en la página web de la asignatura:
http://tauja.ujaen.es/bitstream/10953.1/11437/1/ALONSO_ARANDA_CARLOS_TFM_INFORMATICA.pdf
- Para el manejo de las fechas:
<https://www.javatpoint.com/java-timestamp-to-date>