PRÁCTICA Nº 2

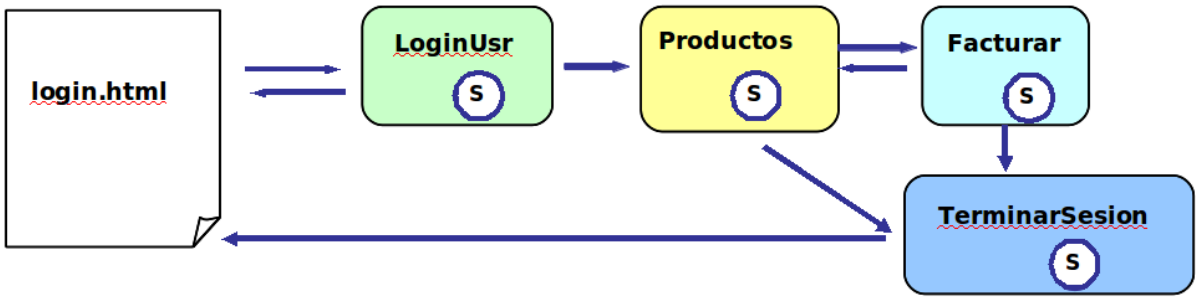
Temas

- Implementación de Servlets usando **sesiones.**
- Recolección de información (inputs) a través de páginas HTML para la implementación de un changuito de compras simple usando Servlets.

En esta práctica se desarrollará un pequeño sistema Web para realizar compras de golosinas que requiere de una autenticación para ingresar al mismo.

Cree un proyecto Web dinámico de nombre **Practica2**. Asigne como nombre de la aplicación Web: **compras**.

1. La aplicación Web **compras** debe permitir elegir golosinas desde una lista y comprarlas. La aplicación requiere autenticación. El diagrama de navegación de la aplicación es el siguiente:



Descripción de cada componente:

(Ubique las clases de los servlets en el paquete **misservlets.practica2)**

- login.html: es un formulario HTML con cuatro campos de texto: nombre de usuario, contraseña, apellido y nombres y dirección postal y, un botón *submit* que permite enviar los datos del formulario al servlet **LoginUsr**. La página debe contener un título. **Recuerde que este es el punto de entrada para ejecutar la aplicación.**

- LoginUsr: es un servlet que autentica al usuario en la aplicación, tiene la misma funcionalidad que el servlet **LoginUsr** del **ejercicio 4 de la práctica 1:** el método *init()* debe inicializar una variable de instancia del servlet, de tipo Hashtable llamada **logins**, con los parámetros de inicialización del **web.xml**. La tabla **logins** mantendrá como clave el nombre del usuario y como valor la contraseña.
Además de la funcionalidad anterior, este servlet debe:

- Si el usuario ingresado es **válido** -> se recupera la sesión del usuario, se almacenan todos sus datos y luego se redirecciona el requerimiento http al servlet **Productos**.
- Si el usuario ingresado es **inválido** -> el requerimiento http se redirecciona nuevamente a **login.html** para permitirle al usuario corregir los datos ingresados.

Ayuda: tenga en cuenta que el método *sendRedirect(URL)* de la interface *HttpServletResponse* redirecciona la respuesta a la URL pasada como parámetro.
Por ejemplo: **response.sendRedirect("/compras/productos");**

- Productos: es un servlet que dinámicamente arma un formulario HTML, que incluye una tabla HTML con tres columnas: nombres de las golosinas (no editable), precio unitario (no editable) y un campo de texto en donde el usuario ingresará la cantidad a comprar. Finalmente, el botón *submit* envía el requerimiento al servlet **Facturar**.
Los datos de las golosinas, se cargan en memoria, cuando se inicializa el servlet (en el método **init()** del servlet) a partir de parámetros de inicialización que se leen del archivo **web.xml**.
En el **web.xml**, en la declaración del servlet **Productos**, podría declarar por ejemplo los siguientes productos (nombre, valor):

param-name	param-value
cantidadTotal	5
golo1	**chupetin**
pu1	**2**
golo2	**chocolate**
pu2	**3**
.....	.....
.....	.....
golo5	**chicle**
pu5	**1**

Permite determinar la cantidad total de golosinas a cargar. También podría recorrer todos los parámetros y considerar solo los **golox.**

Otra posibilidad podría ser guardar en el **web.xml**:
golo1 chupetin,2
golo2 chocolate,3
. . .
y parsear los datos.

La página HTML resultante de ejecutar el servlet **Productos**, mostrará actualizada la cantidad de golosinas que el usuario va a comprar. Esto significa que inicialmente dichas cantidades tienen el valor cero, y que cada vez que se ejecuta el servlet, estas cantidades se actualizarán de acuerdo a los datos ingresados por el usuario, guardados en la sesión del usuario.
La página incluye un *link* cuyo texto es "Salir" que envía un requerimiento al servlet **TerminarSesion**.

- Facturar: es un servlet que arma dinámicamente una tabla HTML con tres columnas y tantas filas como golosinas compró el usuario. La primer columna contiene el nombre de la golosina, la segunda la cantidad comprada y la tercera el precio total. Todos son campos no editables. Al final de la tabla se muestra el **total general**.
En la sesión, se guardan los datos de las golosinas compradas (nombre, cantidad y precio unitario) por lo que los datos a mostrar se recuperan de la sesión.

Además, la página contiene dos *links:* uno con el texto "Seguir comprando" que al presionarlo ejecuta el servlet **Productos**, para que el usuario pueda seguir comprando (agregar otras golosinas, modificar la cantidad comprada, etc) y, otro con el texto "Salir" que envía un requerimiento al servlet **TerminarSesion**.

- TerminarSesion: es un servlet que invalida la sesión del usuario, pues el usuario ya terminó de comprar y redirecciona el requerimiento a la página **login.html**. No tiene pantalla asociada.

2. Configure el navegador para que no acepte cookies. ¿La aplicación web **compras** sigue funcionando?. En caso de no funcionar correctamente, escriba una solución.

3. Ejecute el servlet **Productos** directamente (botón derecho sobre Productos -> Run On Server…). ¿Es correcto que un usuario ingrese a la aplicación desde cualquier componente, sin previa autenticación?. ¿Cómo haría para rechazarlo?.