Práctica 2. Programación Web dinámica de un sistema de venta de DVDs

Versión 1.15 3/10/2018

Objetivos docentes

Tras la realización de la Práctica 2, el estudiante debe ser capaz de:

- Comprender el papel de los lenguajes de script en la interfaz de usuario para sistemas Web.
- Conocer los elementos fundamentales del lenguaje de servidor Web Python.
- Practicar programas sencillos con Python y Flask (micro-framework para desarrollar páginas web en Python): procesamiento de formularios y generación dinámica de la página, persistencia de sesión y mantenimiento de estado, etc.

Enunciado

En esta entrega continuamos con el trabajo iniciado en la anterior para el desarrollo de un portal de venta de DVDs.

- Las páginas se realizarán en Python haciendo uso de Flask y **como mínimo**, de la siguiente funcionalidad:
    - Mezcla de código HTML y Python.
    - Uso de diccionarios y arrays en Python.
    - Estructuras de programación (if, bucles, etc.).
    - Funciones de Python.
    - Persistencia de información de la sesión.
- Las páginas no accederán a los contenidos de una base de datos relacional, sino que los datos serán almacenados en ficheros JSON. En particular, se creará un fichero, `historial.json`, uno por cada usuario, que se actualizará automáticamente a partir de las compras que realice el usuario.
- La aplicación web se ejecutará en el servidor local Apache (*http://localhost/~`usuario`*), publicando los contenidos en el directorio */home/`usuario`/public_html*.

  Nota: El servidor Apache debe tener instalado el paquete **mod_wsgi**, que permite ejecutar cualquier aplicación web implementada en Python. Consultar el ejemplo en 'Instrucciones Plataforma'
- En esta entrega la información simulada se encontrará en un fichero JSON (`catalogo.json`). *Ayuda*: Un fichero JSON que se puede estructurar; por ejemplo:

```
{
   "peliculas": [
      {
         "titulo": "La guerra de las galaxias: Episodio 1-La amenaza fantasma",
         "poster": "imagenes/guerraepisodio1.jpg",
         "director": "George Lucas",
         "precio": 18.99,
         "categoria": "Aventura",
         "anno": 1999,
         "actores": [
           {
              "nombre": "Jake Lloyd",
              "personaje": "Anakin Skywalker",
              "foto": "imagenes/anakin.jpg"
           },
           {
              "nombre": "Ewan McGregor",
              "personaje": "Obi-Wan Kenobi",
              "foto": "imagenes/kenobi.jpg"
           }
         ]
      }
   ]
}
```


Funcionalidad

A continuación, se describen las modificaciones sobre las funcionalidades de la entrega anterior.

Entrada: Página Principal

La página muestra en la zona de contenidos una selección de DVDs disponibles, así como una caja de texto para realizar búsquedas y un desplegable para filtrar por categoría. La funcionalidad de búsqueda

y filtrado se implementarán mostrando una página en función de la categoría y el texto insertado por el usuario, usando diccionarios con datos precargados en Python.

Los datos para esta página se tomarán del fichero `catalogo.json`, que contendrá todas las películas disponibles. El filtrado de esos datos se debe realizar mediante Python.

Acceso de un Usuario

La página de acceso deberá comprobar que el usuario existe y validar la contraseña. Para ello, abrirá el archivo del usuario indicado (ver Registro), se compararán las contraseñas en md5 y si coinciden se dará acceso al sistema.

Se mantendrá, mediante sesión, la información del usuario al ir navegando por el sistema.

Se deberá implementar también la funcionalidad de salir o desconexión de un usuario.

La información del último usuario que inició sesión se mantendrá en una cookie, que se usará para precargar el campo 'usuario' (o similar) del formulario de acceso con este valor.

Registro de un Usuario

Dentro de la carpeta de la aplicación web (/home/alumnos/*`usuario`*/public_html) se creará una carpeta de nombre `usuarios` que contendrá una subcarpeta para cada usuario. Dentro de esta carpeta se creará un fichero (`datos.dat`) donde se guardará en variables el nombre del usuario, contraseña (cifrada en md5), correo electrónico, número de tarjeta de crédito y saldo. El saldo inicial que se asignará a cada usuario cuando se registre será determinado por un número entero aleatorio entre 0 y 100.

Ayuda: Cuidado con los permisos de la carpeta `usuarios`, ya que la carpeta debe ser accesible por el usuario bajo el que se ejecuta el servidor web Apache.

Esta carpeta propia de cada usuario también contendrá el historial de compras (fichero `historial.json`).

La página de registro deberá comprobar si el usuario ya existe. Para ello, comprobará si existe una subcarpeta con el nombre del usuario. Si es así, generará un error diciendo que el usuario ya existe. En caso contrario, continuará con el registro y se crearán la carpeta y el archivo correspondiente.

Esta página mostrará en la zona de contenido un formulario para pedir datos de un nuevo usuario. Se solicitará al menos la siguiente información: nombre de usuario, contraseña, confirmación de contraseña, email y datos de la tarjeta de crédito. También habrá un botón para confirmar. Si todo es correcto se creará una nueva carpeta dentro de la carpeta usuarios mencionada anteriormente.

Los datos introducidos en el formulario deberán ser validados en cliente (por ejemplo, formato correcto de dirección de correo, contraseña de 8 caracteres como mínimo, ausencia de caracteres prohibidos y blancos en el campo usuario, etc.). En los casos donde exista un control HTML5 específico con soporte en los dos navegadores principales (Chrome y Firefox), y ese control haga la verificación sintáctica, podrá ser usado. En otro caso, los datos se validarán mediante funciones de JavaScript.

Detalle de película

En esta página se mostrará la película seleccionada y permitirá añadirla al carrito de la compra.

La información de cada película se extraerá del fichero `catalogo.json`.

Carrito de la compra

Se deberá implementar la funcionalidad de carrito de la compra. Este carrito debe ser implementado utilizando sesiones en **Flask Session** y debe estar disponible en todas las páginas.

No será necesario que el usuario esté registrado para que el sistema le mantenga el carrito de compra, pero sí para poder confirmarlo: de no estar registrado y querer confirmar el carrito, se pedirá al usuario que se registre.

Se deberá implementar la funcionalidad de añadir un DVD al carrito (en la página de detalle de DVD), borrar un DVD del carrito, mostrar el contenido del carrito y finalizar la compra.

Finalizar compra de DVD

Finalizar la compra implica solicitar todos los productos incluidos en el carrito. Para ello el usuario debe estar registrado. Si no lo estuviera, se solicitará que lo haga al intentar finalizar la compra.

Antes de finalizar la compra se debe comprobar que el usuario tenga saldo. En caso de que no hubiese saldo disponible, ninguna compra del carrito se confirma. En caso de que sí disponga de saldo suficiente, el importe del carrito se descontará del saldo y éste se actualizará en el fichero de datos del usuario.

Una vez confirmadas y pagadas las compras, los datos de las mismas se añaden al final del fichero `historial.json`.

Mostrar historial de un usuario

Un usuario registrado podrá ver todas las compras realizadas hasta el momento, así como el saldo de su cuenta. También podrá incrementar el saldo disponible, que se actualizará en el fichero `datos.dat` con los datos del usuario.

Para esta práctica se implementará un fichero JSON, `historial.json`, que codificará los datos del historial. Estos datos se accederán y mostrarán utilizando funciones Python.

Otras funcionalidades

- Utilizar jQuery para desplegar/contraer detalles de los pedidos en el historial del usuario.
- Utilizar jQuery para implementar un medidor de la fortaleza de la contraseña, pudiendo incluir una barra de progreso.
- Utilizar AJAX para mostrar un banner con el número (generado aleatoriamente en el servidor llamando a un método de Python) de usuarios conectados al sistema de Venta de DVDs, que se debe actualizar cada 3 segundos.

Documentación a aportar

En esta segunda entrega deberéis aportar:

- Memoria con contenido similar a la entrega anterior: ficheros que componen la entrega, instrucciones de uso y detalles de implementación, capturas de pantalla como resultado de la entrega.
- La memoria deberá contener también referencias a todas las fuentes de código que hayan servido de inspiración para la elaboración de la práctica.
- Ficheros HTML, JSON, Python y JavaScript (y cualquier otro adicional) necesarios para la ejecución de la práctica.

Entrega

Esta práctica se dividirá en cuatro entregas, una cada semana. Las tres primeras entregas, si bien obligatorias, no serán evaluadas y servirán para ver el progreso del proyecto. El contenido de las entregas será:

- Primera semana: Se deberán entregar los ficheros correspondientes a la búsqueda, selección y detalle de películas.
- Segunda semana: Se deberán entregar los ficheros correspondientes al registro y acceso de un usuario.
- Tercera semana: Se deberá entregar los ficheros correspondientes al carrito.
- Entrega final: Se deberá entregar los ficheros correspondientes al historial, las funcionalidades adicionales, toda la documentación y ficheros requeridos.

La fecha y normas de entrega de las prácticas se encuentran en Moodle.

Bibliografía

[1] Aprendizaje interactivo de Python: http://www.diveintopython3.net/

[2] Uso de JSON en Python: https://docs.python.org/2/library/json.html

[3] Tipos de datos: https://docs.python.org/2/tutorial/datastructures.html

[4] Aprendizaje interactivo de JavaScript: https://www.w3schools.com/js/

[5] Flask: https://www.tutorialspoint.com/flask/index.htm

[6] Flask cookies: https://www.tutorialspoint.com/flask/flask_cookies.htm

[7] Sesiones con Flask-Session: https://pythonhosted.org/Flask-Session/

[8] WSGI: https://en.wikipedia.org/wiki/Web_Server_Gateway_Interface

[9] WSGI con Flask: http://flask.pocoo.org/docs/0.12/deploying/mod_wsgi/

[10] WSGI con virtual environment: https://modwsgi.readthedocs.io/en/develop/user-guides/virtual-environments.html