AUT06_03 Proyecto MiAli

1. Creación del proyecto y dependencias

1. Se creará un proyecto de nombre **MiAli** (en singular, para no pisar el nombre del primer ejercicio), Maven y Java EE 7.
2. Se añadirá un descriptor de despliegue jboss-web.xml y se establecerá el nombre de contexto a **miAli_ApellidoNombre**
3. Se añadirá el framework **Java Server Faces 2.2** y se escogerá si quieren como librería de implementación **PrimeFaces**
4. Posteriormente, actualizaremos **pom.xml**, para que la dependencia sea a PrimeFaces 6.0 y no 5.0.
5. Se añadirá al pom.xml las dependencias necesarias para utilizar **slf4j** como logger, con binding a **Log4j (Obligatorio)**.

1.1 Configuración de base de datos

1. Se creará una base de datos MySQL, de nombre **miali**dB
2. ~~Se definirá un pool de conexiones de nombre **MiAliPool**~~
3. Se definirá un recurso DataSource, con nombre JNDI **java:/miAliDB**

1.2 Configuración de seguridad (wildfly)

1. Se creará un realm de tipo JDBC y nombre **miali-realm**
2. Este realm utilizará el datasource, con nombre JNDI **jdbc/miAliDB**
3. Las tablas de usuarios y grupos de usuarios pueden copiar el esquema del ejercicio de seguridad, de la unidad 5. Habrá que crear tablas similares en la base de datos **mialiDB**.
4. Los grupos disponibles serán: Administradores, Gestores, Clientes

1.3 Configuración de seguridad (aplicación web)

1. Se definirá, como mecanismo de login, **autenticación por formularios**.
2. Se definirán tres roles: ADMIN, MANAGER, CUSTOMER
3. Se garantizará la confidencialidad de contraseñas.
4. Se mapearán grupos contra roles de la manera siguiente:
    1. Administradores → ADMIN
    2. Gestores → MANAGER
    3. Clientes → CUSTOMER

2. Base de datos

La base de datos mialiDB almacenará datos de productos, habitaciones, clientes y reservas.

Se deberá comenzar a asegurar la integridad de los datos, creando todas las restricciones de: clave primaria, clave única, clave ajena, campos obligatorios, restricciones checks, defaults.

Muy conveniente definir los índices para búsqueda que se consideren.

2.1 Tablas de usuarios y grupos

Al estilo de las usadas en el ejemplo de la unidad 5. Pueden adaptarse de él para este ejercicio.

2.2 Producto

NOTA: Se recomienda añadir campos subrogados de tipo id_cliente o similar. Asegurar si se usan que la clave primaria original no se repite.

La entidad Product almacenará: el nombre del producto (ej: Mesa Grande), así como el municipio, provincia y precio.

Obviamente los usuarios podrán elegir la cantidad de productos deseados. Se podría añadir una columna de stock a la base de datos.

2.3 Clientes

Se almacenará: login y contraseña y, además: nombre, apellidos, NifNie, nacionalidad, correo electrónico y teléfono fijo y móvil. Con formato controlado.

2.4 Resto de tablas

Se crearán las tablas adicionales que se precisen para resolver el proyecto.

3. Casos de uso

3.1 Acceso bienvenida [1 punto]

Ubicada en **/index.xhtml**

La aplicación mostrará una JSF index, como welcome page. Esta página podrá definir un cuadro de texto e invocar al bean de búsqueda de productos con ese texto. El bean de búsqueda de productos podría mandarnos a la página de resultados, si encuentra algo, o a la página principal de búsqueda si no ha encontrado nada con el cuadro de texto de búsqueda de la página inicial.

Pinchando en la imagen accederíamos a la página principal de búsqueda.

Esta página será similar a la siguiente:

Desde esta página podremos ir a:

Botón **Identificarse → /login.xhtml**

Botón **Registro → /customer/register.xhtml**

Botón **Lupa → ProductSearchBean o vamos a /search/index.xhtml**

Click en la imagen → **/search/index.xhtml**

3.2 Registro de usuario [2 puntos]

Página en **/customer/register.xhtml**

Debe almacenar los datos del cliente, obviamente si el cliente no está ya dado de alta (no hay un NifNie, email o login igual en base de datos)

A la hora de insertar el cliente, se añadirán sus datos personales y, además, hay que añadir una entrada en la tabla de users (la de login) y asociar al usuario al grupo de base de datos Clientes (todo esto para autorización).

3.2.1 Información mínima a recoger

La de la tabla de customers

3.2.2 Validaciones

Se validarán campos obligatorios, contraseña y confirmación de contraseña iguales, formato de correo electrónico, de fecha, etc.

3.3 Registro de productos[2 puntos]

Página en /manager/create.xhtml

NOTA: A las opciones de manager se puede llegar a través de un link en la cabecera que se muestre si el usuario tiene el rol MANAGER o en una página adicional que tenga los link de manager.

Debe almacenar los datos del producto, obviamente si el producto no está ya dado de alta (no hay un producto con el mismo nombre y categoría en el mismo municipio)

Se deberá poder subir, al menos, una imagen de cada producto. Lo ideal sería poder subir varias y tener una galería de imágenes subidas.

3.3.1 Información mínima a recoger

Se recogeran los campos incluidos en la tabla productoffer

3.3.2 Validaciones

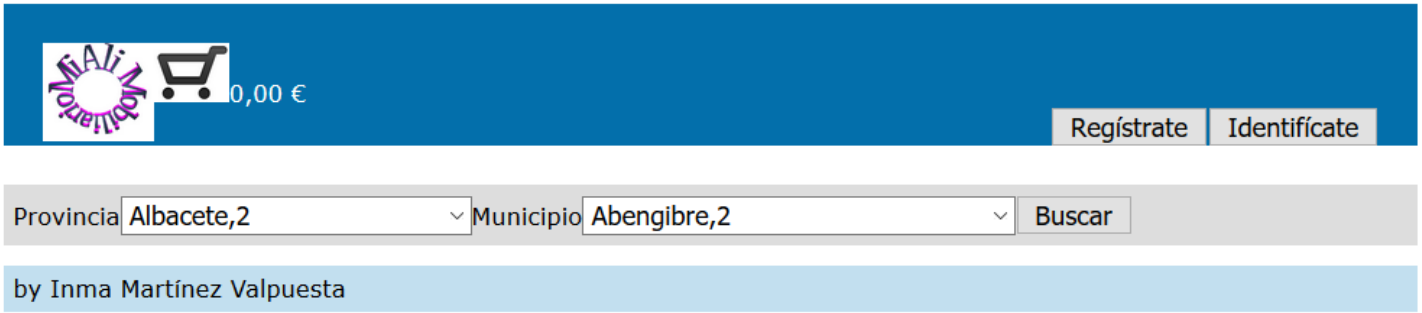
Se validarán campos obligatorios, numericos mayor que 0.

3.4 Página principal de búsqueda [2 puntos]

Ubicada en **/search/index.xhtml**

La página principal de búsqueda contiene varias secciones:

1. Barra de identificación / registro
   i. Sólo visible si el usuario no está autenticado.
   ii. Permite acceder al formulario de login **/login.xhtml** o registro **/newCustomer.xhtml**
2. Formulario principal de búsqueda



3. Información del carro de compra
   i. Mostrará el número de productos añadidos al carrito en sesión, así como el total de la compra.

3.5 Página de resultados [2 puntos]

Ubicada en **/search/results.xhtml**

Mostrará las ofertas disponibles para los productos encontrados en el formulario de búsqueda. Para aquellos productos de la zona que no tengan plazas puede optar por no mostrarlos, o mostrarlos con 0 plazas.

Las ofertas podrán añadirse al carrito de la compra, el cual mostrará la información actualizadas del número de paquetes reservados y precio total.

NOTA: Pinchando en el carrito de la compra debería verse un resumen de lo comprado, en otra página.

3.6 Login [1 punto]

Realizará una autenticación estándar basada en formularios o utilizando un ManagedBean.

Una vez que el usuario esté autenticado se mantendrá un Managed Bean de sesión, con los datos principales del usuario (login y nombre, al menos).

En la página siguiente se discuten diversas opciones para, una vez, validado, tener un bean de sesión con la información del usuario validado.

http://stackoverflow.com/questions/15022473/how-to-trigger-bean-after-j-security-check-complete-authentication-in-java

4. Fecha de entrega

Por determinar

5. Criterios de corrección aplicables

5.1 Puntuación

Acceso bienvenida	**1 punto**
Registro de cliente	**2 puntos**
Registro de producto	**2 puntos**
Página principal de búsqueda	**2 puntos**
Pagina de resultados	**2 puntos**
Login	**1 punto**

5.2 Descuentos

i. **Se descontarán 4 puntos si no compila o se ejecuta el programa, independientemente de lo que se haya hecho.**

ii. **Se descontará 1 punto si se detecta que no se cierran recursos de base de datos**

iii. **Se descontará 1 punto en caso de errores graves de validación**

iv. **Se descontará 0,5 puntos en el caso de que se defina mal el ámbito de alguna variable.**

v. **Se descontará 0,25 puntos si se encuentran errores en el nombrado de elementos Java (clases, paquetes, variables, …)**

vi. **Se descontará 0,5 puntos en el que caso de que una excepción no tratada termine mostrando una página en blanco.**

vii. **Se descontará 1 punto si no se usa el fichero propierties**