

*Autor: José Fco. Soto Camacho*

*17 de Marzo de 2019*

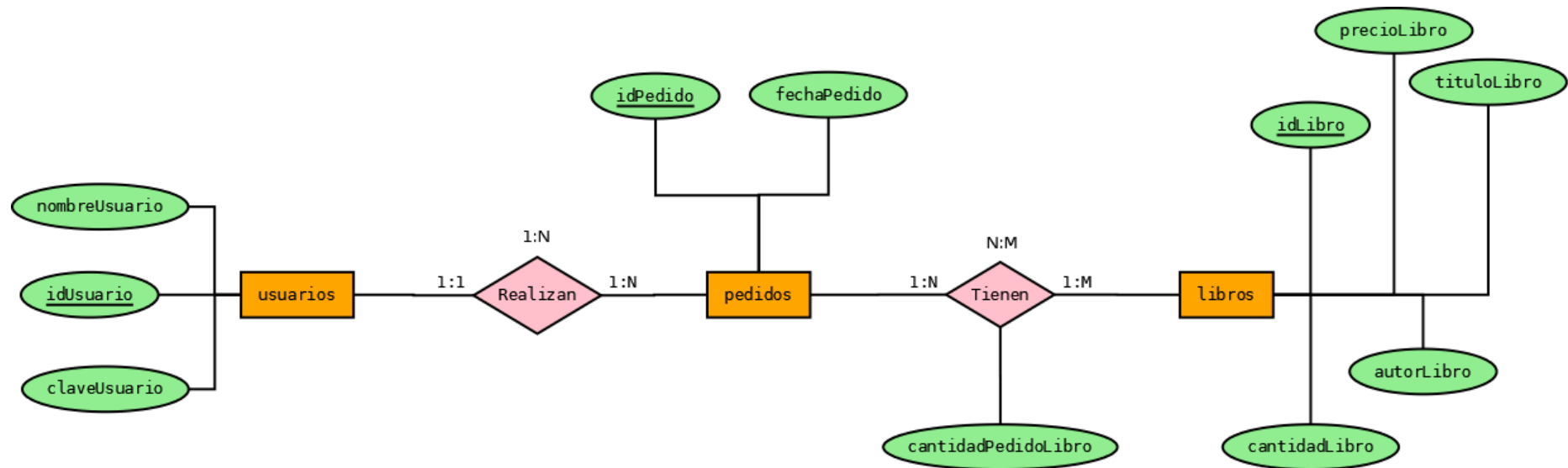
# Práctica Tema 4

*Despliegue de Aplicaciones Web*

## Índice

<b>1 – Diagrama Entidad Relación</b>	<b>3</b>
<b>2 – Esquema Relacional</b>	<b>4</b>
<b>3 – Creación de la Base de Datos</b>	<b>4</b>
3.1 Tabla libros	4
3.2 Tabla pedidos	5
3.3 Tabla pedidosLibros	6
3.4 Tabla usuarios	7
<b>4 – Código de la Parte del Servidor</b>	<b>8</b>
4.1 LoginServlet.java	8
4.2 LogoutServlet.java	15
4.3 Index.html	18
4.4 Libros.jsp	19
4.5 Pedidos.jsp	22
<b>5 – Código de la Parte del Servidor</b>	<b>26</b>
5.1 ElementoPedido.java	26
5.2 LibrosMVC.java	28
5.3 ServletControlador.java	32
5.4 checkout.jsp	40
5.5 order.jsp	42
<b>6 – Resultados</b>	<b>46</b>
6.1 Cliente	46
6.2 Servidor	47
<b>7 – Valoración Personal</b>	<b>49</b>
<b>8 – Referencias</b>	<b>49</b>
8.1 Grupo Studium	49

## 1 – Diagrama Entidad Relación



## 2 – Esquema Relacional

### Entidades Directas:

- Usuarios (idUsuario, nombreUsuario, claveUsuario)
- Pedidos (idPedido, fechaPedido, idUsuarioFK1)
- Libros (idLibro, tituloLibro, autorLibro, precioLibro, cantidadLibro)

### Entidades Creadas por Relaciones N:M

- PedidosLibros (idPedidoLibro, idLibroFK2, idPedidoFK3, cantidadPedidoLibro)

## 3 – Creación de la Base de Datos

```
CREATE DATABASE tiendaLibros CHARSET utf-8 COLLATE utf8_spanish2_ci  
USE tiendaLibros
```

### 3.1 Tabla libros

```
CREATE TABLE `libros` (  
  `idLibro` int(11) NOT NULL,  
  `tituloLibro` varchar(45) COLLATE utf8_spanish2_ci DEFAULT NULL,  
  `autorLibro` varchar(45) COLLATE utf8_spanish2_ci DEFAULT NULL,  
  `precioLibro` decimal(10,2) DEFAULT NULL,  
  `cantidadLibro` int(11) DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_spanish2_ci;
```

DESCRIBE libros;

```
mysql> DESCRIBE libros;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra           |
+-----+-----+-----+-----+-----+-----+
| idLibro    | int(11)       | NO   | PRI | NULL    | auto_increment |
| tituloLibro | varchar(45)   | YES  |     | NULL    |                 |
| autorLibro  | varchar(45)   | YES  |     | NULL    |                 |
| precioLibro | decimal(10,2) | YES  |     | NULL    |                 |
| cantidadLibro | int(11)      | YES  |     | NULL    |                 |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.01 sec)
```

INSERT INTO `libros` (`idLibro`, `tituloLibro`, `autorLibro`, `precioLibro`, `cantidadLibro`) VALUES

(1, 'El señor de los anillos', 'J.R.R. Tolkien', 60.00, 10),

(2, 'El capitán Alatriste', 'Arturo Pérez Reverte', 25.00, 20),

(3, 'Harry Potter y La Piedra Filosofal', 'J.K. Rowling', 30.00, 15),

(4, 'Percy Jackson y Los Dioses del Olimpo: El Lad', 'Rick Riordan', 30.00, 20);

Result Grid					
Filter Rows:		Edit:		Export/Import:	
	idLibro	tituloLibro	autorLibro	precioLibro	cantidadLibro
▶	1	El señor de los anillos	J.R.R. Tolkien	60.00	10
	2	El capitán Alatriste	Arturo Pérez Reverte	25.00	20
	3	Harry Potter y La Piedra Filosofal	J.K. Rowling	30.00	15
	4	Percy Jackson y Los Dioses del Olimpo: El Lad	Rick Riordan	30.00	20
*	NULL	NULL	NULL	NULL	NULL

## 3.2 Tabla pedidos

CREATE TABLE `pedidos` (

`idPedido` int(11) NOT NULL,

`fechaPedido` date DEFAULT NULL,

`idUsuarioFK1` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_spanish2\_ci;



DESCRIBE pedidos;

```
mysql> DESCRIBE pedidos;
+-----+-----+-----+-----+-----+-----+
| Field      | Type   | Null | Key | Default | Extra           |
+-----+-----+-----+-----+-----+-----+
| idPedido   | int(11) | NO   | PRI | NULL    | auto_increment |
| fechaPedido | date   | YES  |     | NULL    |                 |
| idUsuarioFK1 | int(11) | NO   | MUL | NULL    |                 |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.02 sec)
```

## 3.3 Tabla pedidosLibros

```
CREATE TABLE `pedidoslibros` (
  `idPedidoLibro` int(11) NOT NULL,
  `idLibroFK2` int(11) NOT NULL,
  `idPedidoFK3` int(11) NOT NULL,
  `cantidadPedidoLibro` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_spanish2_ci;
```

DESCRIBE pedidosLibros;

```
mysql> DESCRIBE pedidoslibros;
+-----+-----+-----+-----+-----+-----+
| Field              | Type   | Null | Key | Default | Extra           |
+-----+-----+-----+-----+-----+-----+
| idPedidoLibro      | int(11) | NO   | PRI | NULL    | auto_increment |
| idLibroFK2         | int(11) | NO   | MUL | NULL    |                 |
| idPedidoFK3        | int(11) | NO   | MUL | NULL    |                 |
| cantidadPedidoLibro | int(11) | YES  |     | NULL    |                 |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.02 sec)
```


## 3.4 Tabla usuarios

```
CREATE TABLE `usuarios` (  
  `idUsuario` int(11) NOT NULL,  
  `nombreUsuario` varchar(16) COLLATE utf8_spanish2_ci DEFAULT NULL,  
  `claveUsuario` varchar(41) COLLATE utf8_spanish2_ci DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_spanish2_ci;
```

DESCRIBE usuarios;

```
mysql> DESCRIBE usuarios;  
+-----+-----+-----+-----+-----+-----+  
| Field          | Type          | Null | Key | Default | Extra          |  
+-----+-----+-----+-----+-----+-----+  
| idUsuario      | int(11)       | NO   | PRI | NULL    | auto_increment |  
| nombreUsuario  | varchar(16)   | YES  |     | NULL    |                |  
| claveUsuario   | varchar(41)   | YES  |     | NULL    |                |  
+-----+-----+-----+-----+-----+-----+  
3 rows in set (0.02 sec)
```

```
INSERT INTO `usuarios` (`idUsuario`, `nombreUsuario`, `claveUsuario`) VALUES  
(1, 'usuario1', '122b738600a0f74f7c331c0ef59bc34c'),  
(2, 'usuario2', '2fb6c8d2f3842a5ceaa9bf320e649ff0');
```

Result Grid			
Filter Rows: <input type="text"/>			
Edit: 			
	idUsuario	nombreUsuario	claveUsuario
▶	1	usuario1	122b738600a0f74f7c331c0ef59bc34c
	2	usuario2	2fb6c8d2f3842a5ceaa9bf320e649ff0
*	NULL	NULL	NULL

### 4 – Código de la Parte del Servidor

#### 4.1 LoginServlet.java

```
package es.studium.LoginServlet;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import javax.sql.DataSource;

/**
 * Servlet implementation class LoginServlet
```



```
*/

@WebServlet(

    name = "LoginServlet",

    urlPatterns = {"/login"})

public class LoginServlet extends HttpServlet

{

    private static final long serialVersionUID = 1L;

    // Pool de conexiones a la base de datos

    private DataSource pool;

    /**

     * @see HttpServlet#HttpServlet()

     */

    public LoginServlet()

    {

        super();

    }

    /**

     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

     */

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws

        ServletException, IOException

    {

        response.setContentType("text/html;charset=UTF-8");

        PrintWriter out = response.getWriter();

        Connection conn = null;
```

```
Statement stmt = null;

try
{
    out.println("<html>");

    out.println("<head>");

    out.println("<link rel=\"stylesheet\"
href=\"https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.m
in.css\" \r\n\" + \"integrity=\"sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZ
w1T\" crossorigin=\"anonymous\">");

    out.println("<title>Login</title>");

    out.println("</head>");

    out.println("<body>");

    out.println("<h2>Login</h2>");

    // Obtener una conexión del pool

    conn = pool.getConnection();

    stmt = conn.createStatement();

    // Recuperar los parámetros usuario y password de la petición request

    String usuario = request.getParameter("usuario");

    String password = request.getParameter("password");

    //Validar los parámetros de la petición request

    if(usuario.length()==0)

    {

        out.println("<h3>Debes introducir tu usuario</h3>");

    }

    else if(password.length()==0)
```

```
{  
  
    out.println("<h3>Debes introducir tu contraseña</h3>");  
  
}  
  
else  
  
{  
  
    //Verificar que existe el usuario y su correspondiente clave  
  
    StringBuilder sqlStr = new StringBuilder();  
  
    sqlStr.append("SELECT * FROM usuarios WHERE ");  
  
    sqlStr.append("STRCMP(usuarios.nombreUsuario,  
    "").append(usuario).append(") = 0");  
  
    sqlStr.append(" AND STRCMP(usuarios.claveUsuario,  
    MD5(").append(password).append(")) = 0");  
  
    out.println("<p>" + sqlStr.toString() + "</p>");  
  
    ResultSet rset = stmt.executeQuery(sqlStr.toString());  
  
    if(!rset.next())  
  
    {  
  
        //Si el resultset no está vacío  
  
        out.println("<h3>Nombre de usuario o contraseña  
        incorrectos</h3>");  
  
        out.println("<p><a href='index.html'>Volver a  
        Login</a></p>");  
  
    }  
  
    else  
  
    {  
  
        //Si los datos introducidos son correctos  
  
        //Crear una sesión nueva y guardar el usuario como variable  
        de sesión
```

```
//Primero, invalida la sesión si ya existe

HttpSession session = request.getSession(false);

if(session != null)

{

    session.invalidate();

}

session = request.getSession(true);

synchronized(session)

{

    session.setAttribute("usuario", usuario);

}

out.println("<p>Hola, " + usuario + "!</p>");

out.println("<a href='logout'><button class='btn btn-
primary'>Logout</button></a>");

out.println("<a href='Libros.jsp'><button class='btn btn-
primary'>Libros</button></a>");

out.println("<a href='Pedidos.jsp'><button class='btn btn-
primary'>Pedidos</button></a>");

}

}

out.println("</body>");

out.println("</html>");

}

catch(SQLException ex)

{

    out.println("<p>Servicio no disponible</p>");

}
```

```
        out.println("</body>");

        out.println("</html>");

    }

    finally

    {

        // Cerramos objetos

        out.close();

        try

        {

            if(stmt != null)

            {

                stmt.close();

            }

            if(conn != null)

            {

                // Esto devolvería la conexión al pool

                conn.close();

            }

        }

        catch(SQLException ex){}

    }

    response.getWriter().append("Served at: ").append(request.getContextPath());

}

/**

 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
```

```
*/

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException
{
    // TODO Auto-generated method stub
    doGet(request, response);
}

public void init(ServletConfig config) throws ServletException
{
    try
    {
        // Crea un conector para poder luego buscar el recurso DataSource
        InitialContext ctx = new InitialContext();

        // Busca el recurso DataSource en el contexto
        pool = (DataSource)ctx.lookup("java:comp/env/jdbc/mysql_tiadalibros");

        if(pool == null)
        {
            throw new ServletException("DataSource desconocida
            'mysql_tiadalibros'");
        }
    }
    catch(NamingException ex){}
}
}
```



### 4.2 LogoutServlet.java

```
package es.studium.LoginServlet;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

/**
 * Servlet implementation class LogoutServlet
 */
@WebServlet(
    name = "LogoutServlet",
    urlPatterns = {"/logout"})
public class LogoutServlet extends HttpServlet
{
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
}
```

```
public LogoutServlet()
{
    super();

    // TODO Auto-generated constructor stub
}

/**
 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
 */
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException
{
    response.setContentType("text/html;charset=UTF-8");

    PrintWriter out = response.getWriter();

    try
    {
        out.println("<html>");

        out.println("<head>");

        out.println("<link rel=\"stylesheet\"
href=\"https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.m
in.css\" \r\n\" + \"integrity=\"sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZ
w1T\" crossorigin=\"anonymous\">");

        out.println("<title>Logout</title>");

        out.println("</head>");

        out.println("<body>");

        out.println("<h2>Logout</h2>");
    }
}
```

```
        HttpSession session = request.getSession(false);

        if(session == null)
        {
            out.println("<h3>No has iniciado sesión</h3>");
        }
        else
        {
            session.invalidate();

            out.println("<p>Adiós</p>");

            out.println("<a href='index.html'><button class='btn btn-
            primary'>Login</button></a>");
        }

        out.println("</body>");
        out.println("</html>");
    }

    finally
    {
        // Cerramos objetos

        out.close();
    }

    response.getWriter().append("Served at: ").append(request.getContextPath());
}

/**
 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
 */
```

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException
{
    // TODO Auto-generated method stub
    doGet(request, response);
}
}
```

### 4.3 Index.html

```
<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

<title>Login</title>

</head>

<body>

    <h2>Acceso</h2>

    <form method="get" action="login">

        <table>

            <tr>
```

```
        <td>Introduce tu nombre de usuario:</td>

        <td><input type="text" name="usuario" /></td>

    </tr>

    <tr>

        <td>Introduce tu clave:</td>

        <td><input type="password" name="password" /></td>

    </tr>

</table>

<br /> <input class="btn btn-primary" type="submit" name="Acceder" /> <input
class="btn btn-primary" type="reset" name="Cancelar" />

</form>

</body>

</html>
```

### 4.4 Libros.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//ES"
"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF8">

<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
```

```
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

<title>Consulta de Libros</title>

</head>

<body>

    <%@ page import="java.sql.*"%>

    <%

        // Paso 1: Cargamos el driver

        Class.forName("com.mysql.jdbc.Driver");

        // Paso 2: Conectarse a la base de datos utilizando un objeto de la clase
        Connection

        String userName = "servletUser2";

        String password = "Studium2018;";

        // URL de la base de datos

        String url = "jdbc:mysql://localhost:3306/tiendalibros";

        Connection conn = DriverManager.getConnection(url, userName, password);

        // Paso 3: Crear las sentencias SQL utilizando objetos de la clase Statement

        Statement stmt = conn.createStatement();

        // Paso 4: Ejecutar las sentencias

        String sqlStr = "SELECT * FROM libros ";

        sqlStr += "ORDER by tituloLibro DESC";

        System.out.println(sqlStr);

        ResultSet rs = stmt.executeQuery(sqlStr);

    %>

    <hr />
```



```
<table class="table table-dark">

    <tr>

        <th>Autor</th>

        <th>Título</th>

        <th>Precio</th>

        <th>Cantidad</th>

    </tr>

    <%

        // Paso 5: Recoger los resultados y procesarlos

        while (rs.next())

            {

    %>

    <tr>

        <td><%=rs.getString("autorLibro")%></td>

        <td><%=rs.getString("tituloLibro")%></td>

        <td><%=rs.getString("precioLibro")%></td>

        <td><%=rs.getString("cantidadLibro")%></td>

    </tr>

    <%

        }

    %>

</table>

<%

    // Cierre de recursos

    rs.close();
```

```
        stmt.close();

        conn.close();

    %>

    <a href="login?usuario=usuario1&password=usuario1&Acceder=Enviar+consulta"><button
    class="btn btn-primary">Volver</button></a>

</body>

</html>
```

### 4.5 Pedidos.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//ES"
"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF8">

<title>Consulta de Libros</title>

<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

</head>

<body>

    <%@ page import="java.sql.*"%>

    <%

        // Paso 1: Cargamos el driver
```

```
Class.forName("com.mysql.jdbc.Driver");

// Paso 2: Conectarse a la base de datos utilizando un objeto de la clase
Connection

String userName = "servletUser2";

String password = "Studium2018;";

// URL de la base de datos

String url = "jdbc:mysql://localhost:3306/tiendalibros";

Connection conn = DriverManager.getConnection(url, userName, password);

// Paso 3: Crear las sentencias SQL utilizando objetos de la clase Statement

Statement stmt = conn.createStatement();

// Paso 4: Ejecutar las sentencias

String sqlStr = "SELECT * FROM pedidos ";

sqlStr += "ORDER by idPedido ASC";

System.out.println(sqlStr);

ResultSet rs = stmt.executeQuery(sqlStr);

%>

<hr />

<table class="table table-dark">

    <tr>

        <th>ID Pedido</th>

        <th>Fecha de Pedido</th>

        <th>ID Usuario</th>

        <th>Estado del Pedido</th>

        <th>Procesar Pedido</th>

    </tr>
```

```
<%

    // Paso 5: Recoger los resultados y procesarlos

    while (rs.next())
    {
%>

<tr>

    <td><%=rs.getString("idPedido")%></td>

    <td><%=rs.getString("fechaPedido")%></td>

    <td><%=rs.getString("idUsuarioFK1")%></td>

    <td><%=rs.getString("estadoPedido")%></td>

    <td>

        <form action="#" method="post">

            <input type="hidden" name="idPedido"
            value="<%=rs.getString("idPedido")%>">

            <input class="btn btn-primary" type="submit"
            value="Procesar" />

        </form>

    </td>

</tr>

<%

    }

%>

</table>

<%

    String[] idPedido = request.getParameterValues("idPedido");

    if (idPedido != null)
```

```
{

    String estadoPedido = "UPDATE pedidos SET estadoPedido = 1 WHERE
    idPedido = " + idPedido[0] + ";";

    System.out.println(estadoPedido);

    try
    {

        stmt.executeUpdate(estadoPedido);

    } catch (SQLException e)

    {

        e.printStackTrace();

    }

    %><h3><%=idPedido[0]%></h3><%=

}

// Cierre de recursos

    rs.close();

    stmt.close();

    conn.close();%>

<a href="login?usuario=usuario1&password=usuario1&Acceder=Enviar+consulta"><button
class="btn btn-primary">Volver</button></a>

</body>

</html>
```

### 5 – Código de la Parte del Servidor

#### 5.1 ElementoPedido.java

```
package es.studium.LibreriaMVC;

/**
 *
 * @author Jorge
 * ElementoPedido
 * Representa un elemento del pedido
 * Incluye identificador del libro y cantidad
 *
 */
public class ElementoPedido
{
    private int idLibro;
    private int cantidad;

    public ElementoPedido(int idLibro, int cantidad)
    {
        this.idLibro = idLibro;
        this.cantidad = cantidad;
    }

    public int getIdLibro()
    {

```



```
        return idLibro;
    }

    public void setIdLibro(int idLibro)
    {
        this.idLibro = idLibro;
    }

    public int getCantidad()
    {
        return cantidad;
    }

    public void setCantidad(int cantidad)
    {
        this.cantidad = cantidad;
    }

    public String getAutor()
    {
        return LibrosMVC.getAutor(idLibro);
    }

    public String getTitulo()
    {
        return LibrosMVC.getTitulo(idLibro);
    }

    public String getPrecio()
    {
        return LibrosMVC.getPrecio(idLibro);
    }
}
```

```
    }  
}
```

### 5.2 LibrosMVC.java

```
package es.studium.LibreriaMVC;  
  
import java.sql.Connection;  
  
import java.sql.DriverManager;  
  
import java.sql.ResultSet;  
  
import java.sql.Statement;  
  
/**  
 *  
 * @author Jorge  
 * LibrosMVC  
 * Encapsula la comunicación con la base de datos  
 * Almacena títulos, autores y precios en tres tablas  
 *  
 */  
  
public class LibrosMVC  
{  
  
    private static final int MAX_SIZE = 4;  
  
    private static String[] titulos = new String[MAX_SIZE];  
  
    private static String[] autores = new String[MAX_SIZE];  
  
    private static String[] precios = new String[MAX_SIZE];  
  
    public static void cargarDatos()
```

```
{  
  
    // Creamos objetos para la conexión  
  
    Connection conn = null;  
  
    Statement stmt = null;  
  
    try  
    {  
  
        // Paso 1: Cargamos el driver  
  
        Class.forName("com.mysql.jdbc.Driver");  
  
        // Paso 2: Conectarse a la base de datos utilizando un objeto de la clase  
        Connection  
  
        String userName = "servletUser2";  
  
        String password = "Stadium2018";  
  
        // URL de la base de datos  
  
        String url = "jdbc:mysql://localhost:3306/tiendalibros?useSSL=false";  
  
        conn = DriverManager.getConnection(url, userName, password);  
  
        // Paso 3: Crear las sentencias SQL utilizando objetos de la clase Statement  
  
        stmt = conn.createStatement();  
  
        // Paso 4: Ejecutar las sentencias  
  
        String sqlStr = "SELECT * FROM libros";  
  
        ResultSet rs = stmt.executeQuery(sqlStr);  
  
        // Paso 5: Recoger los resultados y procesarlos  
  
        int cont = 0;  
  
        while(rs.next())  
        {  
  
            autores[cont] = rs.getString("autorLibro");  
  
        }  
    }  
}
```

```
        titulos[cont] = rs.getString("tituloLibro");

        precios[cont] = rs.getString("precioLibro");

        cont++;

    }

}

catch(Exception ex)

{

    ex.printStackTrace();

}

finally

{

    try

    {

        // Cerramos el resto de recursos

        if(stmt != null)

        {

            stmt.close();

        }

        if(conn != null)

        {

            conn.close();

        }

    }

    catch(Exception ex)

    {
```

```
                ex.printStackTrace();
            }
        }
    }

    /**
     * Devuelve el número de libros obtenidos
     */
    public static int tamano()
    {
        return titulos.length;
    }

    /**
     * Devuelve el título del libro identificado con idLibro
     */
    public static String getTitulo(int idLibro)
    {
        return titulos[idLibro];
    }

    /**
     * Devuelve el autor del libro identificado con idLibro
     */
    public static String getAutor(int idLibro)
    {
        return autores[idLibro];
    }
}
```

```
/**
 * Devuelve el precio del libro identificado con idLibro
 */
public static String getPrecio(int idLibro)
{
    return precios[idLibro];
}
}
```

### 5.3 ServletControlador.java

```
package es.studium.LibreriaMVC;

import java.io.IOException;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.time.LocalDate;

import java.util.ArrayList;

import java.util.Formatter;

import java.util.Iterator;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletConfig;
```



```
import javax.servlet.ServletContext;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

/**
 * Servlet implementation class ServletControlador
 */
@WebServlet("/ServletControlador")
public class ServletControlador extends HttpServlet
{
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public ServletControlador()
    {
        super();
        // TODO Auto-generated constructor stub
    }

    public void init(ServletConfig conf) throws ServletException
    {
```

```
        super.init(conf);

        LibrosMVC.cargarDatos();
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
        ServletException, IOException
    {
        doPost(request, response);
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
        ServletException, IOException
    {
        // Recupera la sesión actual o crea una nueva si no existe
        HttpSession session = request.getSession(true);

        // Recupera el carrito de la sesión actual
        ArrayList<ElementoPedido> elCarrito = (ArrayList<ElementoPedido>)
            session.getAttribute("carrito");

        // Determina a qué página jsp se dirigirá
        String nextPage = "";

        String todo = request.getParameter("todo");
```

```
if(todo==null)
{
    // Primer acceso, redirigir a order.jsp
    nextPage = "/order.jsp";
}
else if(todo.equals("add"))
{
    // Mandado por order.jsp con los parámetros idLibro y cantidad
    // Creamos un elementoPedido y lo añadimos al carrito
    ElementoPedido nuevoElementoPedido = new ElementoPedido(
        Integer.parseInt(request.getParameter("idLibro")),
        Integer.parseInt(request.getParameter("cantidad")));
    if(elCarrito==null)
    {
        // El carrito está vacío
        elCarrito = new ArrayList<>();
        elCarrito.add(nuevoElementoPedido);
        // Enlazar el carrito con la sesión
        session.setAttribute("carrito", elCarrito);
    }
    else
    {
        // Comprueba si el libro está ya en el carrito
        // Si lo está, actualizamos la cantidad
        // Si no está, lo añadimos
    }
}
```

```
boolean encontrado = false;

Iterator<ElementoPedido> iter = elCarrito.iterator();

while(!encontrado && iter.hasNext())
{
    ElementoPedido unElementoPedido =
        (ElementoPedido) iter.next();

    if(unElementoPedido.getIdLibro() ==
        nuevoElementoPedido.getIdLibro())
    {

        unElementoPedido.setCantidad(unElementoPedido.getCantidad()
            + nuevoElementoPedido.getCantidad());

        encontrado = true;

    }
}

if(!encontrado)
{
    // Lo añade al carrito

    elCarrito.add(nuevoElementoPedido);

}

// Volvemos a order.jsp para seguir con la compra
nextPage = "/order.jsp";

}

else if(todo.equals("remove"))
{
```

```
// Enviado por order.jsp con el parámetro indiceElemento

// Borra el elemento indiceElemento del carrito

int indiceCarrito = Integer.parseInt(request.getParameter("indiceElemento"));

elCarrito.remove(indiceCarrito);

// Vuelve a order.jsp para seguir con la compra

nextPage = "/order.jsp";

}

else if (todo.equals("checkout"))

{

    // Enviado por order.jsp

    // Calcula el precio total de todos los elementos del carrito

    float precioTotal = 0;

    int cantidadTotalOrdenada = 0;

    // Alta en pedidos

    Connection conn = null;

    Statement stmt = null;

    int idPedidoFK = 0;

    try

    {

        Class.forName("com.mysql.jdbc.Driver");

        String userName = "servletUser2";

        String password = "Studium2018;";

        String url = "jdbc:mysql://localhost:3306/TiendaLibros?useSSL=true";

        conn = DriverManager.getConnection(url, userName, password);

        stmt = conn.createStatement();
```

```
        LocalDate fecha = LocalDate.now();

        String sqlStr = "INSERT INTO pedidos (idPedido, fechaPedido,
        idUsuarioFK1, estadoPedido) VALUES(null,'" + fecha + "',1,0)";

        stmt.executeUpdate(sqlStr);

        // Sacar el id recién creado idPedido-->idPedidoFK

        String sacarIdPedido = "SELECT MAX(idPedido) AS idPedidoFK FROM
        pedidos;";

        ResultSet rs = stmt.executeQuery(sacarIdPedido);

        rs.next();

        idPedidoFK = rs.getInt("idPedidoFK");
    }

    catch (ClassNotFoundException | SQLException e)
    {

        e.printStackTrace();
    }

    // Por cada elemento del carrito, hacer un INSERT en pedidosLibros
    for(ElementoPedido item: elCarrito)
    {

        float precio = Float.parseFloat(item.getPrecio());

        int cantidadOrdenada = item.getCantidad();

        int idLibroFK2 = item.getIdLibro();

        idLibroFK2++;

        System.out.println(idLibroFK2);

        precioTotal += precio * cantidadOrdenada;

        cantidadTotalOrdenada += cantidadOrdenada;
    }
}
```

```
String insertPedidosLibros = "INSERT INTO pedidosLibros
(idPedidoLibro, idLibroFK2, idPedidoFK3, cantidadPedidoLibro)
VALUES(null,"+idLibroFK2+",
+idPedidoFK+", "+cantidadOrdenada+");";

try
{
    stmt.executeUpdate(insertPedidosLibros);
} catch (SQLException e)
{
    e.printStackTrace();
}

}

// Da formato al precio con dos decimales
StringBuilder sb = new StringBuilder();
Formatter formatter = new Formatter(sb);
formatter.format("%.2f", precioTotal);
formatter.close();

// Coloca el precioTotal y la cantidadtotal en el request
request.setAttribute("precioTotal", sb.toString());
request.setAttribute("cantidadTotal", cantidadTotalOrdenada+ "");

//Redirige a checkout.jsp
nextPage = "/checkout.jsp";

}

ServletContext servletContext = getServletContext();
```

```
        RequestDispatcher requestDispatcher =  
            servletContext.getRequestDispatcher(nextPage);  
        requestDispatcher.forward(request, response);  
    }  
}
```

### 5.4 checkout.jsp

```
<!-- Página de confirmación del pedido --%>  
  
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  
  
<%@ page session="true" import="java.util.*, es.studium.LibreriaMVC.*"%>  
  
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//ES"  
"http://www.w3.org/TR/html4/loose.dtd">  
  
<html>  
  
<head>  
  
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  
  
<title>Confirmación</title>  
  
</head>  
  
<body>  
  
    <h1>Librería MVC: Confirmación</h1>  
  
    <hr />  
  
    <br />  
  
    <p>  
  
        <strong>Vas a comprar los siguientes libros:</strong>  
  
    </p>  
  
    <table border="1" cellspacing="0" cellpadding="5">
```



```
<tr>

    <th>Título</th>

    <th>Autor</th>

    <th>Precio</th>

    <th>Cantidad</th>

</tr>

<%

    // Muestra los elementos del carrito

    List<ElementoPedido> cesta = (List<ElementoPedido>)
    session.getAttribute("carrito");

    for (ElementoPedido item : cesta)

    {

%>

<tr>

    <td><%=item.getTitulo()%></td>

    <td><%=item.getAutor()%></td>

    <td align="right"><%=item.getPrecio()%> €</td>

    <td align="right"><%=item.getCantidad()%></td>

</tr>

<%

    }

    session.invalidate();

%>

<tr>

    <th align="right" colspan="2">Total</th>
```

```
<th align="right"><%=request.getAttribute("precioTotal")%></th>

<th align="right"><%=request.getAttribute("cantidadTotal")%></th>

</tr>

</table>

<br />

<a href="shopping">Pulsa aquí para comprar más libros</a>

</body>

</html>
```

### 5.5 order.jsp

```
<%-- Página de órdenes de pedido --%>

<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

<%@ page session="true" import="java.util.*, es.studium.LibreriaMVC.*"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//ES"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Pedido</title>

</head>

<body>

<h1>Librería MVC</h1>

<hr />

<br />
```

```
<p>

    <strong>Elegir un libro y la cantidad:</strong>

</p>

<form name="AgregarForm" action="shopping" method="POST">

    <input type="hidden" name="todo" value="add"> Título: <select

        name="idLibro">

            <%

                // Scriptlet 1: Carga los libros en el SELECT

                for (int i = 0; i < LibrosMVC.tamano(); i++)

                {

                    out.println("<option value='" + i + "'>");

                    out.println(LibrosMVC.getTitulo(i) + " | " +

                        LibrosMVC.getAutor(i) + " | " + LibrosMVC.getPrecio(i));

                    out.println("</option>");

                }

            %>

        </select> Cantidad: <input type="text" name="cantidad" size="10" value="1">

        <input type="submit" value="Añadir a la cesta">

    </form>

    <hr />

    <br />

    <%

        // Scriptlet 2: Chequea el contenido de la cesta

        List<ElementoPedido> cesta = (List<ElementoPedido>)

        session.getAttribute("carrito");

        if (cesta != null && cesta.size() > 0)
```

```
{
%>
<p>
    <strong>Tu cesta contiene:</strong>
</p>
<table border="1" cellspacing="0" cellpadding="5">
    <tr>
        <th>Título</th>
        <th>Autor</th>
        <th>Precio</th>
        <th>Cantidad</th>
        <th>&nbsp;</th>
    </tr>
    <%
        // Scriptlet 3: Muestra los libros del carrito
        for (int i = 0; i < cesta.size(); i++)
        {
            ElementoPedido elementoPedido = cesta.get(i);
%>
    <tr>
        <form name="borrarForm" action="shopping" method="POST">
            <input type="hidden" name="todo" value="remove"> <input
                type="hidden" name="indiceElemento" value="<%=i%>">
            <td><%=elementoPedido.getTitulo()%></td>
            <td><%=elementoPedido.getAutor()%></td>
```

```
<td align="right"><%=elementoPedido.getPrecio()%> €</td>

<td align="right"><%=elementoPedido.getCantidad()%></td>

<td><input type="submit" value="Eliminar de la cesta"></td>

</form>

</tr>

<%

    }

%>

</table>

<br />

<form name="checkoutForm" action="shopping" method="POST">

    <input type="hidden" name="todo" value="checkout"> <input type="submit"
    value="Confirmar compra">

</form>

<%

}

%>

</body>

</html>
```

## 6 – Resultados

### 6.1 Cliente

**Librería MVC**

Elegir un libro y la cantidad:

Título:  Cantidad:

Tu cesta contiene:

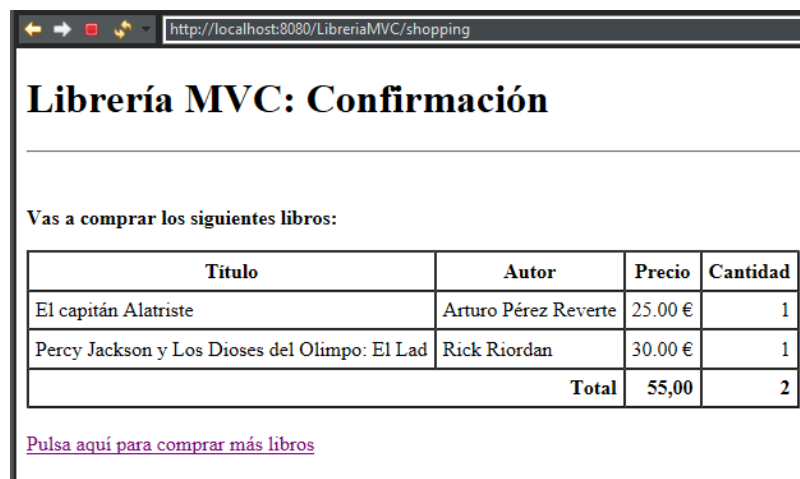
Título	Autor	Precio	Cantidad	
El capitán Alatriste	Arturo Pérez Reverte	25.00 €	1	<input type="button" value="Eliminar de la cesta"/>
Percy Jackson y Los Dioses del Olimpo: El Lad	Rick Riordan	30.00 €	1	<input type="button" value="Eliminar de la cesta"/>

Esta es la pantalla de compra del carrito. Cuando pulsas el botón “Confirmar compra”, hace insert en las tablas “pedidos” y “pedidosLibros” de nuestra base de datos.

	idPedido	fechaPedido	idUsuarioFK1	estadoPedido
▶	1	2019-03-07	1	1
	2	2019-03-07	1	1
	3	2019-03-07	1	0
	4	2019-03-07	1	0
	5	2019-03-07	1	0
	6	2019-03-07	1	0
	7	2019-03-07	1	0
	8	2019-03-07	1	0
	9	2019-03-07	1	0
	10	2019-03-07	1	0
	11	2019-03-12	1	0
*	NULL	NULL	NULL	NULL

	idPedidoLibro	idLibroFK2	idPedidoFK3	cantidadPedidoLibro
▶	1	1	10	1
	2	2	10	2
	3	1	11	2
	4	3	11	1
*	NULL	NULL	NULL	NULL

1



**Librería MVC: Confirmación**

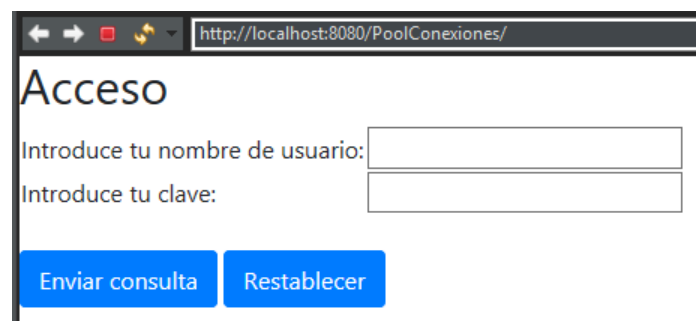
Vas a comprar los siguientes libros:

Título	Autor	Precio	Cantidad
El capitán Alatriste	Arturo Pérez Reverte	25.00 €	1
Percy Jackson y Los Dioses del Olimpo: El Lad	Rick Riordan	30.00 €	1
Total		55,00	2

[Pulsa aquí para comprar más libros](#)

Y esta sería la ventana después de haber presionado el botón “Confirmar compra”

## 6.2 Servidor

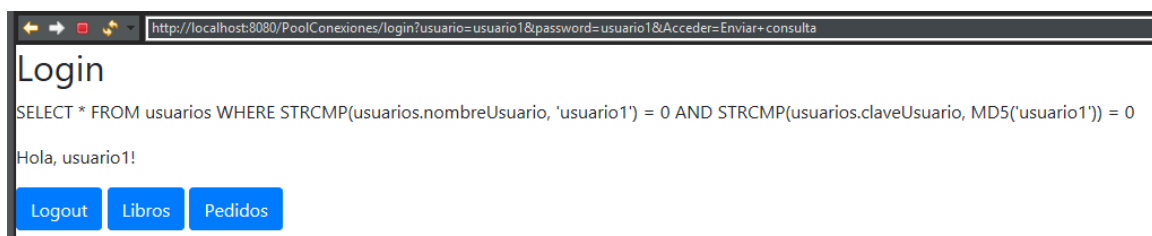


**Acceso**

Introduce tu nombre de usuario:

Introduce tu clave:

Esta es la pantalla de login para acceder a la parte del servidor del carrito.

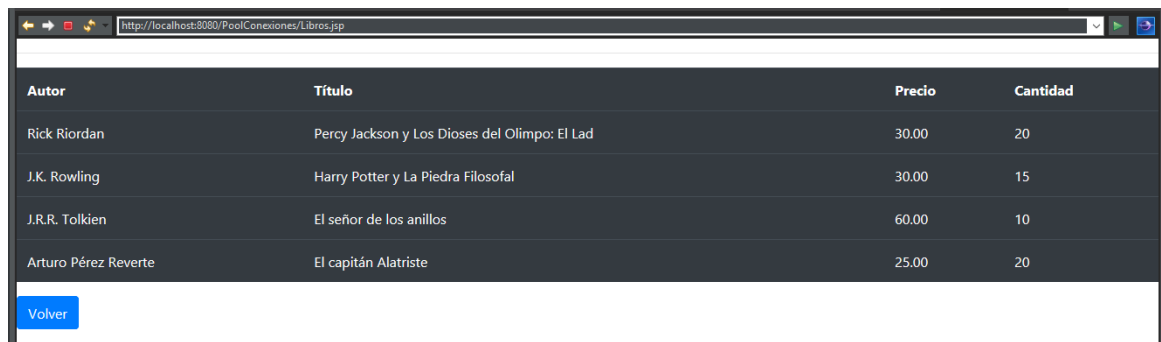


**Login**

SELECT \* FROM usuarios WHERE STRCMP(usuarios.nombreUsuario, 'usuario1') = 0 AND STRCMP(usuarios.claveUsuario, MD5('usuario1')) = 0

Hola, usuario1!

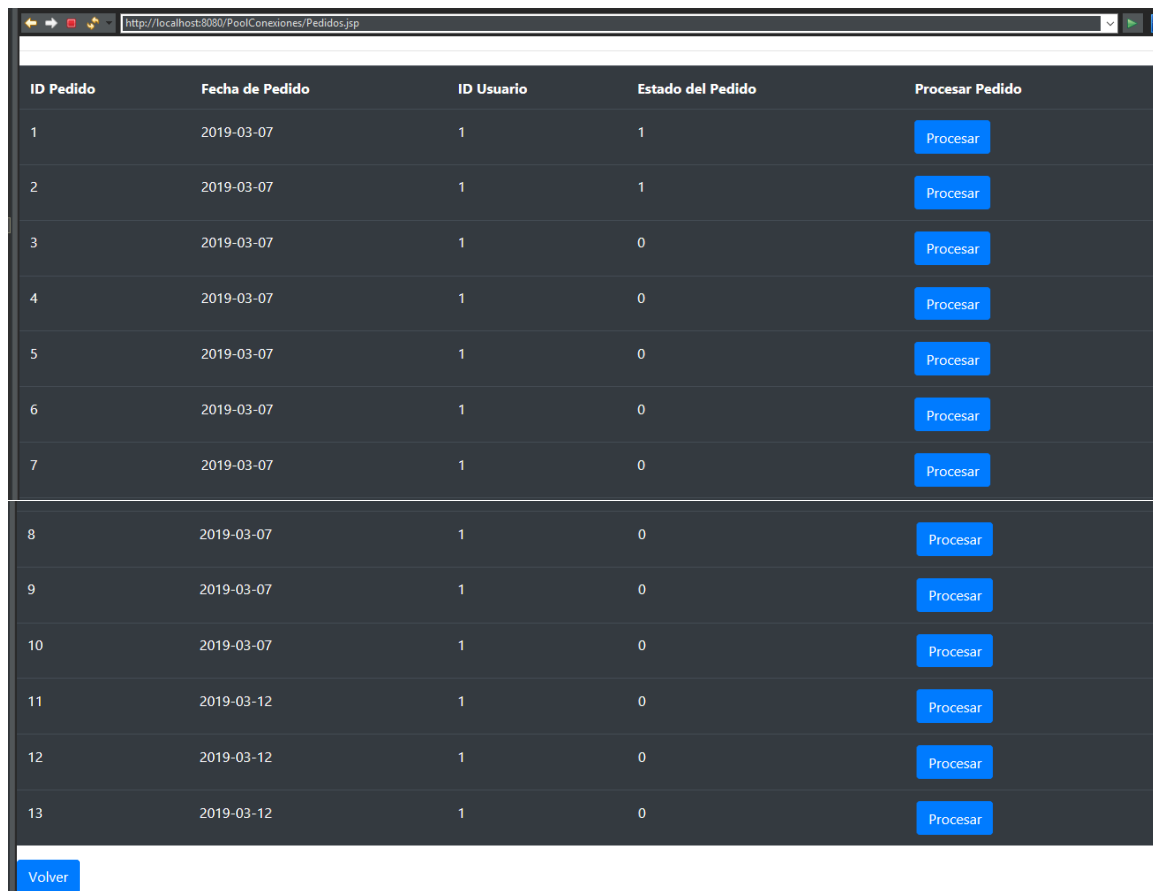
Esta es la pantalla una vez inicias sesión en la parte del servidor del carrito.



Autor	Título	Precio	Cantidad
Rick Riordan	Percy Jackson y Los Dioses del Olimpo: El Lad	30.00	20
J.K. Rowling	Harry Potter y La Piedra Filosofal	30.00	15
J.R.R. Tolkien	El señor de los anillos	60.00	10
Arturo Pérez Reverte	El capitán Alatriste	25.00	20

Volver

Pulse el botón “Libros” para ver un listado de los libros añadidos en la base de datos.



ID Pedido	Fecha de Pedido	ID Usuario	Estado del Pedido	Procesar Pedido
1	2019-03-07	1	1	Procesar
2	2019-03-07	1	1	Procesar
3	2019-03-07	1	0	Procesar
4	2019-03-07	1	0	Procesar
5	2019-03-07	1	0	Procesar
6	2019-03-07	1	0	Procesar
7	2019-03-07	1	0	Procesar
8	2019-03-07	1	0	Procesar
9	2019-03-07	1	0	Procesar
10	2019-03-07	1	0	Procesar
11	2019-03-12	1	0	Procesar
12	2019-03-12	1	0	Procesar
13	2019-03-12	1	0	Procesar

Volver

Pulse el botón “Pedidos” para ver un listado de los pedidos añadidos en la base de datos.  
Pulse el botón “Procesar” para cambiar el estado del pedido de 0 a 1.





Pulse el botón “Logout” para cerrar la sesión con la parte del servidor.

## 7 – Valoración Personal

En esta práctica hemos aprendido a realizar un carrito de la compra, tanto la parte del cliente como del servidor, en un servidor tomcat con servlets de java. También hemos aprendido a realizar archivos jsp, introduciendo java en un archivo html.

En cuanto a la dificultad de esta práctica, me ha resultado más complicada de lo normal, debido a que no se me han dado especialmente bien los servlets de java, aunque al término de la práctica he logrado entender mejor su funcionamiento.

## 8 – Referencias

### 8.1 Grupo Studium

Para la realización de esta práctica han sido necesarios los conocimientos adquiridos a través del temario “Tema 4: Aplicaciones J2EE” del módulo Despliegue de Aplicaciones Web, del ciclo formativo de grado superior de Desarrollo de Aplicaciones Web, realizado por el centro privado Grupo Studium, colgado en su plataforma online.