MEMORIA PRACTICA 2

SOFTWARE DISTRIBUIT

Carlos Navas
Oriol Riu

La aplicacio es basa en el Model-Vista-Controlador, on el model i controlador són les classes java (dades d'usuaris i productes) i (servletdispacher) respectivament i la vista implementada en els diferents jsp.

Controlador

Com s'ha comentat, com a controlador tenim una única classe (ServletDispatch) que rebrà totes les peticions que arriben pel port 8080 configurat pel Servidor Web tomcat. Aquest controla l'aplicació reencaminant als jsp corresponents, controlant les dades de l'aplicació així com la lectura de la base de dades.

El controlador al rebre un POST implementa el Post-redirect-Get per evitar enviar dades de forma visible per URL. El que fem es llegir els paràmetres rebuts per POST i els guardem a la sessió, així podran ser llegits i tractats sense exposar-los. Fem el redirect a la pàgina que correspon i entra per get de nou al controlador on ja podrem tractar les dades.

També tenim el controlador del Servei Web que rebrà les peticions que arriben pel port 8080 configurat pel Servidor Web tomcat. Hem hagut de modificar l'arxiu web.xml, perquè les peticions que apunten a /llibreria/API/* anessin a parar a un altre servlet, que es el que fa el servei web, donant el catàleg diferenciat per BOOK, VIDEO o AUDIO, o mostrant un sol ítem de diferents servidors amb la mateixa API.

Model

Per a fer el model de l'aplicació tenim dues classes base:

- Usuaris.java → aquesta classe permet emmagatzemar tota la informació corresponent a l'usuari, quan s'inicia el servlet, es a dir, es fa un mètode init, carreguem una array d'usuaris amb les dades que estan al fitxer usuaris.csv, aquest fitxer actua de "base de dades", ens servirà per donar permanència a les dades de l'usuari.

  En aquesta classe guardem la informació rellevant, el codi de l'usuari, el seu nom, el crèdit que li queda, i una llista de productes, que seran els productes que ha anat comprant, en aquesta llista de productes es guardaran els identificadors dels productes separats amb un ';'.

  Dades.java → En aquesta classe estem emmagatzemant tota la informació necessària dels productes que com a usuaris.java, la carreguem al començament del servlet, és a dir, a la funció ini, i guardem totes les dades d'aquest arxiu a una array global de productes, on es guardarà la informació rellevant a tots els productes, el seu identificador, nom, una petita descripció, la secció a la qual pertanyen, el seu preu.

Aquest identificador, és l'identificador que guardem a l'array d'usuaris per a saber quin producte té comprat l'usuari.

Les dades que hi ha a l'arxiu cataleg.csv, que es l'arxiu d'on traiem les dades, només són de lectura, mai reescrivim aquest arxiu.

La permanència de les dades és un tema important, tenim 2 mètodes per fer-ho possible, el primer mètode és el mètode destroy, sempre que per algun motiu es tanca el servlet, es truca a aquest mètode destroy que guardarà totes les dades de l'array d'Usuaris a l'arxiu usuaris.csv.
El segon és que cada vegada que nosaltres li donem al botó comprar directament o escrivim al fitxer d'usuaris.csv, això o fem per què si tanquem el navegador, no es guardaria res, i hem pensat que seria una bona opció.

Vista

A la vista trobem diferents jsp que son codi java amb html. Dels diferents jsp als que podem accedir, mantindrem la llista de la compra com a protegida. La resta de jsp són sense protegir.

Catàleg: Mostrem tots els articles disponibles emagatzemats en els diferents csv. No protegit.

Index: Aquesta es la pagina d'inici des de la que podem anar a qualsevol part de la web. No protegit.

Llista: Aquesta mostra els productes adquirits per l'usuari. Protegit, un cop intentem accedir a la llista, que està protegida, si no hem estat identificats, ens demanarà que ens loguegem i ens mostrarà la llista de l'usuari logat.

Error: Aquesta mostra l'error d'autentificació al fer login.

Header: Aquesta pàgina mostra la part de color negre a tota la resta de les pàgines, és la capçalera on va informació sobre l'usuari, si estàs logat o no i el crèdit que et queda, aquesta pàgina es truca a tota la resta de pàgines.

Login: Pagina que es mostra quan tenim que logarnos a la web, ja sigui perquè li donem a connectar, o perquè l'hi hem donat a comprat i no estem logats, aquest login servirà per anar a la part protegida del servlet.

Consulta: En aquesta pàgina, mostrem un formulari per a la consulta del servei web, consulta dels items per preu.

Explicació de la implementació de la llibreria

La implementació de la llibreria va ser un tema complicat, sobretot al principi, el més costos el tema de configurar XML.

Al xml hem configurat el servlet llibreria i la seva servlet-class com a servletdispatch, que és el nom de la classe que s'encarregarà d'això, també configurant el tema dels static, com el css, amb el codi que ens donàveu al campus.

Després de configurar tot el XML ens vam posar amb el servletDispatch, el servletdispatch funciona de la següent manera, si arriba una petició dopost fem el post-redirect-get i si arriba una petició get directament truquem a locationproxy que s'encarregarà de crear una sessió i fer un control de l'usuari, per a veure si l'usuari que està logat existeix o no existeix.

Més endavant comparem totes les URL que ens arribem amb les url que nosaltres volem tenir, i segons el que ens arriba, l'hi carreguem un jsp o un altre.

La implementació dels JSP va ser una implementació fàcil, el més complicat va ser mostrar tot el catàleg i tots els ítems que l'usuari havia estat comprant des de la seva llista d'items comprats a l'array d'usuari.

Quan mostrem el catàleg el que fem es treure-ho de les variables globals que tenim i no directament dels arxius .csv, la càrrega d'aquestes variables globals la fem al principi, a l'ini del servlet, on carreguem la llista d'usuaris trucant a la funció lectorUsuari() i carreguem el catàleg trucant a la funció lectorCataleg().

Quan mostrem les dades de l'usuari, l'únic que fem és primer mirar si ja hi tenia dades, si hi tenia dades, les mostrem, ara mirem els paràmetres que ens arriben que seran els nous productes a afegir a la llista de l'usuari de productes comprats, si aquests productes ja hi són, no fem res, directament mostrem la llista sense fer res, i si no hi són, els afegim a l'usuari que està en l'array d'usuaris i modifiquem el seu crèdit, posteriorment o escrivim a l'arxiu d'usuaris per a no perdre les dades.

Pel que fa al tema de la descàrrega, vam utilitzar la funció que se'ns donava a la web de la pràctica, i modificant-la una mica per adaptar-la a les nostres necessitats.

El tema de la sincronització, va ser un tema complicat, que vam estar debatin molt, ja que no sabíem que sincronitzar, ja que com es sincronitzen les variables globals perquè 2 usuaris no puguin accedir a la mateixa part del codi a la vegada.

Al final ens hem decantat en fer el sincronisme dels threads en la part en la qual emmagatzemem els usuaris a l'arxiu, ja que l'array d'usuaris és un array global, si es vol accedir a aquest array, només és podrà accedir d'un en un.

El serveiweb no l'hem pogut acabar, així que només llegeix l'item de la nostra ip, però mentre apunti a la nostra ip fa tot el que té fer, és a dir, mostrar catàleg de BOOK, AUDIO i VIDEO, i treure un item del, catàleg per jquery i mostrar-ho al client.

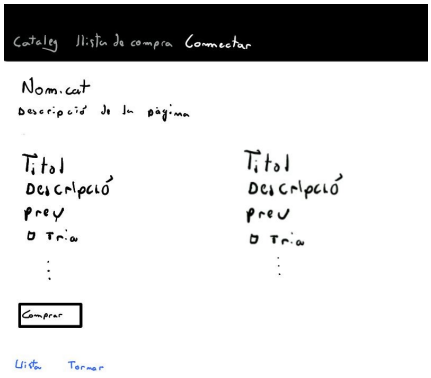
Per a la implementació del serveiweb hem fet un altre servlet, el que, hem definit en el web.xml seguint els mateixos passos que amb el servletDispatch, però amb diferents url, com es pot veure cada vegada que traiem les dades de l'array, mirem quins estan demanats i depèn del que ens estigui demanant, enviem per gson unes dades o unes altres.

Diagrama de classes simplificat MVC

Els diagrames fets a classe són:
- per al tema del disseny **front-end** de la web:

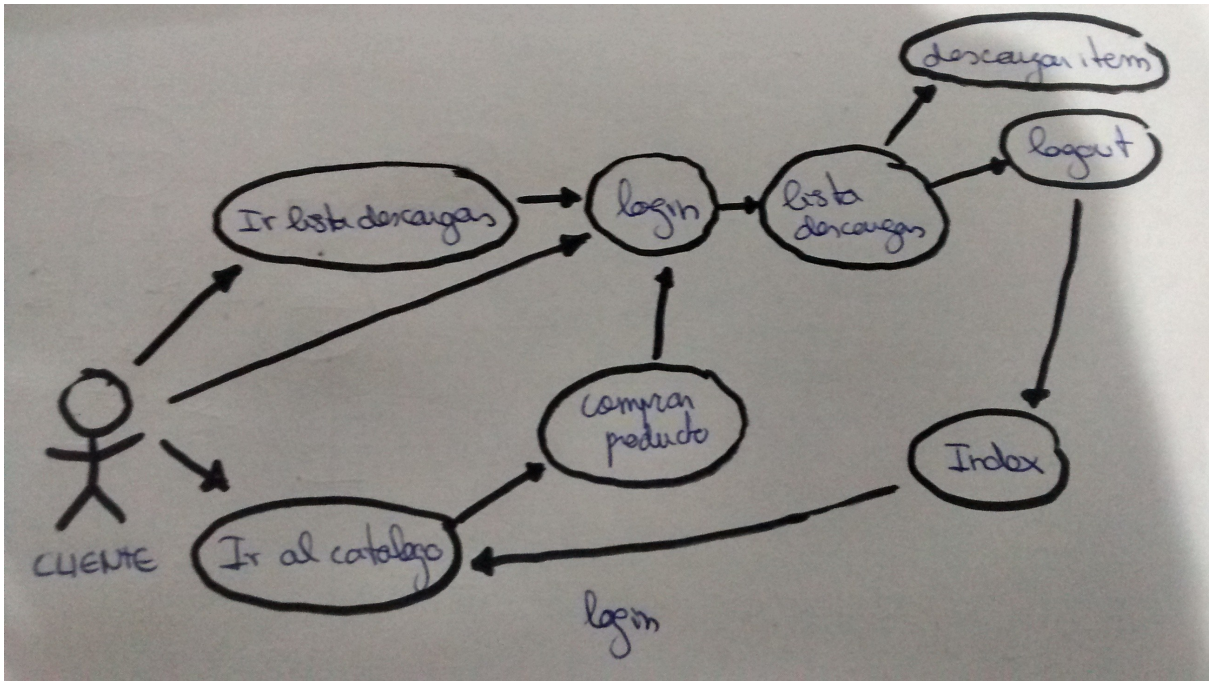
Index.jsp

cataleg.jsp



llista.jsp

diagrama UML de la pràctica:



Mapa Web

Respecte al nostre mapa web, vam seguir les instruccions que sortien a la pàgina de la pràctica sent aquest:

http://localhost:8080/llibreria

→ index.jsp
→ cataleg.jsp
→ **protegit/**
→ llista.jsp
→ login.jsp
→ test.jsp

Cal remarcar que el jsp llista que cau de la carpeta protegit, és perquè és el jsp privat, és a dir, per accedir es necessita fer login, els altres jsp són públics i per tant no es necessita fer login per poder accedir-hi.

Test

L'experiència en la sessió de test no va ser tant satisfactòria com nosaltres volíem que fos, mes que res perquè només tenien fet la webapp, i no teníem res del webservice fet, aleshores només vam poder provar la webapp.

Segons el que ens han anat comentant els nostres companys a la sessió de test, el nostre major problema és el tema de què no funcionava si deshabilitàvem les cookies.

També cal dir que aquest era un problema que teníem gran part dels grups, però sense contar això, de la resta de la webapp no vam tenir queixa, sense contar una mica les queixes d'entorn gràfic, que ens van dir que podíem ficar un botó en un altra banda per què és distingeixi millor, però de funcionalitat, la de les cookies va ser l'única que ens van dir.

Respecte als altres grups, el tema de les cookies va ser el tema que més va donar problemes, altre grup que no em podia logar, etc…

Vam pujar l'informe de la sessió de test on es pot veure per cadascun dels grups els errors que tenien i un petit comentari de l'experiència en la seva pagina web.