**INFORME PRÁCTICA 3**



David Izquierdo Delgado

Mathieu Hernández Vélez

Grupo Mediano 2

**INTRODUCCIÓN**

Este documento tiene la finalidad de explicar detalladamente los aspectos más importantes de nuestro proyecto, de tal forma que quien lo lea pueda entender adecuadamente qué y cómo hemos realizado el trabajo.

El mismo estará estructurado de la siguiente forma:

1. Diseño general, en donde veremos los aspectos más generales de la aplicación (tanto en el código como en los datos).
2. Decisiones de diseño, en donde entramos en detalles más concretos del porqué hemos hecho determinadas partes de determinadas maneras.
3. Pila de tecnología utilizada (tanto en el lado del cliente como del servidor
4. Flujo de trabajo del programa
5. Dificultades encontradas
6. Bibliografía

**DISEÑO GENERAL**

**Gestión del código**

Teniendo en cuenta que la implementación del programa se realiza en el lenguaje Java, hemos utilizado el paradigma orientado a objetos para poder modelar tanto el gestor como el tablón de anuncios (entre otras entidades también importantes en el problema).

Además, hemos hecho uso de los siguientes patrones de diseño:

 •    **El Singleton** para el gestor de contactos, ya que dicha clase realmente solo necesita de una única instancia, la cual se encarga de realizar las operaciones CRUE del conjunto de contactos.

•  **El Factory Method**, utilizando un creador de anuncios para poder implementar los diferentes tipos de anuncios mediante una única interfaz, facilitando así la implementación y mantenibilidad del código.

• **Modelo, Vista, Controlador**, el cual nos ha permitido dividir adecuadamente el front end (vistas a través de jsp), el back end (controladores mediante servlets) y el modelo (el acceso a la información).

• **DAO y DTO**, para interactuar con la base de datos y para brindar dicha información a la aplicación, respectivamente.

En cuanto al tratamiento de esta última, hemos utilizado una base de datos alojada en los servidores de la universidad, donde creamos diferentes tablas mediante sentencias SQL, y accediendo a la misma a través de la API JDBC, el fichero de propiedades sql.properties (del que se hablará más adelante) y de los patrones de diseño DAO y DTO.

El código está dividido tanto en vistas (que serían los formularios de ingreso de datos, el Frontend) como en controladores (los que se encargan del flujo de datos, el Backend).

**Gestión de los datos**

Aparte de la carpeta en donde reflejamos qué tablas hemos utilizado y qué atributos contienen, aquí indicaremos el modelo relacional de nuestra base de datos, para que quede reflejada la relación de todas las tablas presentes en esta:

Contacto (Email, nombre, apellidos, fecha\_nacimiento, intereses)

Anuncio (Id, tipo, Titulo, Cuerpo, Fecha\_publicacion, Fecha\_fin, **Email\_propietario**, Estado\_anuncio, temas)

Intereses (Id, nombre\_interes)

Contacto\_Password (**email\_usuario**, password)

Detinado\_A (id\_anuncio, **email\_destinatario**)

**Aclaraciones:**

1. Los atributos subrayados son claves primarias y los que están en negrita foráneas (se sobreentiende a qué tabla hacen referencia nada más viendo el nombre de estas).

**2.** El atributo “intereses” de la tabla Contacto  consta de una lista de intereses en forma de cadena de caracteres (en vez de ser una sola). Decidimos hacerlo así ya que facilitaba la obtención de los intereses de cada usuario.

Cabe aclarar que  contamos con un fichero de propiedades (sql properties), en donde almacenamos información de la base de datos (como su url, usuario y password) y las consultas utilizadas por los diferentes DAO de nuestro programa, pudiendo así acceder, modificar y eliminar los diferentes datos de nuestro programa.

Dichos DAOs se encuentran documentados en el fichero de documentación del que se hablará más adelante.

**DECISIONES DE DISEÑO E IMPLEMENTACIÓN**

**Back end**

Implementamos toda la funcionalidad descrita en la práctica a excepción de los filtros y barra de búsqueda.

Para llevar a cabo esto, contamos con código java puro (clases, DTOs y DAOs) y varios servlets.

De manera general, tenemos un par servlet-vista, de forma que el primero obtiene información que requiere el segundo (a través del DTO correspondiente) enviándolo posteriormente.

**Front end**

Contamos con los siguientes jsp, cada uno correspondiente a una página de la aplicación:

Index: La primera que se muestra al usuario y que cuenta con un botón para iniciar sesión y otra para registrarse, además de los 10 anuncios más recientes.

signUp: Aquí es donde el usuario se registra a través del formulario presente en la página.

login: Justo después de iniciar sesión o crear una cuenta, esta es la página de inicio del usuario. En ella se muestran las diferentes opciones con las que cuenta el usuario: modificar su cuenta, publicar anuncio, buscar anuncios o cerrar sesión.

modify: En donde el usuario puede modificar sus datos personales.

modifyAdd: Permite al usuario modificar los anuncios que ha publicado.

errorPage: Página de error, el cual muestra al usuario qué ha ocurrido concretamente.

**FLUJO DE TRABAJO**

En el index se comprueba si el usuario está logueado no. En caso afirmativo, se le muestra mediante dos botones la opción de modificar sus datos o cerrar sesión (en el primer caso iría a la página correspondiente de modificación de estos, mientras que en el segundo caso el index se actualizaría); en caso contrario este mostrará otros dos botones con la opción de iniciar sesión o de registrarse (nuevamente redirigiendo a sus páginas correspondientes).

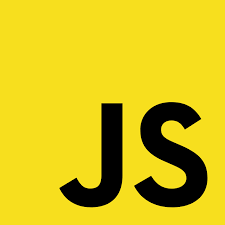
Una vez el usuario haya iniciado sesión (o creado una cuenta), la página de login contendrá los siguientes enlaces:

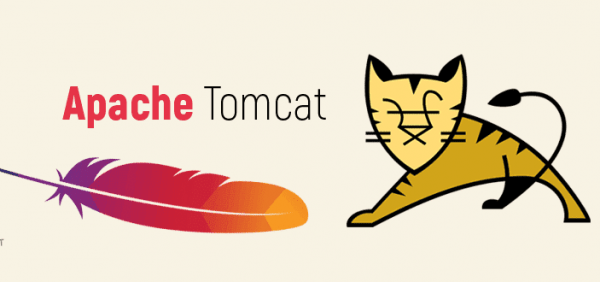
* Modificar sus datos
* Ver sus propios anuncios
* Crear un anunción
* Cerrar sesión

Además, los 10 anuncios más recientes aparecerán en dicha página.

En cuanto a los controladores, estos se encargan de recoger la información de los formularios y pasárselos mediante DAOs a la base de datos, la cual se encargará no solo de almacenar la información sino de comprobar que el inicio de sesión sea correcto, la modificación de datos, etc.

**PILA DE TRABAJO**

**Lado del cliente Lado del servidor**



**DIFICULTADES ENCONTRADAS**

* Tuvimos problemas a la hora de gestionar las carpetas, clases… ya que teníamos muchos problemas a la hora de ensamblar todo el código.
* Fue difícil diseñar la página web, ya que estamos acostumbrados a ejercicios más pequeños en cuanto a número de funcionalidades, interacciones entre código, etc.
* La conversión de JSP a Servlets, ya que había ciertas diferencias en cuanto a sintaxis, métodos… que complicaron un poco esta tarea.
* Coordinarnos entre nosotros, ya que aunque nos sirvió bastante utilizar GitHub como un medio de compartir el código, aún nos cuenta trabajar en equipo dado que no es una habilidad que se desarrolle demasiado en la carrera.

**FUENTES CONSULTADAS**

* <https://www.w3schools.com/html/> (HTML)
* <https://www.w3schools.com/css/> (CSS)
* <https://stackoverflow.com/> (para buscar solución a ciertos problemas presentados)