**Laborator 2 - Tema EjbJpa**

Pavaluca Tudor-Ioan 244-2

Tema EjbJpa presupune instalarea pe calculatoarele proprii a serverelor de aplicatii (AS) WildFly si GlassFish, si configurarea conectarii la un SGBD, altul decat cel implicit al fiecarui AS.

**Descrierea aplicatiei**

Aplicatia implementata de mine are ca scop principal gestionarea cartilor dintr-o biblioteca. Aplicatia permite utilizatorilor sa se autentifice si sa imprumute anumite carti care sunt momentan disponibile. Pe langa acestea, aplicatia mai permite si adaugarea, stergerea si returnarea unei carti. Utilizatorul poate vedea atat cartile care sunt momentan disponibile, cat si cartile imprumutate de acesta.

**Mod de implementare**

Ca si baza de date este folosit MySQL. Serverul MySQL ruleaza pe portul default 3306. Baza de date are numele „lab2” si contine doua tabele: book si user. Tabelele bazei de date au fost generate automat folsind Hibernate.

Entitatile persistate la nivelul aplicatiei sunt marcate cu adnotarea @Entity, pentru a desemna faptul ca vor reprezenta tabele in baza de date. In cazul nostru, avem entitatile User si Book.

Pentru **Book** avem urmatoarele proprietati, alaturi de metodele get si set.

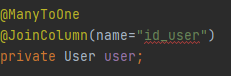
* Id
* Title
* Author
* User

Pentru **User** avem urmatoarele proprietati, alaturi de metodele get si set.

* Id
* Username
* Password
* Books

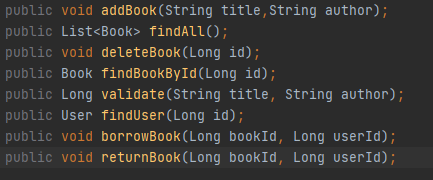
Intre cele doua entitati avem o relatie de tipul One-To-Many. La nivel de implementare, aceasta corespondenta este realizata prin intermediul adnotarilor @OneToMany si @ManyToOne.



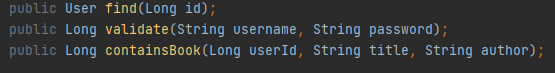


Modulul EJB consta in 2 interfete (BookService si UserService), respectiv in 2 Bean-uri care implementeaza aceste interfete: BookBean si UserBean.

BookService contine urmatoarele metode:



UserService contine urmatoarele metode:



Aceasta aplicatie foloseste 5 servleturi si 3 pagini jsp pentru a raspunde cererilor, plus 2 fisiere CSS pentru stilizare. Toate cele 5 servleturi folosesc injectarea EJB, iar 2 pagini jsp(books.jsp si borrowedBook.jsp) folosesc JNDI.

Servletul Authentication contine o metoda doPost care valideaza credentialele introduse de catre utilizator, iar in cazul in care acestea sunt valide, id-ul utilizatorul este pus pe sesiune si se executa redirectionarea catre books.jsp. Servletul AddBook contine o metoda doPost care apeleaza metoda addBook al bean-ului BookBean pentru a adauga o noua carte in baza de date. Servletul DeleteBook contine si el la randul sau o metoda doPost care apeleaza metoda din BookBean pentru stergerea unei carti din baza de date. Servletele BorrowBook si ReturnBook contin fiecare o metoda doPost pentru imprumutul unei carti, respectiv pentru returnarea acesteia.

Cele 3 pagini JSP au urmatoarele roluri:

* index.jsp – afisarea paginii de start a aplicatiei care contine form-ul de autentificare si trimiterea catre Servlet a datelor introduse de utilizator.
* books.jsp – afisarea cartilor disponibile plus afisarea a 3 form-uri care sunt utilizate pentru adaugarea unei noi carti in baza de date, stergerea unei carti din baza de date si imprumutarea unei carti.
* borrowedBooks.jsp – afisarea cartilor imprumutate de utilizatorul logat alaturi de un form, cu ajutorul caruia se pot returna cartile

Aplicatia mai contine si o clasa Util. Aceasta contine metoda de validare folosita in aplicatie. Pentru maparea servleturilor s-a folosit adnotarea @WebServlet.

**Configurare si modalitati de deploy**

Configurarea WildFly cu MySQL are urmatorii pasi:

* Se downloadeaza (google java jdbc connector mysql) mysql-connector-java-\*.zip din care se extrage mysql-connector-java-\*.jar caruia i se schimba numele in mysqlconnector-java.jar
* Dupa pornirea WildFly (cu $JBOSS\_HOME/bin/standalone) si a serverului MySQL, se lanseaza utilitarul jboss-cli.bat. Prompterul lui este [standalone@localhost:9990 /] si la el se vor da urmatoarele trei comenzi:
* Instalarea modulului connector, cu numele com.mysql, (comanda pe o singura linie): module add --name=com.mysql --resources=mysql-connector-java.jar -- dependencies=javax.api,javax.transaction.api
* Instalarea unui driver, numit mysql, (comanda pe o singura linie, la terminare apare {"outcome" => "success"}): /subsystem=datasources/jdbc-driver=mysql: add(driver-name=mysql,driver-module-name=com.mysql,driver-xa-datasourceclass-name=com.mysql.jdbc.jdbc2.optional.MysqlXADataSource)
* Definirea DataSource, numita MySqlDS, (comanda pe o singura linie), unde baza\_de\_date este inlocuit cu numele bazei de date, user\_name cu numele de utilizator din baza de date, respective user\_password cu parola utilizatorului din baza de date: data-source add --name=MySqlDS --driver-name=mysql --jndiname=java:jboss/datasources/MySqlDS --connectionurl=jdbc:mysql://localhost:3306/baza\_de\_date --user-name=user\_name -- password=user\_password --enabled=true
* Se sterge fisierul $JBOSS\_HOME/bin/mysql-connector-java.jar