.10. Autenticación y autorización (*MemoryRealm*)

Protege el acceso a la aplicación **curso** utilizando un *MemoryRealm* con autenticación BASIC ara que solo puedan acceder los usuarios **profesor** y **alumno** (pertenecientes al **role curso**). La ıformación sobre usuarios y roles se almacenará en el fichero **$CATALINA_BASE/conf/tomcat-sers.xml**.

1. **Eliminar la aplicación curso del servidor.**

   1.1. Inicia sesión en **DesarrolloW7XX**.

   1.2. Elimina la aplicación **curso** del servidor *Tomcat* (puedes usar el *manager* o la tarea de *Ant*).

2. **Crear el fichero con los usuarios y roles.**

   2.1. Inicia sesión en **ServidorLinuxXX** con un usuario privilegios de administración.

   2.2. Edita el fichero **/var/lib/tomcat7/conf/tomcat-users.xml** y añade los siguientes elementos, Figura 8.81.

```
<role rolename="manager-gui"/>
<role rolename="manager-script"/>
<role rolename="curso"/>
<user username="tomcat" password="despliegue" roles="manager-gui"/>
<user username="tomcatscript" password="despliegue" roles="manager-script"/>
<user username="alumno" password="alumno" roles="curso"/>
<user username="profesor" password="profesor" roles="curso"/>
```

Figura 8.81: Fichero **/var/lib/tomcat7/conf/tomcat-users.xml**

   2.3. Reinicia *Tomcat* para que se lea de nuevo el fichero.

3. **Configurar el *Realm*.**

   3.1. En **DesarrolloW7XX** crea un fichero **context.xml** para la aplicación **curso** (**META-INF/context.xml**) y añade los siguientes elementos, Figura 8.82.

```
<Context>
    <Realm className="org.apache.catalina.realm.MemoryRealm" />
</Context>
```

Figura 8.82: Fichero **META-INF/context.xml**

4. **Proteger la aplicación con el *MemoryRealm* (Autenticación BASIC).**

   4.1. En **DesarrolloW7XX** edita el descriptor de despliegue (**web.xml**) de la aplicación **curso** y añade los siguientes elementos, Figura 8.83.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
        http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
    id="WebApp_ID" version="3.0">

    <display-name>curso</display-name>
    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
    </welcome-file-list>

    <security-constraint>
        <web-resource-collection>
            <web-resource-name>MemoryRealm</web-resource-name>
            <url-pattern>/*</url-pattern>
        </web-resource-collection>
        <auth-constraint>
            <role-name>curso</role-name>
        </auth-constraint>
    </security-constraint>

    <login-config>
        <auth-method>BASIC</auth-method>
        <realm-name>Acceso al curso</realm-name>
    </login-config>

    <servlet>
        <servlet-name>HolaServlet</servlet-name>
        <servlet-class>net.daw01.curso.HolaServlet</servlet-class>
    </servlet>
```

Figura 8.83: Fichero **web.xml** de la aplicación **curso**

4.2. Elimina la aplicación **curso** del servidor *Tomcat*.

4.3. Genera el fichero **curso.war** con los cambios.

4.4. Despliega el nuevo fichero **curso.war**.

5. **Proteger la aplicación con el *MemoryRealm* y configurar la autenticación (BASIC).**

5.1. Accede a la aplicación y observa que se pide usuario y contraseña. Solo puedes acceder como alumno o profesor, Figuras 8.84 y 8.85.

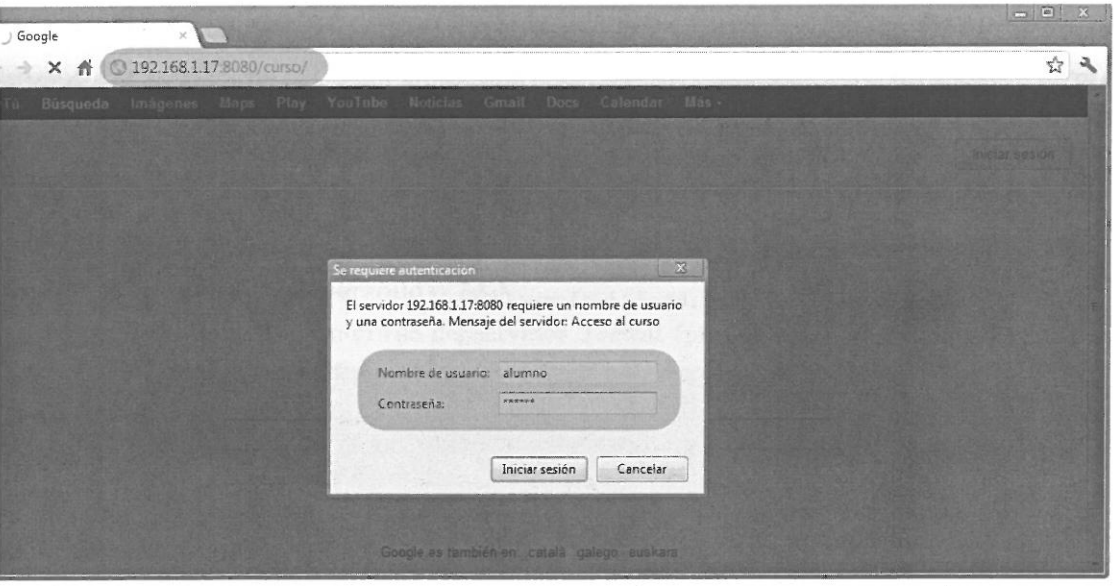

Figura 8.84: Acceso a la aplicación curso

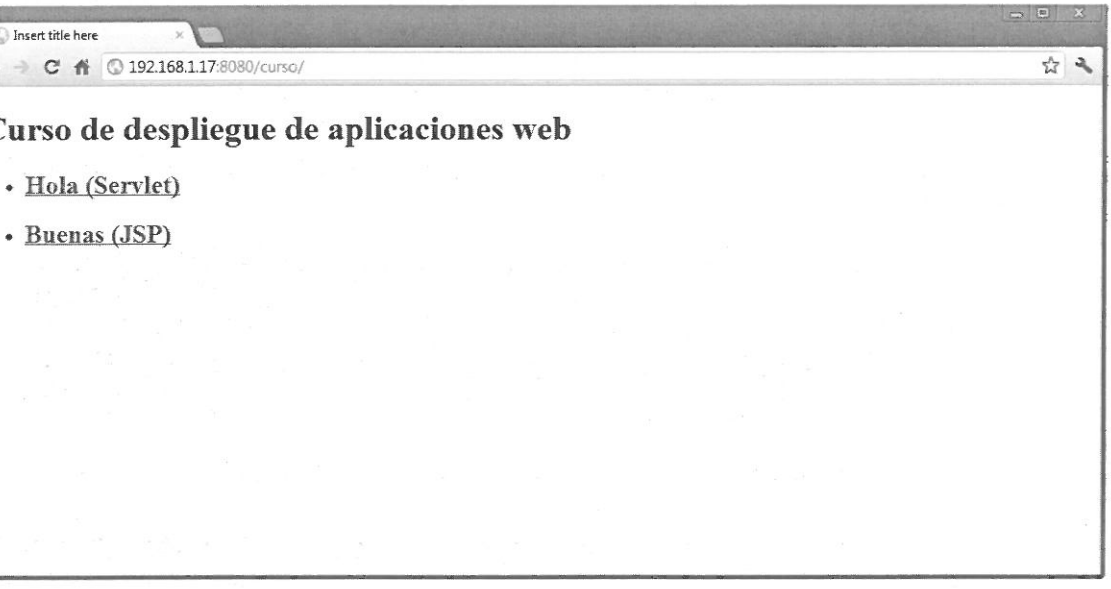

Figura 8.85: Acceso a la aplicación curso

8.11. Autenticación y autorización (*UserDatabaseRealm*)

Analiza la configuración del *UserDatabaseRealm* definido por defecto en *Tomcat* y cómo se utiliza en la aplicación ***manager***.

1. Inicia sesión en **ServidorLinuxXX** con un usuario privilegios de administración.
2. Consulta el fichero **/var/lib/tomcat7/server.xml** y observa el recurso JNDI que define un *UserDatabase*, Figura 8.86.

```
<!-- Global JNDI resources
     Documentation at /docs/jndi-resources-howto.html
-->
<GlobalNamingResources>
  <!-- Editable user database that can also be used by
       UserDatabaseRealm to authenticate users
  -->
  <Resource name="UserDatabase" auth="Container"
            type="org.apache.catalina.UserDatabase"

            description="User database that can be updated and saved"
            factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
            pathname="conf/tomcat-users.xml" />
</GlobalNamingResources>
```

Figura 8.86: Fichero **/var/lib/tomcat7/server.xml**

3. Consulta el fichero **/var/lib/tomcat7/server.xml** y observa *UserDatabaseRealm* definido a nivel de `<Engine>`, Figura 8.87.

```
<!-- This Realm uses the UserDatabase configured in the global JNDI
     resources under the key "UserDatabase".  Any edits
     that are performed against this UserDatabase are immediately
     available for use by the Realm.  -->
      <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
       resourceName="UserDatabase"/>
```

Figura 8.87: Fichero **/var/lib/tomcat7/server.xml**

4. Consulta del descriptor de despliegue de la aplicación manager en **/usr/share/tomcat7-admin /manager/WEB-INF/web.xml** y observa cómo se protege el acceso a la misma utilizando el *UserDatabaseRealm*, Figuras 8.88 y 8.89.

```
<security-constraint>
  <web-resource-collection>
    <web-resource-name>HTML Manager interface (for humans)</web-resource-name>
    <url-pattern>/html/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
     <role-name>manager-gui</role-name>
  </auth-constraint>
</security-constraint>
<security-constraint>
  <web-resource-collection>
    <web-resource-name>Text Manager interface (for scripts)</web-resource-nam$
    <url-pattern>/text/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
     <role-name>manager-script</role-name>
  </auth-constraint>
</security-constraint>
<security-constraint>
```

Figura 8.88: Fichero **/usr/share/tomcat7-admin/manager/WEB-INF/web.xml**

```
<!-- Define the Login Configuration for this Application -->
<login-config>
  <auth-method>BASIC</auth-method>
  <realm-name>Tomcat Manager Application</realm-name>
</login-config>
```

Figura 8.89: Fichero **/usr/share/tomcat7-admin/manager/WEB-INF/web.xml**

8.12. Autenticación y autorización (*JDBCRealm*)

Protege el acceso a la aplicación **compras** utilizando un *JDBCRealm* con autenticación FORM para que solo puedan acceder los usuarios **mortadelo** y **filemon** (pertenecientes al *role* **compras**).

1. **Eliminar la aplicación curso del servidor**.
   1.1. Inicia sesión en **DeasarrolloWindowsXX**.
   1.2. Elimina la aplicación **compras** del servidor *Tomcat* (puedes usar el *manager* o la tarea de *Ant*).
2. **Configuración de la base de datos**.
   2.1. Inicia sesión en **DeasarrolloWindowsXX**.
   2.2. Inicia una navegador y accede a `http://192.168.1.X7/phpmyadmin`.
   2.3. Inicia sesión con el usuario **root**.
   2.4. Accede a **Privilegios**, pincha en **Agregar un nuevo usuario**. Introduce **tomcatusers** como nombre de usuario, **localhost** como servidor y **tomcatusers** como clave del usuario. Marca la opción **Crear base de datos con el mismo nombre y otorgar todos los privilegios** y pincha en **Crear Usuario**, Figura 8.90.

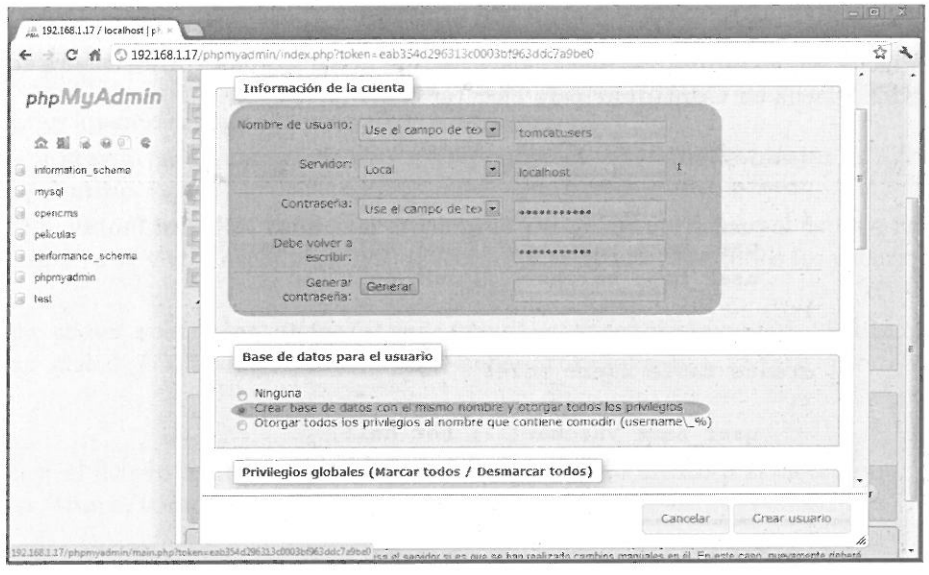

Figura 8.90: Crear el usuario y la base de datos **tomcatusers**

2.5. Se han creado el usuario **tomcatusers**, la base de datos **tomcatusers** y se han otorgado todos los privilegios al usuario sobre la base de datos, Figura 8.91.

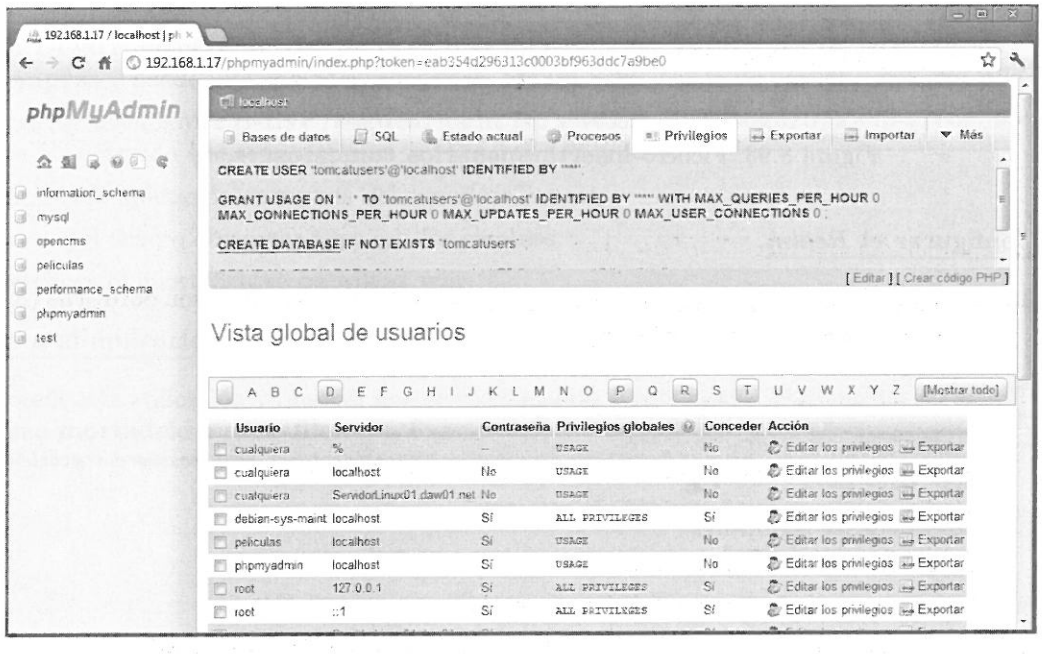

Figura 8.91: Usuario y la base de datos creados

2.6. Cierra la sesión de *phpmyadmin* del usuario **root.**

2.7. Inicia sesión en *phpmyadmin* con el usuario **tomcatusers.**

2.8. En la parte izquierda selecciona la base de datos **tomcatusers.**

2.9. Pincha en **Importar**.

2.10. Pincha en **Examinar** y selecciona el script **crear_tablas_tomcatusers.sql**, Figura 8.92. Pincha en **Continuar** para ejecutar el *script*.

```
use tomcatusers;
create table users
(
    user_name varchar(12) not null primary key,
    user_pass varchar(12) not null
);

create table users_roles
(
    user_name varchar(12) not null,
    role_name varchar(12) not null,
    primary key(user_name, role_name)
);
```

Figura 8.92: Fichero **crear_tablas_tomcatusers.sql**

2.11. Pincha en **Examinar** y selecciona el script **insertar_usuarios_tomcatusers.sql**, Figura 8.93. Pincha en **Continuar** para ejecutar el *script*.

```
insert into users values("mortadelo", "mortadelo");
insert into users values("filemon", "filemon");
insert into users_roles values("mortadelo", "compras");
insert into users_roles values("filemon", "compras");
```

Figura 8.93: Fichero **insertar_usuarios_tomcatusers.sql**

3. **Configurar el *Realm*.**

3.1. En **DesarrolloW7XX** crea un fichero **context.xml** para la aplicación **compras** (**META-INF/context.xml**) y añade los siguientes elementos, Figura 8.94.

```
:Context>
   <Realm  className="org.apache.catalina.realm.JDBCRealm"
           driverName="com.mysql.jdbc.Driver"
           connectionURL="jdbc:mysql://localhost/tomcatusers?user=tomcatusers&amp;password=tomcatusers"
           userTable="users"
           userNameCol="user_name" userCredCol="user_pass"
           userRoleTable="users_roles"
           roleNameCol="role_name"
   />
:/Context>
```

Figura 8.94: Fichero **META-INF/context.xml**

3.2. Para que la aplicación se pueda conectar a *MySQL* usando JDBC es necesario que el driver JDBC esté disponible. Podemos añadirlo en el directorio **WEB-INF/lib** de la aplicación o en **$CATALINA_HOME/lib**. Vamos a optar por la segunda opción y así estará disponible para todas las aplicaciones del servidor.

a. Accede a `http://dev.mysql.com/downloads/connector/j/` y descarga el driver JDBDC para *MySQL*.

b. Descomprime el fichero descargado para obtener el fichero (en el momento de realizar la práctica **mysqlconnectorjava5.1.25bin.jar**).

c. Usa *Filezilla* para subir el fichero **mysqlconnectorjava5.1.25bin.jar** a **/home/alumno**.

d. En **ServidorLinuxXX** cambia el propietario y el grupo del fichero al usuario **root** y grupo **root** y concédele permisos de lectura y escritura para todos los usuarios.

```
sudo chown root:root /home/alumno/mysql-connector-java-5.1.25-bin.jar
sudo chmod 777 /home/alumno/mysql-connector-java-5.1.25-bin.jar
```

e. Mueve el fichero desde del directorio **home** del usuario **alumno** al directorio **/usr/share/tomcat7/lib**.

```
sudo mv /home/alumno/mysql-connector-java-5.1.20-bin.jar /usr/share/tomcat7/lib
```

3.3. Reinicia *Tomcat*.

4. **Proteger la aplicación con el *JDBCRealm* (Autenticación FORM).**

4.1. En **DesarrolloW7XX** edita el descriptor de despliegue (**web.xml**) de la aplicación **compras** y añade los siguientes elementos, Figura 8.95.

4.2. Crea los ficheros **WEB-INF/seguro/login.jsp** y **WEB-INF/seguro/login-error.jsp** con el contenido que se muestra en las Figuras 8.96 y 8.97.

4.3. Elimina la aplicación **compras** del servidor *Tomcat*.

4.4. Genera el fichero **compras.war** con los cambios.

4.5. Despliega el nuevo fichero **compras.war**.

5. **Acceso a al aplicación.**

5.1. Accede a la aplicación y observa que se pide usuario y contraseña. Solo puedes acceder como **mortadelo** o como **filemon**, Figuras 8.98 y 8.99.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://java.sun.com/xml/ns/javaee" xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
    id="WebApp_ID" version="3.0">

    <display-name>compras</display-name>

    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>

    <security-constraint>
        <web-resource-collection>
            <web-resource-name>JDBCRealm</web-resource-name>
            <url-pattern>/*</url-pattern>
        </web-resource-collection>
        <auth-constraint>
            <role-name>compras</role-name>
        </auth-constraint>
    </security-constraint>

    <login-config>
        <auth-method>FORM</auth-method>
        <realm-name>Compras</realm-name>
        <form-login-config>
            <form-login-page>/WEB-INF/seguro/login.jsp</form-login-page>
            <form-error-page>/WEB-INF/seguro/login-error.jsp</form-error-page>
        </form-login-config>
    </login-config>

    <session-config>
        <session-timeout>15</session-timeout>
    </session-config>
```

Figura 8.95: Fichero **web.xml** de la aplicación compras

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>

    <form method="POST"
        action='<%=response.encodeURL("j_security_check")%>'>
        Usuario:<input type="text" name="j_username">
        Password:<input type="password" name="j_password">
        <input type="submit" value="Login">
    </form>

</body>
</html>
```

Figura 8.96: Fichero **WEB-INF/seguro/login.jsp**

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Página de error de login</title>
</head>
<body>

Usuario o password icorrectos, prueba <a href="<%= response.encodeURL("index.jsp") %>"> de nuevo</a>.

</body>
</html>
```

Figura 8.97: Fichero **WEB-INF/seguro/login.jsp**

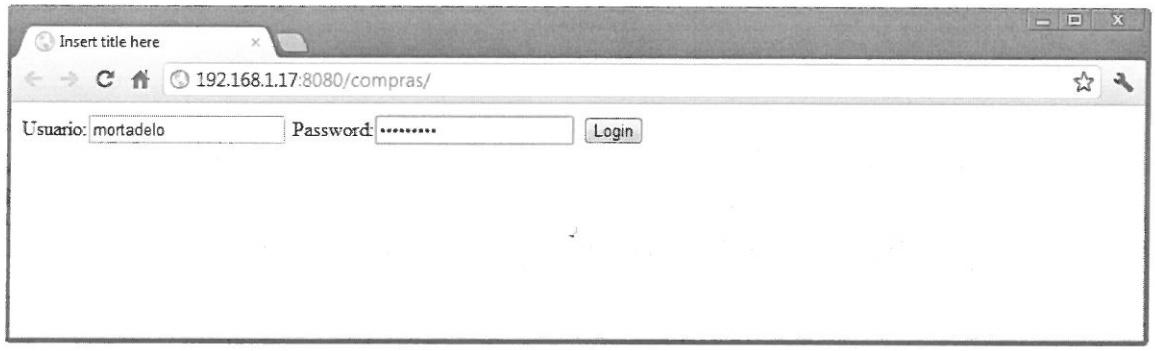

Figura 8.98: Acceso a la aplicación compras(1)

Figura 8.99: Acceso a la aplicación compras(2)