Curso 2014-2015

LinkUnedIn

Práctica Tecnologías web – Grupo 11

Juan Camba | David Díaz | Xavi Mira

Índice de contenido

1Introducción....4
2Composición del grupo y roles....4
3Plan de trabajo....4
4Decisiones arquitectónicas y restricciones de diseño....5
5Especificación de requisitos....7
5.1Diagrama de contexto del sistema....7
5.2Diseño de Casos de Uso....8
5.2.1Diagrama de Casos de Uso....8
5.3Detalle de los Casos de Uso....9
6Diseño de datos....16
6.1Diagrama E-R....16
6.2Descripción de tablas de la base de datos....17
7Arquitectura....19
7.1Diagrama de subsistemas....19
8Componentes....19
8.1Diagrama de componentes....19
8.2Diseño de componentes....21
9Mapa WEB....23
10Tecnologías usadas....24
10.1JPA....24
10.2Tiles....24
10.3Bootstrap....24
10.4HSQLDB HyperSqlDB....24
10.5Framework Struts....24
11Anexos....25
11.1Ficheros de configuración....25

1 Introducción

Mediante esta práctica hacemos uso de las técnicas y paradigmas que se abordan en la asignatura Tecnologías Web, a través del desarrollo de una página (portal) Web que se encargará de la puesta en contacto entre profesionales y empleadores de cualquier sector:

- Especificación detallada de un sistema
- Aplicación de los principios de buen diseño.
- Separación de capas (presentación, modelo, datos)
- Uso patrones de diseño (modelo Vista-Controlador)
- Uso y diseño de BBDD orientadas a la aplicación
- Creación de documentación tanto técnica como de uso del sistema

2 Composición del grupo y roles

Esta práctica ha sido programada por Juan Marcelo Camba, David Díaz y Javi Mira. Dados los plazos de entrega, desarrollo del proyecto e hitos marcados, los tres componentes del grupo han realizado en distintas partes del proyecto todas las funciones, a fin tanto de proporcionar al mismo nuestros distintos enfoques así como para proporcionar a todos los integrantes de la experiencia en los distintos roles.

Por no pertenecer los tres a la misma ciudad, hemos tenido que usar herramientas para poder trabajar y reunirnos en remoto (Skype / Teamviewer / Git), lo que nos ha permitido, pese a las diferentes localizaciones de los miembros, trabajar de una manera fluída.

3 Plan de trabajo

26/03/15: Constitución del grupo.
05/03/15: Registro en la página web. Toma de contacto con la práctica.
10/03/15: Reparto preliminar de tareas. Puesta en común del entorno de trabajo.
17/03/15: Pruebas preliminares con Netbeans / configuración entornos trabajo
23/03/15: Discusiones E-R / Framework /Puesta en común interfaces
24/03/15: Puesta en común E-R definitiva / Esbozo flujo web
26/03/15: Creación Git / Revisión casos de uso
29/03/15: Pruebas bootstrap
02/04/15: Persistencia y creación entidades
06/04/15: Tiles y pruebas logueo usuario
08/04/15: Correcciones pruebas usuario
11/04/15: Correcciones con las clases del patrón Facade
15/04/15: Actualizados formularios con campo ID. Añadidas funciones modify

para entidades
16/04/15: Diseño formularios usuario / pruebas bootstrap - tiles
21/04/15: Creación de consultas para búsquedas
23/04/15: Alta usuario implementada. Creado formulario registro y Logout
30/04/15: Añadidas experiencias laborales
04/05/15: Nuevo formulario de búsqueda
06/05/15: Se añade la info de educación a los perfiles
09/05/15: Arreglados errores con el borrado de ítems "Educación".Añadidos Conocimientos
10/05/15: Se modifica el listado de usuarios
16/05/15: Modificados estilos y layouts de formularios
23/05/15: Corrección de errores varios
13/06/15: Ahora se muestran correctamente fotos y pdf. Añadida vista de currículums
14/06/15: Pruebas con la máquina virtual y empaquetado

4 Decisiones arquitectónicas y restricciones de diseño

- Se opta por un modelo por capas combinado con MVC para reducir el acoplamiento y aumentar la cohesión de los distintos elementos software que componen el sistema.
- La capa de persistencia se implementa mediante una capa adicional de abstracción, usando JPA, por las siguientes razones :
  - JPA nos proporciona independencia del Sistema de Gestión de Bases de Datos.
  - Las tareas de mantenimiento correctivo y adaptivo son menos costosas dado que no se han de generar de nuevo scripts SQL .
- La capa de modelo se implementará usando clases Manager, que en la práctica lo que hará será implementar el patrón Facade para reducir la interacción entre las distintas clases. La capa de modelo implementará la lógica de negocio accediendo directamente a la capa de persistencia.
- Se usa el patrón Singleton para la creación de las fachadas (clases Manager) para evitar la creación/destrucción de objetos repetidamente.
- Usamos el framework Bootstrap para la maquetación ya que utiliza componentes y servicios creados por la comunidad web, tales como: HTML5 shim, Normalize.css, OOCSS, jQuery UI, LESS y GitHub. Es un conjunto de buenas prácticas que perduran en el tiempo. Quizás dentro de sus características más destacadas, podríamos mencionar:
  - La implementación de HTML5 + CSS3.
  - El famoso Grid system, que por defecto incluye 12 columnas fijas o fluidas, dependiendo de si tu diseño será responsivo o no.
  - El uso de LESS, que es una ampliación a las famosas hojas de estilo CSS, pero a

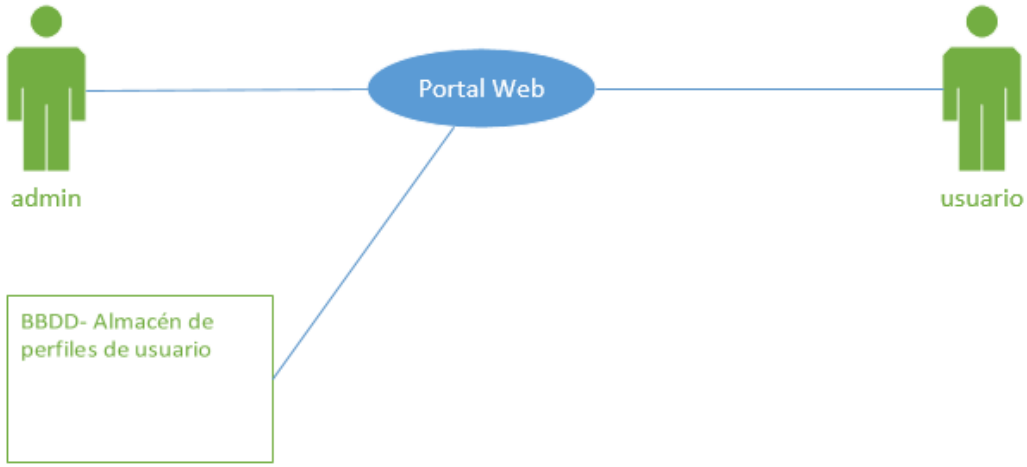
diferencia de estás, funciona como un lenguaje de programación, permitiendo el uso de variables, funciones, operaciones aritméticas, entre otras, para acelerar y enriquecer los estilos en un sitio web.

- OOCSS, css orientado a objetos, que está organizado por módulos independientes y reutilizables en todo el proyecto.
- Hay una enorme comunicad que soporta este desarrollo y cuenta con implementaciones externas como WordPress, Drupal, SASS o jQuery UI.
- Herramienta sencilla y ágil para construir sitios web e interfaces. Una vez que entiendas y domines su funcionamiento, verás lo fácil que es hacer efectos y diseñar interfaces que te ahorran realmente mucho tiempo de trabajo.
- Además y por si fuera poco tiene un theme por defecto bastante optimizado y que puedes modificar o crear tu propio theme.

- Una única foto (opcional) por currículum para facilitar búsquedas
- Un único documento relativo al currículum por perfil (experiencia). Posibilidad de ser un pdf con todos los documentos relativos a experiencia escaneados.

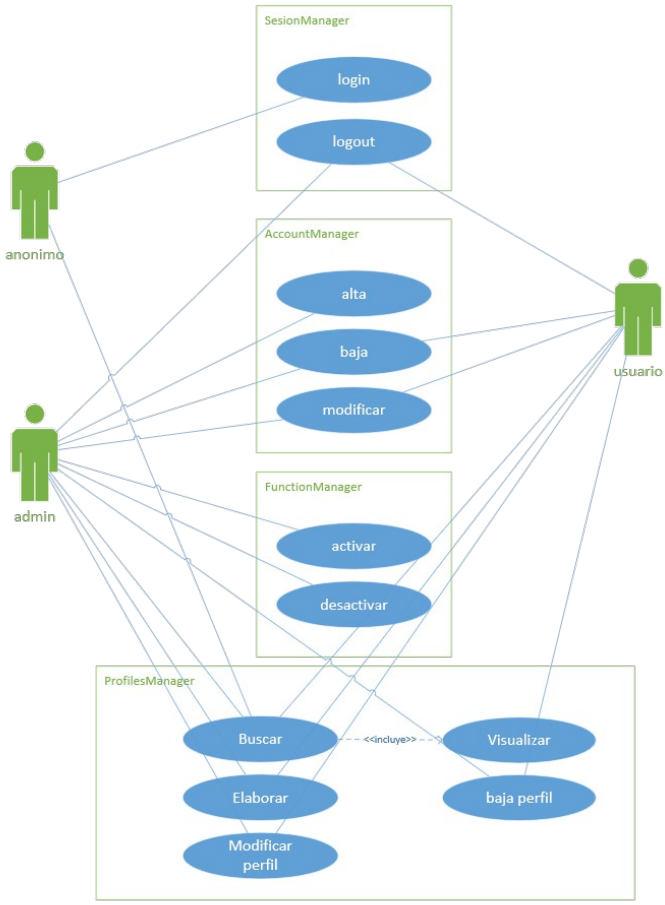
5 Especificación de requisitos

5.1 Diagrama de contexto del sistema



5.2 Diseño de Casos de Uso

5.2.1 Diagrama de Casos de Uso



5.3 Detalle de los Casos de Uso

C.U. - 001	Logarse
Descripción	Login en el sistema (usr/pass)
Actores	Usuario Anónimo
Página de inicio	
Secuencia de eventos normal	• El sistema comprueba si el usuario ya está logeado, si no lo está se le redirigirá a la página de búsqueda. • Se le presentará un formulario con usuario y contraseña • El usuario escribe el usuario y la contraseña y pulsa enviar • El sistema comprueba el usuario y la contraseña en la base de datos • Si no existe o la contraseña no coincide se lanza una excepción (podría cambiarse esto) y vuelve a la página de login • Si se logea con éxito mostrará la página de búsqueda de perfiles
Excepciones	Error al validar -> Mostramos error y se vuelve a la misma página.
Precondiciones	El usuario debe estar dado de alta previamente.
Postcondiciones	Usuario pasa a ser Usuario o Administrador
Comentarios	

C.U. - 002	Deslogarse
Descripción	El usuario o administrador pasa a ser anónimo
Actores	Usuario, Administrador
Página de inicio	
Secuencia de eventos normal	
Excepciones	
Precondiciones	El usuario / administrador debe estar logado

	previamente
Postcondiciones	
Comentarios	

C.U. - 003	Buscar
Descripción	Búsqueda de currículums
Actores	Usuario, Usuario Anónimo
Página de inicio	
Secuencia de eventos normal	• Se le muestra al usuario un formulario en el que hayan campos de selección múltiple tales como ◦ localización ▪ ciudad ▪ pais ◦ experiencia profesional ▪ menos de 1 año ▪ entre 1 y 3 años ▪ mas de 3 años ◦ titulaciones ▪ ingeniero ▪ ingeniero técnico ▪ graduado en ingeniería ▪ FPII ▪ FPI ▪ Graduado escolar ▪ Sin formación ◦ Conocimientos ▪ introducción manual de etiquetas • Al realizarse la búsqueda debe presentar los resultados proveyendo enlaces a los casos de uso extendidos (visualizar, mostrar interés) • los resultados deberían mostrarse en la misma pantalla de búsqueda para facilitar el uso de la aplicación

Excepciones	Búsqueda no genera ningún resultado Tratarlo como una excepción?
Precondiciones	
Postcondiciones	
Comentarios	

C.U. - 004	Alta
Descripción	Se crea un usuario
Actores	Usuario Anónimo, Administrador
Página de inicio	
Secuencia de eventos normal	• El sistema comprueba si ya está logeado. • Si está logeado y no es administrador lanza una excepción y muestra la pantalla de búsqueda de currículums. • Sino se le muestra un formulario de registro en el que introducir los datos. • Al enviar los datos el sistema crea el perfil en la base de datos y se redirige al usuario para que empiece a crearse un perfil.
Excepciones	El usuario no es admin y está logeado
Precondiciones	El usuario, Administrador debe estar logado previamente
Postcondiciones	Se crea un perfil en el sistema
Comentarios	

C.U. - 005	Modificar perfil
Descripción	Se modifica un usuario
Actores	Usuario, Administrador
Página de inicio	
Secuencia de eventos normal	

Excepciones	
Precondiciones	
Postcondiciones	Perfil modificado
Comentarios	Usuario modifica sólo su perfil; Administrador puede modificarlos todos

C.U. - 006	Baja perfil
Descripción	Se da de baja un perfil de usuario
Actores	Usuario, Administrador
Página de inicio	
Secuencia de eventos normal	
Excepciones	
Precondiciones	El perfil debe existir previamente
Postcondiciones	
Comentarios	Usuario borra únicamente su perfil; Administrador puede borrar todos

C.U. - 007	Activar
Descripción	El administrador activa las funciones de un usuario
Actores	Administrador
Página de inicio	
Secuencia de eventos normal	• El sistema comprueba si el usuario es administrador. • Si no lo es mostrará una página indicando la falta de permisos. • Sino se activará la función y se devolverá al administrador a la página principal de administración.
Excepciones	La función ya está activada
Precondiciones	El usuario debe estar creado previamente
Postcondiciones	
Comentarios	

C.U. - 008	Desactivar
Descripción	El administrador desactiva las funciones de un usuario
Actores	Administrador
Página de inicio	
Secuencia de eventos normal	• El sistema comprueba si el usuario es administrador. • Si no lo es mostrará una página indicando la falta de permisos. • Sino se desactivará la función y se devolverá al administrador a la página principal de administración.
Excepciones	La función ya está desactivada
Precondiciones	
Postcondiciones	
Comentarios	

C.U. - 009	Elaborar perfil (antiguo caso de uso alta)
Descripción	Se rellena el perfil de usuario
Actores	Usuario, Administrador
Página de inicio	
Secuencia de eventos normal	• Si el usuario no está logeado redirigir a la búsqueda de currículums. • Si el usuario es administrador se le mostrará un listado de usuarios para poder seleccionar aquel en el que quiere elaborar un perfil. • Sea usuario o administrador (y tras selecionar usuario en caso de que sea el administrador) se le presentará el formulario para elaborar el perfil. • El sistema obtendrá la información y lo guardará
Excepciones	
Precondiciones	El usuario está logeado
Postcondiciones	

Comentarios	

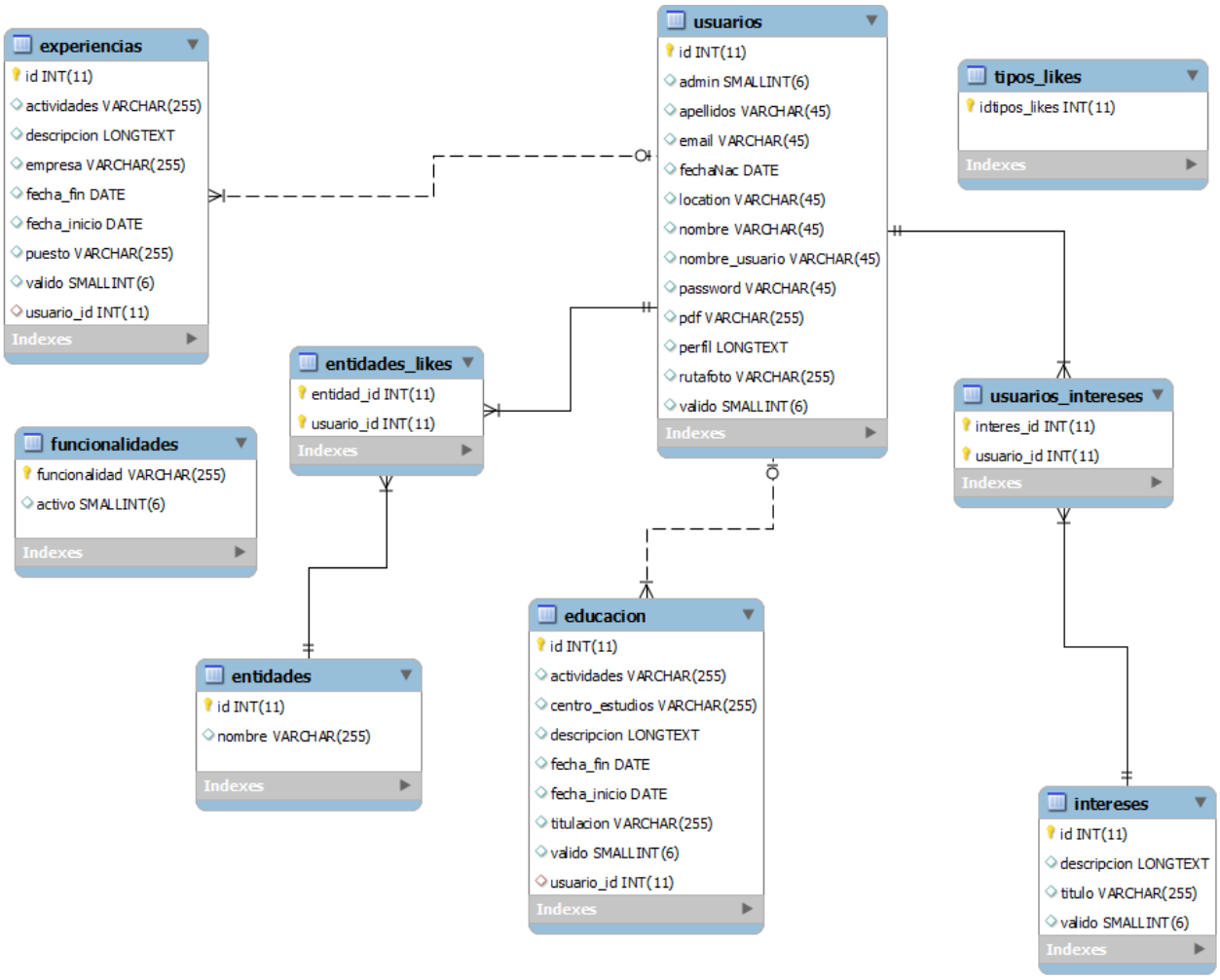
C.U. - 010	Modificar perfil (antiguo caso de uso alta)
Descripción	Se rellena el perfil de usuario
Actores	Usuario, Administrador
Página de inicio	
Secuencia de eventos normal	• La secuencia de eventos será la misma que en el caso de uso C.U.010
Excepciones	
Precondiciones	El usuario está logeado
Postcondiciones	
Comentarios	

C.U. - 011	Visualizar
Descripción	Se rellena el perfil de usuario
Actores	Usuario, Administrador, Usuario Anónimo
Página de inicio	
Secuencia de eventos normal	• El sistema comprueba que el usuario existe. Si no existe se mostrará un mensaje de error y se le devolverá a la página de visita. • El sistema mostrará los datos en un formulario a ser posible en la misma pantalla de búsqueda para facilitar el uso
Excepciones	
Precondiciones	El usuario/perfil a consultar existe
Postcondiciones	
Comentarios	

C.U. - 012	Baja (de AccountsManager)
Descripción	Elimina el usuario de la bbdd
Actores	Usuario, Administrador
Página de inicio	
Secuencia de eventos normal	• Si el usuario no está logeado redirigir a la búsqueda de currículums. • Si el usuario es administrador se le mostrará un listado de usuarios para poder seleccionar aquel al que quiere dar de baja. • Sea usuario o administrador (y tras selecionar usuario en caso de que sea el administrador) se procederá a borrar el perfil. • Por último se mostrará un mensaje indicando que la operación se ha realizado con éxito
Excepciones	
Precondiciones	El usuario está logeado
Postcondiciones	El usuario deja de existir
Comentarios	

6 Diseño de datos

6.1 Diagrama E-R



6.2 Descripción de tablas de la base de datos

TABLA USUARIOS

id	Primary key
admin	1 para los administradores, 0 para usuario normal
apellidos	Apellidos del usuario
email	Email del usuario
fechaNac	Fecha de nacimiento
location	Dirección del usuario
nombre	Nombre del usuario
nombre_usuario	Nombre con el que inicia la sesión
password	Password
pdf	Ruta al archivo pdf
perfil	
rutafoto	Ruta a la foto de perfil
valido	1 puede acceder, 0 desactivado

EDUCACION

En esta tabla se guarda los títulos , cursos, etc que ha obtenido el usuario

id	Primary key
centro_estudios	Nombre del centro donde se ha realizado la actividad
titulacion	Nombre de la titulación recibida
especialidad	
descripcion	Descripción de lo que ha estudiado
usuario_id	Usuario al que pertenece la educación
fecha_inicio	Fecha inicio de la actividad
fecha_fin	Fecha fin de la actividad
valido	1 activo, 0 desactivado

EXPERIENCIAS

En esta tabla se guardan las experiencias laborales del usuario

id	Primary key
empresa	Nombre de la empresa
puesto	Puesto ocupado en la empresa
actividades	Actividades que realizaba en la empresa
descripcion	Descripción detallada de las actividades
usuario_id	Usuario al que pertenece la educación
fecha_inicio	Fecha inicio de la actividad
fecha_fin	Fecha fin de la actividad
activo	1 activo, 0 desactivado

FUNCIONALIDAD
En esta tabla se guardan la funcionalidades administrables del sistema. Si estan desactivadas no se muestran

id	Primary key
funcionalidad	Nombre de la funcionalidad
activo	1 activo, 0 desactivado

INTERESES

id	Primary key
descripcion	Descripción del interés
titulo	Titulo del interés
valido	1 activo, 0 desactivado

USUARIOS_INTERESES
Esta tabla relaciona los intereses con los usuarios

usuario_id	Id del usuario
interes_id	id interés

ENTIDADES

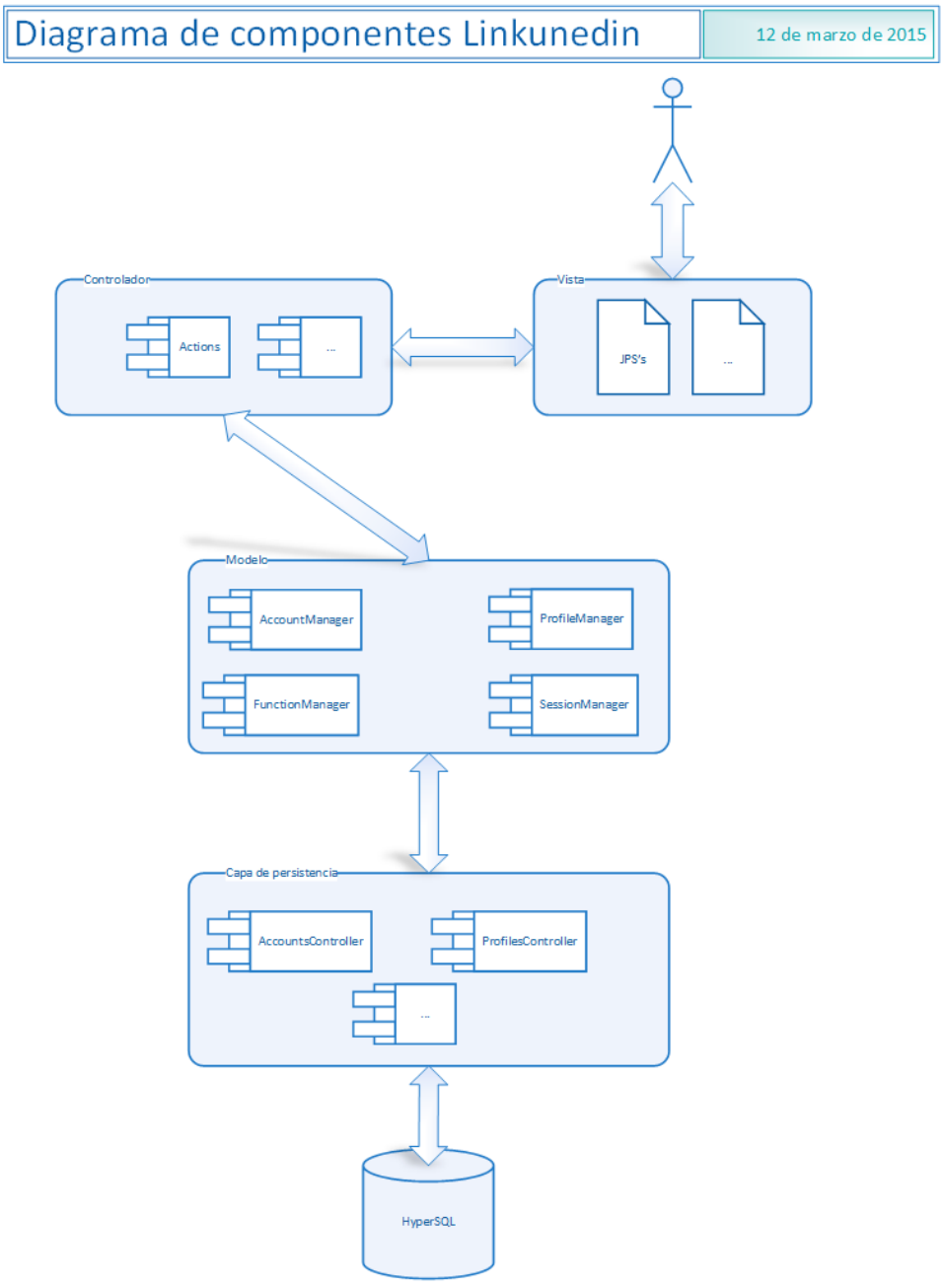
id	Primary key
nombre	Nombre de la entidad

ENTIDADES_LIKES
En esta tabla se guardan los likes que le da un usuario a una entidad

usuario_id	Id del usuario
entidad_id	Id de la entidad

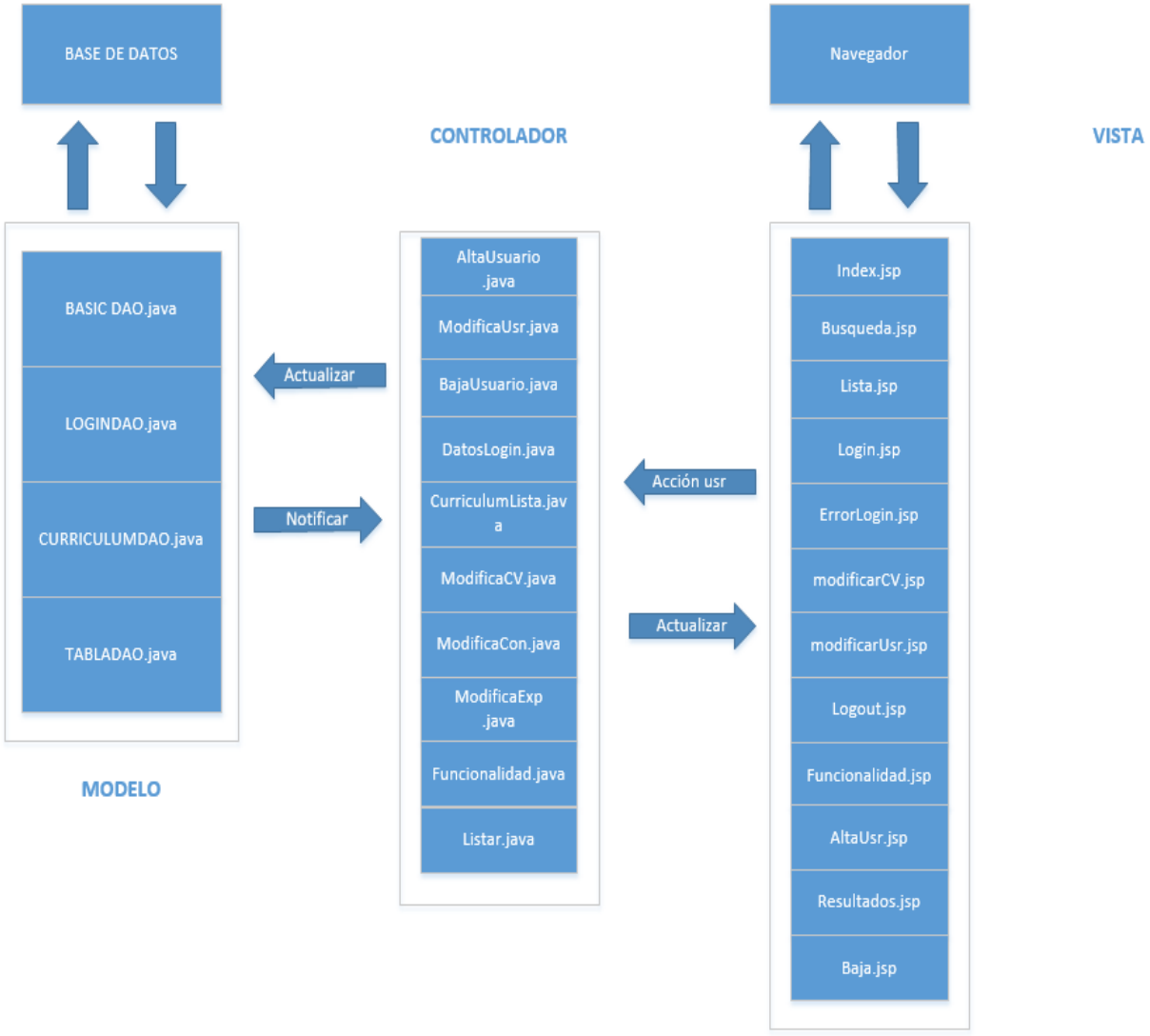
7 Arquitectura

7.1 Diagrama de subsistemas



8 Componentes

8.1 Diagrama de componentes


BASE DE DATOS
Navegador
CONTROLADOR
VISTA
BASIC DAO.java
LOGINDAO.java
CURRICULUMDAO.java
TABLADAO.java
MODELO
Actualizar
Notificar
AltaUsuario .java
ModificaUsr.java
BajaUsuario.java
DatosLogin.java
CurriculumLista.java
ModificaCV.java
ModificaCon.java
ModificaExp .java
Funcionalidad.java
Listar.java
Acción usr
Actualizar
Index.jsp
Busqueda.jsp
Lista.jsp
Login.jsp
ErrorLogin.jsp
modificarCV.jsp
modificarUsr.jsp
Logout.jsp
Funcionalidad.jsp
AltaUsr.jsp
Resultados.jsp
Baja.jsp

8.2 Diseño de componentes

1 **Capa de vistas** : se implementan mediante páginas JSP

2 **Capa de controladores** : en esta capa se encuentran los actions necesarios para la interacción del usuario con la aplicación.

Estas clases lo único que hacen es redirigir las peticiones a la capa de modelo, compuesta por fachadas y preparar los datos necesarios para las vistas. Dado que todas las clases Action hacen lo mismo no vamos a entrar en detalle.

3 **Capa de modelo** : en esta capa es donde **se implementa** fundamentalmente toda **la lógica de negocio**.
Esta capa está compuesta por una serie de clases desarrolladas mediante el **patrón Facade** que nos proporcionan interfaces sencillas para el acceso a la capa de persistencia.
Se podría haber optado por separar la lógica de negocio de las fachadas pero nuestra opinión es que el diseño actual **reduce la complejidad, es más comprensible y produce un acoplamiento débil**.

Lo consideramos como el **punto esencial de aplicación de los casos de uso**, por lo que **toda la funcionalidad requerida queda reflejada en las clases** de esta capa.
Esta capa interactúa directamente con la capa de persistencia.

3.1 **AccountManager** : proporciona una interfaz sencilla para el acceso, modificación y consulta de los datos asociados a las cuentas de usuario y administración. Implementada usando el **patrón Singleton** para evitar la creación de múltiples objetos en un mismo hilo.

3.2 **LoginManager** : proporciona una interfaz sencilla para gestionar los accesos al sistema para toda clase de usuarios. Hace las comprobaciones necesarias a la hora de logearse y genera un hash identificativo de la sesión creada.

3.3 **ProfilesManager** : proporciona una interfaz sencilla para el acceso, modificación y consulta de la información de los perfiles asociados a las cuentas de usuario. Permite manipular los datos de los perfiles bajo las restricciones impuestas en los requisitos (admin puede modificar cualquiera, usuarios solamente el suyo).

3.4 **ProfilesSearch** : permite realizar de forma sencilla las búsquedas habituales y contempladas en los casos de uso.

3.5 **UserSession** : una instancia de esta clase representa una sesión válida de usuario, se le asocia una entidad de clase Usuario y un hash identificativo de la sesión.

4 **Capa de persistencia** : se trata de la capa de acceso a datos. Se implementa haciendo uso de un mapeado objeto-relacional (ORM), la API de Java JPA (Java Persistence Api) y la implementación concreta de EclipseLink. Las clases de tipo entidad contienen anotaciones con el tipo de dato asociado, relaciones, cardinalidad de las relaciones, claves, índices y consultas básicas JPQL.

4.1 Entidades

4.1.1 **Usuarios** : representa a la entidad usuario. Los datos asociados al usuario y a su perfil estan en esta clase.

4.1.2 **Experiencias** : representa una experiencia labora.

4.1.3 **Funcionalidades** : representa una funcionalidad concreta (búsquedas, registro de usuarios, altas, bajas...)

4.1.4 **Intereses** : representa un conocimiento adquirido. Se trata simplemente de un tag que representa a ese conocimiento.

4.1.5 **TiposLikes** : clase sin usar

4.1.6 **Educación** : representa una educación reglada sin terminar o terminada.

4.2 **Controladores** : las clases controller básicamente interactuan con los drivers jdbc para ejecutar las consultas necesarias. Básicamente es como CRUD (Create Read Update Delete) + consultas SQL habituales.

5 Configuration

5.1 Tiene una clase que contiene los parámetros básicos de la aplicación

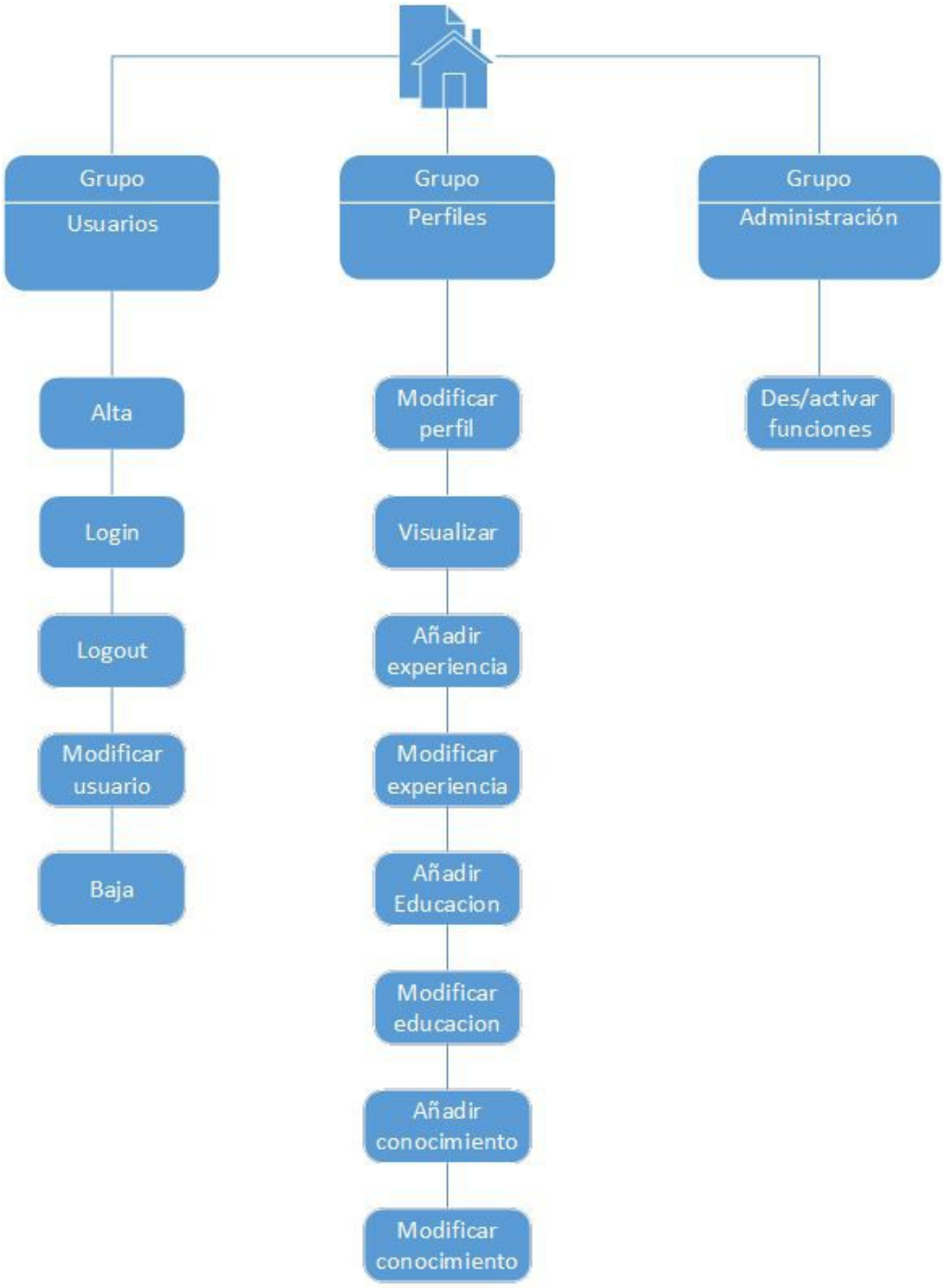
6 Pruebas

6.1 **AccountsManagerTest** : realiza una serie de pruebas que se corresponden con los casos de uso relativos a la gestión de usuarios (alta/baja, modificación y consulta).

6.2 **ProfilesManagerTest** : realiza una serie de pruebas que se corresponden con los casos de uso relativos a la gestión de perfiles

6.3 **FunctionManagerTest** : realiza una serie de pruebas que se corresponden con lso casos de uso relativos a la activación/desactivación de funcionalidades.

9 Mapa WEB


Grupo
Usuarios
Grupo
Perfiles
Grupo
Administración
Alta
Login
Logout
Modificar usuario
Baja
Modificar perfil
Visualizar
Añadir experiencia
Modificar experiencia
Añadir Educacion
Modificar educacion
Añadir conocimiento
Modificar conocimiento
Des/activar funciones

10 Tecnologías usadas

10.1 JPA

proporciona un modelo de persistencia basado en POJO´s para mapear bases de datos relacionales en Java. La persistencia de Java fue desarrollada por expertos de EJB 3.0 como parte de JSR 220, aunque su uso no se limita a los componentes software EJB. Se puede utilizar en aplicaciones web y aplicaciones clientes. Para ello, combina ideas y conceptos de persistencia, como Hibernate, Toplink y JDO. El mapeo objeto-relacional se realiza mediante anotaciones en las propias clases de entidad.

10.2 Tiles

Tiles permite a los desarrolladores definir fragmentos de páginas que pueden ser ensamblados en páginas completas en tiempo de ejecución. Estos fragmentos (tiles) pueden usarse como includes para reducir la duplicación de elementos comunes en las páginas o integrarse en otros tiles para desarrollar una serie de plantillas reutilizables. Estas plantillas permiten el desarrollo de un look homogéneo en toda la aplicación. Crecieron en popularidad como un componente del Framework Struts, y ahora está integrado con varios frameworks, como Spring y Struts 2.

10.3 Bootstrap

Es uno de los framweroks más populares a la hora de diseñar páginas de tipo "responsive" (que se organizan en función del espacio disponible, entre otras cualidades). Incluye botones, formularios de entrada, columnas, y otros objetos de página ya preformateados. La mezcla de Bootstrap con otros framewrosk nos proporciona una "caja de herramientas" poderosa para desarollar una web que se muestre correctamente en equipos de sobremesa y dispositivos móviles.

10.4 HSQLDB HyperSqlDB

Es el software que lidera las bases de datos relacionales en SQL escritas en Java. Ofrece una bbdd relacional pequeña, multihilo que se puede montar en memoria o en disco y soporta modos de servidor de forma embebida. Incluye una línea de comandos SQL potente y herramientas gráficas para querys. Asimismo, soporta un amplio catálogo de características SQLis the leading SQL relational database software written in Java. It offers a small, fast multithreaded and transactional database engine with in-memory and disk-based tables and supports embedded and server modes. It includes a powerful command line SQL tool and simple GUI query tools.HSQLDB supports the widest range of SQL Standard features seen in any open source database engine: SQL:2011 core language features and an extensive list of SQL:2011 optional features. It supports nearly full Advanced ANSI-92 SQL (BNF format). Many extensions to the Standard, including syntax compatibility modes and features of other popular database engines, are also supported.

10.5 Framework Struts

Struts es una herramienta de soporte para el desarrollo de aplicaciones Web bajo el patrón MVCbajo la plataforma Java EE (Java Enterprise Edition). Struts se desarrollaba como parte del proyecto Jakarta de la Apache Software Foundation, pero actualmente es un proyecto conocido como*Apache Struts*.

Struts permite reducir el tiempo de desarrollo. Su carácter de "*software libre*" y su compatibilidad con todas las plataformas en las que Java Entreprise esté disponible lo convierten en una herramienta altamente disponible.

11 Anexos

11.1 Ficheros de configuración

En este apartado incluiremos varios ficheros de configuración, como el struts.xml, web.xml y layout, que dan una idea de cómo está organizada la práctica.

CSS:

```
.rojo{
    color:red;
}
.negrita{
    font-weight: bold;
}
#p1{
    font-style: italic;
}

.etiqueta{
    background-color: #00b5ad!important;
    border-color: #00b5ad!important;
    color: #fff!important;
}

.etiqueta .before{
    //position: absolute;
    -webkit-transform: translateY(-50%)translateX(50%)rotate(-45deg);
    -ms-transform: translateY(-50%)translateX(50%)rotate(-45deg);
    transform: translateY(-50%)translateX(50%)rotate(-45deg);
    top: 50%;
    right: 100%;
    content: '';
    background-color: #e8e8e8;
    background-image: none;
    width: 1.56em;
    height: 1.56em;
    -webkit-transition: background .2s ease;
    transition: background .2s ease;
    background-color: #00b5ad!important;
}

.etiqueta .triangulo {
```

```
    background-color: #00b5ad!important;
    border-color: #00b5ad!important;
    color: #fff!important;
    margin-left: 1em;
    position: relative;
    padding-left: 1.5em;
    padding-right: 1.5em;
    border-radius: 0 .2857rem .2857rem 0;
}

fieldset{
    margin-bottom: 30px;
}
.foto-perfil{
    margin-top: 10px;
}
.page-header{
    background-image: url(/linkunedin3/files/header.png);
    height: 145px;
    margin-top: 0;
}
.slogan {
      color: white;
  /* background-color: #18212f; */
  width: 241px;
  /* height: 20px; */
  padding: 20px;
  /* opacity: 56; */
  margin: 10px;
  /* border: 1px solid white; */
}
```

Struts.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>

<!DOCTYPE struts-config PUBLIC
        "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
        "http://jakarta.apache.org/struts/dtds/struts-config_1_3.dtd">



<struts-config>
   <form-beans>
      <!--form-bean name="PerfilCompletoForm" type="com.myapp.struts.Controlador.Forms.PerfilCompletoForm"/-->
      <form-bean name="BusquedaForm" type="com.myapp.struts.Controlador.Forms.BusquedaForm"/>
      <!--form-bean name="ModificarExpe" type="com.myapp.struts.Controlador.Forms.ModificarExpe"/-->
                                                                <form-bean          name="EntradaModificarPerfilForm"
type="com.myapp.struts.Controlador.Forms.EntradaModificarPerfilForm"/>
      <form-bean name="FuncionalidadForm" type="com.myapp.struts.Controlador.Forms.FuncionalidadForm"/>
                                                                <form-bean          name="EntradaModificarConoForm"
type="com.myapp.struts.Controlador.Forms.EntradaModificarConoForm"/>
      <form-bean name="ConocimientoForm" type="com.myapp.struts.Controlador.Forms.ConocimientoForm"/>
      <form-bean name="ExperienciaForm" type="com.myapp.struts.Controlador.Forms.ExperienciaForm"/>
      <form-bean name="AltaForm" type="com.myapp.struts.Controlador.Forms.AltaForm"/>
      <form-bean name="ModifyProfile" type="com.myapp.struts.Controlador.Forms.ModifyProfile"/>
      <form-bean name="CreateProfileForm" type="com.myapp.struts.Controlador.Forms.CreateProfileForm"/>
      <form-bean name="CreateUserForm" type="com.myapp.struts.Controlador.Forms.CreateUserForm"/>
      <form-bean name="LoginForm" type="com.myapp.struts.Controlador.Forms.LoginForm"/>
      <form-bean name="EducacionForm" type="com.myapp.struts.Controlador.Forms.EducacionForm"/>
      <form-bean name="BajaForm" type="com.myapp.struts.Controlador.Forms.BajaForm"/>

   </form-beans>

   <global-exceptions>

   </global-exceptions>

   <global-forwards>
      <forward name="welcome"  path="/Welcome.do"/>
   </global-forwards>

   <action-mappings>
                <action input="/" name="EntradaModificarConoForm" path="/anadirConocimiento" scope="request"
type="com.myapp.struts.Controlador.Actions.AnadirConocimientoAction"/>
                              <action input="/" name="ExperienciaForm" path="/anadirExpe" scope="request"
type="com.myapp.struts.Controlador.Actions.AnadirExpeAction"/>
                           <action input="/" name="EducacionForm" path="/anadirEducacion" scope="request"
type="com.myapp.struts.Controlador.Actions.AnadirEducacionAction"/>
```

```
        <action input="/" name="EducacionForm" path="/modificarEducacion" validate="false" scope="request" type="com.myapp.struts.Controlador.Actions.ModificarEducacionAction">
            <forward name="success" path="/modificarPerfil.jsp"/>
            <forward name="error" path="/"/>
        </action>
                <action input="/" name="EntradaModificarPerfilForm" path="/modificarCono" scope="request" type="com.myapp.struts.Controlador.Actions.ModificarConoAction"/>
                    <action input="/" name="FuncionalidadForm" path="/modificarFunc" scope="request" type="com.myapp.struts.Controlador.Actions.ModificarFuncAction"/>
                    <action input="/" name="ExperienciaForm" path="/modificarExpe" scope="request" type="com.myapp.struts.Controlador.Actions.ModificarExpeAction">
            <forward name="success" path="/modificarPerfil.jsp" />
            <forward name="error" path="/" />
        </action>
                <action input="/" name="EntradaModificarPerfilForm" path="/modificarperfil" scope="request" type="com.myapp.struts.Controlador.Actions.ModificarPerfilAction"/>

            <action input="/formRegistro.jsp" name="AltaForm" path="/alta" scope="request" validate="true" type="com.myapp.struts.Controlador.Actions.AltaAction">
            <forward name="success" path="/modificarPerfil.jsp"/>
            <forward name="error" path="/formRegistro.jsp"/>
        </action>

                <action input="/" name="EntradaModificarPerfilForm" path="/modificarperfil" scope="request" type="com.myapp.struts.Controlador.Actions.ModificarPerfilAction">
            <forward name="success" path="/modificarPerfil.jsp"/>
            <forward name="error" path="/modificarPerfil.jsp"/>
        </action>

        <action input="/" name="BajaForm" path="/baja" type="com.myapp.struts.Controlador.Actions.BajaAction" validate = "false">
            <forward name="success" path="/listarUsuarios.jsp"/>
        </action>

        <action input="/" path="/insertarusuarios" type="com.myapp.struts.Controlador.Actions.InsertarUsuarios">
            <forward name="success" path="/listarUsuarios.jsp" />
        </action>

                                    <action input="/" name="ExperienciaForm" path="/vermodificarExpe" type="com.myapp.struts.Controlador.Actions.VerModificarExperiencia" validate="false">
            <forward name="success" path="/modificarExperiencia.jsp" />
        </action>

        <action path="/logout" type="com.myapp.struts.Controlador.Actions.LogoutAction" >
            <forward name="success" path="/logout.jsp"/>
```

```
        <forward name="error" path="/login.jsp"/>
      </action>
                                    <action input="/" name="LoginForm" path="/login" scope="request"
type="com.myapp.struts.Controlador.Actions.LoginAction">
        <forward name="success" path="/success.jsp"/>
        <forward name="error" path="/login.jsp"/>
      </action>

                  <action input="/" name="EntradaModificarPerfilForm" path="/ListarUsuarios" scope="session"
type="com.myapp.struts.Controlador.Actions.ListarUsuariosAction">
        <forward name="success" path="/listarUsuarios.jsp"/>
      </action>

                                <action input="/" name="BusquedaForm" path="/buscar" scope="session"
type="com.myapp.struts.Controlador.Actions.BuscarAction" validate="true">
        <forward name="success" path="/listarUsuarios.jsp"/>
        <forward name="error" path="/formBusqueda.jsp"/>
      </action>
              <action input="/" name="EntradaModificarPerfilForm" path="/vermodificarperfil" scope="request"
type="com.myapp.struts.Controlador.Actions.VerModificarPerfil" validate="false">
        <forward name="success" path="/modificarPerfil.jsp"/>
      </action>

         <action input="/formFuncionalidades.jsp" name="FuncionalidadForm" path="/funcionalidad" scope="request"
validate="true" type="com.myapp.struts.Controlador.Actions.ModificarFuncAction">
        <forward name="success" path="/listarFuncionalidades.jsp"/>
        <forward name="error" path="/modificarFuncionalidad.jsp"/>
      </action>

      <!-- fixme juan poner que sea durante la sesion -->
                  <action input="/" name="FuncionalidadForm" path="/listarFuncionalidades" scope="request"
type="com.myapp.struts.Controlador.Actions.ListarFuncionalidadesAction">
        <forward name="success" path="/listarFuncionalidades.jsp"/>
      </action>

                   <action input="/" name="EducacionForm" path="/vermodificarEducacion" scope="request"
type="com.myapp.struts.Controlador.Actions.VerModificarEducacion" validate="false">
        <forward name="success" path="/modificarEducacion.jsp" />
      </action>

                              <action input="/" name="EducacionForm" path="/BorraEdu" scope="request"
type="com.myapp.struts.Controlador.Actions.BorraEduAction" validate="false">
        <forward name="success" path="/modificarPerfil.jsp"/>
      </action>

                            <action input="/" name="EducacionForm" path="/AnadirEdu" scope="request"
type="com.myapp.struts.Controlador.Actions.AnadirEduAction" validate="false"/>
```

```
                    <action input="/" name="ExperienciaForm" path="/BorraExpe" scope="request"
type="com.myapp.struts.Controlador.Actions.BorrarExpeAction" validate="false">
      <forward name="success" path="/modificarPerfil.jsp"/>
    </action>

                    <action input="/" name="EntradaModificarConoForm" path="/borrarCono" scope="request"
type="com.myapp.struts.Controlador.Actions.BorrarConoAction" validate="false">
      <forward name="success" path="/modificarPerfil.jsp"/>
    </action>

    <action path="/Welcome" forward="/welcomeStruts.jsp"/>

  </action-mappings>

  <controller processorClass="org.apache.struts.tiles.TilesRequestProcessor"/>

  <message-resources parameter="com/myapp/struts/ApplicationResource"/>

  <plug-in className="org.apache.struts.tiles.TilesPlugin" >
    <set-property property="definitions-config" value="/WEB-INF/tiles-defs.xml" />
    <set-property property="moduleAware" value="true" />
  </plug-in>

  <!-- ========================= Validator plugin ================================= -->
  <plug-in className="org.apache.struts.validator.ValidatorPlugIn">
    <set-property
      property="pathnames"
      value="/WEB-INF/validator-rules.xml,/WEB-INF/validation.xml"/>
  </plug-in>

</struts-config>
```

Persistence.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1" xmlns="http://xmlns.jcp.org/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
  <persistence-unit name="PracticaPruebastwebPUMySQL" transaction-type="RESOURCE_LOCAL">
    <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>
    <exclude-unlisted-classes>false</exclude-unlisted-classes>
    <properties>
                        <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/mydb?
zeroDateTimeBehavior=convertToNull"/>
      <property name="javax.persistence.jdbc.password" value="admin"/>
      <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver"/>
      <property name="javax.persistence.jdbc.user" value=""/>
      <property name="javax.persistence.schema-generation.database.action" value="drop-and-create"/>
    </properties>
  </persistence-unit>
  <persistence-unit name="PracticaPruebastwebPUHsql" transaction-type="RESOURCE_LOCAL">
    <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>
    <exclude-unlisted-classes>false</exclude-unlisted-classes>
    <properties>
      <property name="javax.persistence.jdbc.url" value="jdbc:hsqldb:mem:hsql://&lt;HOST&gt;[:&lt;PORT&gt;]"/>
      <property name="javax.persistence.jdbc.password" value=""/>
      <property name="javax.persistence.jdbc.driver" value="org.hsqldb.jdbcDriver"/>
      <property name="javax.persistence.jdbc.user" value="SA"/>
      <property name="javax.persistence.schema-generation.database.action" value="create"/>
    </properties>
  </persistence-unit>
</persistence>
```