**ORM**

*Object-relational mapping* (ORM, mapping) este o tehnică de programare pentru convertirea informațiilor/tipurilor dintr-un sistem orientat-obiect într-o bază de date relațională. Principiul mapării obiect-relație/înregistrare este de a delega altor instrumente managementul persistenței și de a lucra doar cu entitățile din domeniu, nu cu structurile dintr-o bază de date relațională. Instrumentele de mapare obiect-relație stabilesc o legătură bidirecțională între o baza de date relațională și obiectele din sistem, pe baza unei configurații și execută interogări SQL la baza de date (interogări construite dinamic).

Pentru proiectul de față, ORM-ul folosit este Hiberante. Clasele care urmează a fi presistate au urmat regual claselor POJO (Plan Old Java Objects). Hibernate oferă posibilitatea mapării acestor clase folosind adnotarea acestora, cât și reprezentarea sub formă unor fisiere XML. Am ales abordarea din urmă pentru a separa maparea ORM de codul JAVA care implementează clasele. Maparea unei clase JAVA (POJO) constă în maparea fiecărei proprietăți a clasei într-un fișier XML. Acest fișier conține numele clasei mapate, proprietățile cu numele acestora urmate de tipul lor și de coloana care o reprezintă în tabel. În urma acestei corelări se creează un tabel cu numele clasei care are ca și coloane proprietățile descrise.

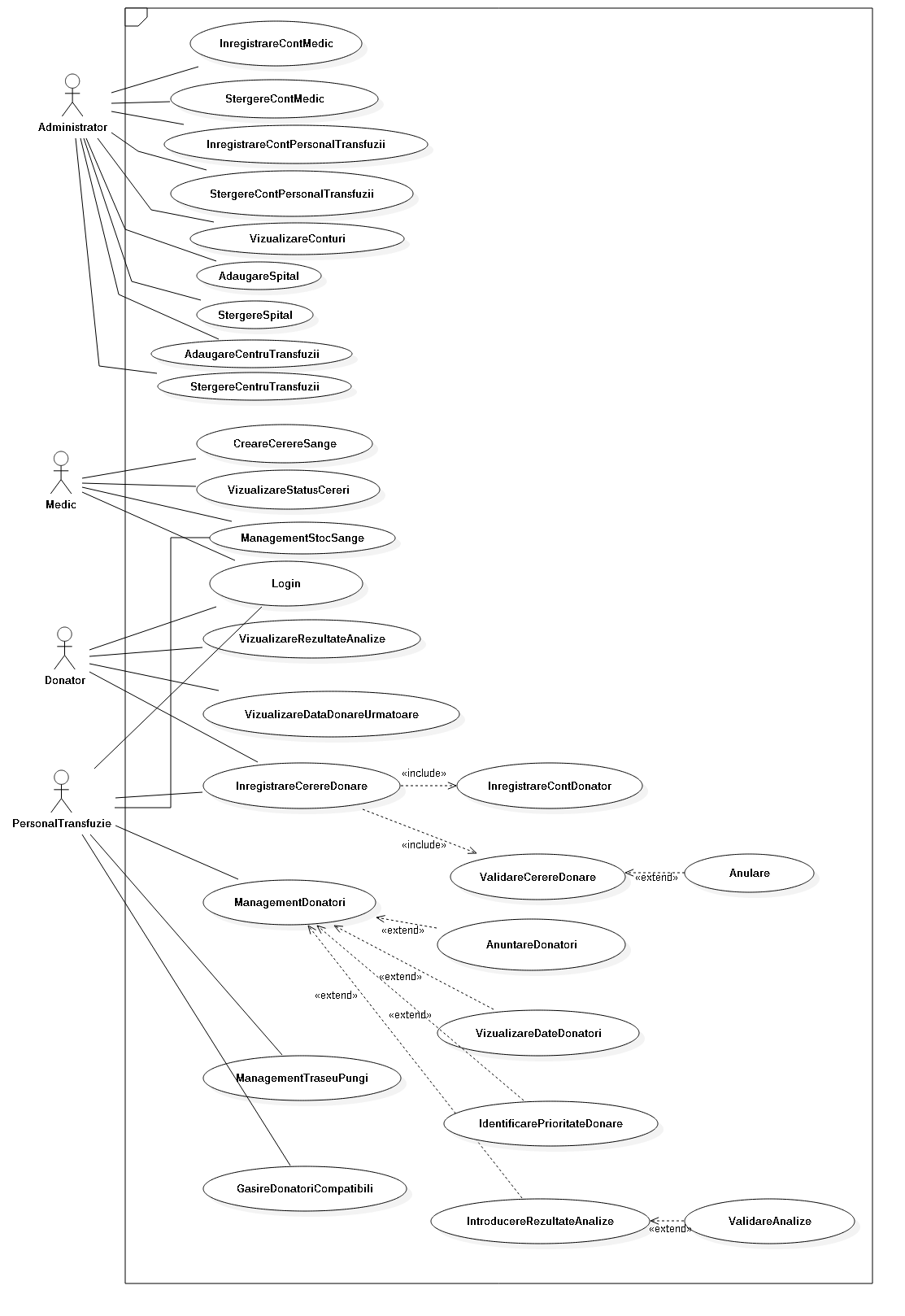
Pentru interogarea bazei de date am folosit atât limbaj SQL, cât și un limbaj specific Hibernate, HQL(Hibernate Query Language).

**StarUML**

UML (unified modeling language) este un limbaj de modelare, folosit pentru a reprezenta soluțiile sistemelor soft, în vederea transpunerii soluțiilor sub formă de cod executabil. Este nu doar un instrument de reprezentare, ci și un instrument de comunicare între participanții la proiect: manageri, clienți, programatori.

StarUML este un soft open-source care folosește limbajul UML pentru modelarea diferitelor diagrame, cu ajutorul cărora se poate genera cod sursă.

Pentru proiectul de față, StarUML a reprezentat o unealtă utilă atât în partea de analiză a cerințelor cât și pentur implementarea soluției (softului). Astfel, am reușit să creem diagrame de secvență, diagrame de comunicare, diagrame de arhitectură, diagramă a cazurilor de utilizare etc., care mai apoi s-au transpus fie în cod sursa utilizat în aplicație, fie într-o mai bună înțelegere atât a domeniului problemei cât și a domeniului soluției.

Diagrama cazurilor de utilizare