Proiect “Car Rental” SGBD



Profesor : Maniu George Constantin

Introducere

Acest proiect are drept scop elaborarea unui database intr-un limbaj SQL cu minim 5 tabele netriviale de forma 3FN. Am ales sa fac o baza de date pentru inchirieri de masini in care vom avea urmatoarele tabele necesare:

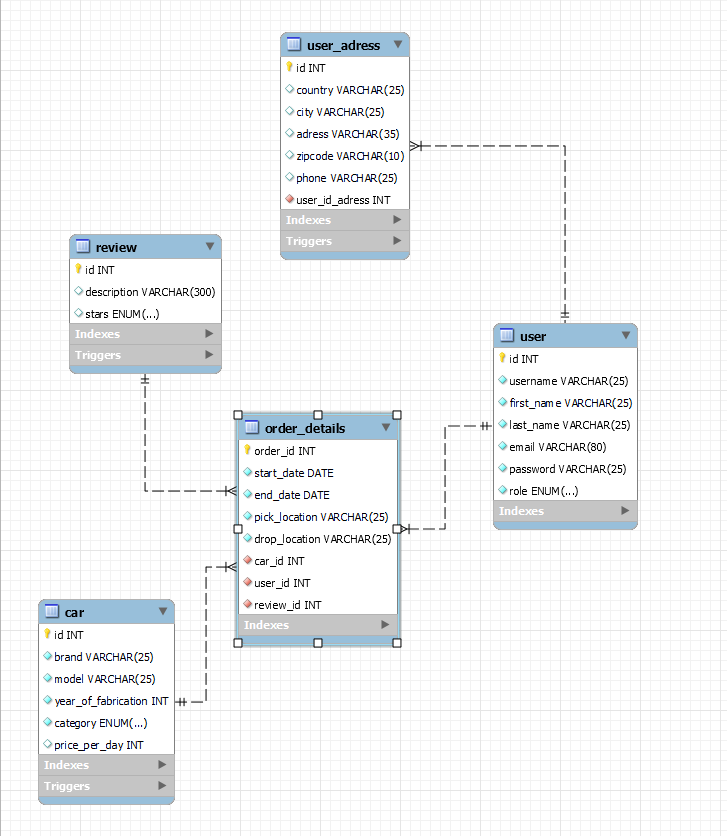
• order\_details

• car

• user

• user\_details

• review



Toate tabelele sunt interconectate prin tabelul Order\_details in care continem informatii despre start date, end-date, pick location, drop location, si bineinteles cu ajutorul FK-urilor, respective car\_id, user\_ir si un review id unic pentru ca un order\_details nu poate sa contina duplicat acelasi review\_id.

Pasii de Lucru

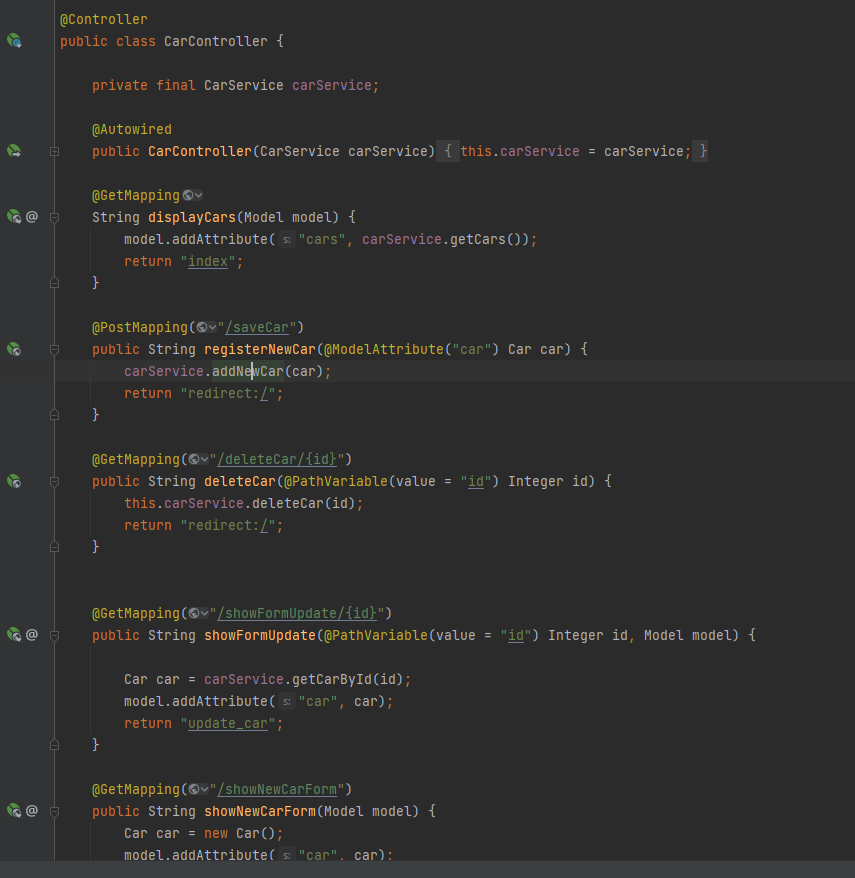
1. Scripturile de creare DB
2. Script Populare DB
3. Creearea a 3 declansatori (Triggers)
4. 3 Functii (fiecare functie sau procedura contine parametric de intrare/iesire dupa caz, instructiuni ciclare (while), cursori, cel putin o comanda select, tartare exceptii).
5. 3 proceduri stocate (fiecare functie sau procedura contine parametric de intrare/iesire dupa caz, instructiuni ciclare (while), cursori, cel putin o comanda select, tartare exceptii).
6. Generarea si explicare Diagramei UML (reprezentata mai sus)
7. Dupa prin intermediul framework-ului Spring din Java am reusit sa fac reverse operation la JPA Hibernate (jpa hibernate este un ORM in care poti prin intermediul claselor sa generezi o baza de date, insa eu am facut operatie inversa, dintr-o baza de date existenta am generat clasele pe care am lucrat.
8. Conectarea cu baza de date
9. CRUD pe un tabel ( am ales tabelul Car pe care pot face crud)
10. Login Authentification protejat de SQL Injection prin intermediul dinamicii, in care folosim si variabile.

Toate scripturile sunt trimise impreuna cu pachetul proiectului SGBD.

Pentru a reusi lega interfata de backend, avem nevoie de mai multe clase care leaga acest process, cum ar fi o interfata Java ce implementeaza JpaRepository(parametru1,parametru2) ce permite CRUD.

Dupa pentru a putea prelua si a efectua operatiile crud, apelam metodele din clasa Jpa, (save,deleteById, saveAll).

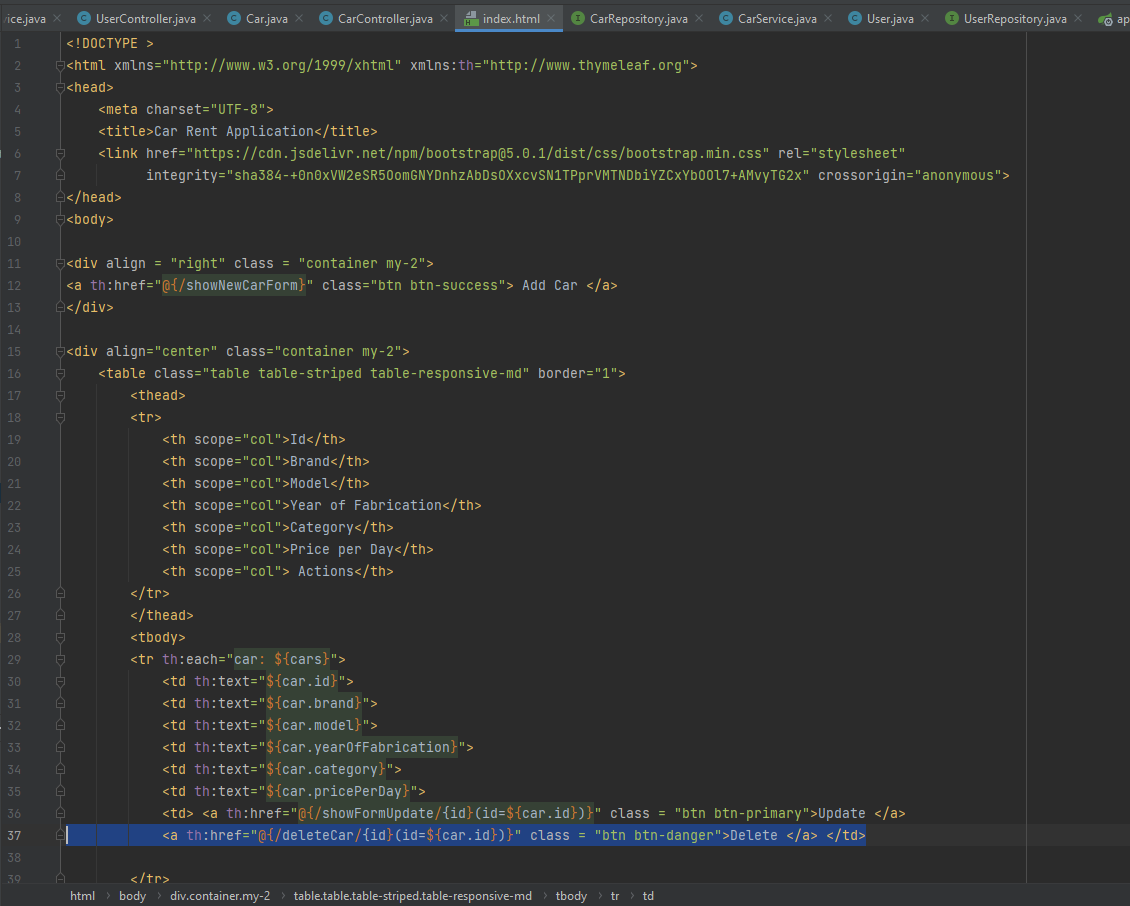
Controllerul este responsabil pentru maparea metodelor pe interfata.

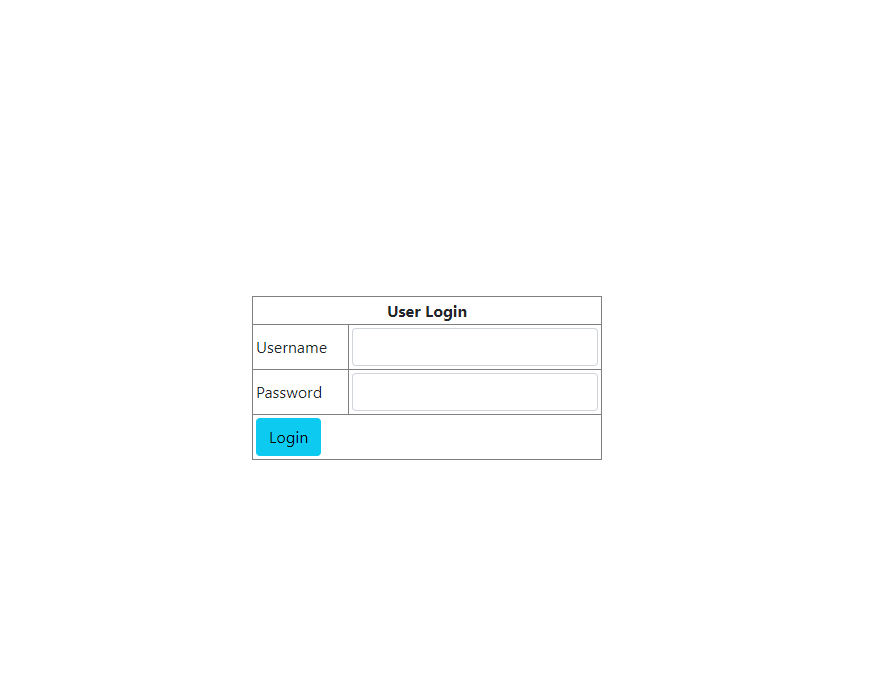


Am respectat principiul SOLID “dependency injection” in care am injectat un carService (carService injecteaza repository-ul), pentru a nu avea cod duplicat si instantiere in plus de obiecte in aplicatie.

Interfata (UI)

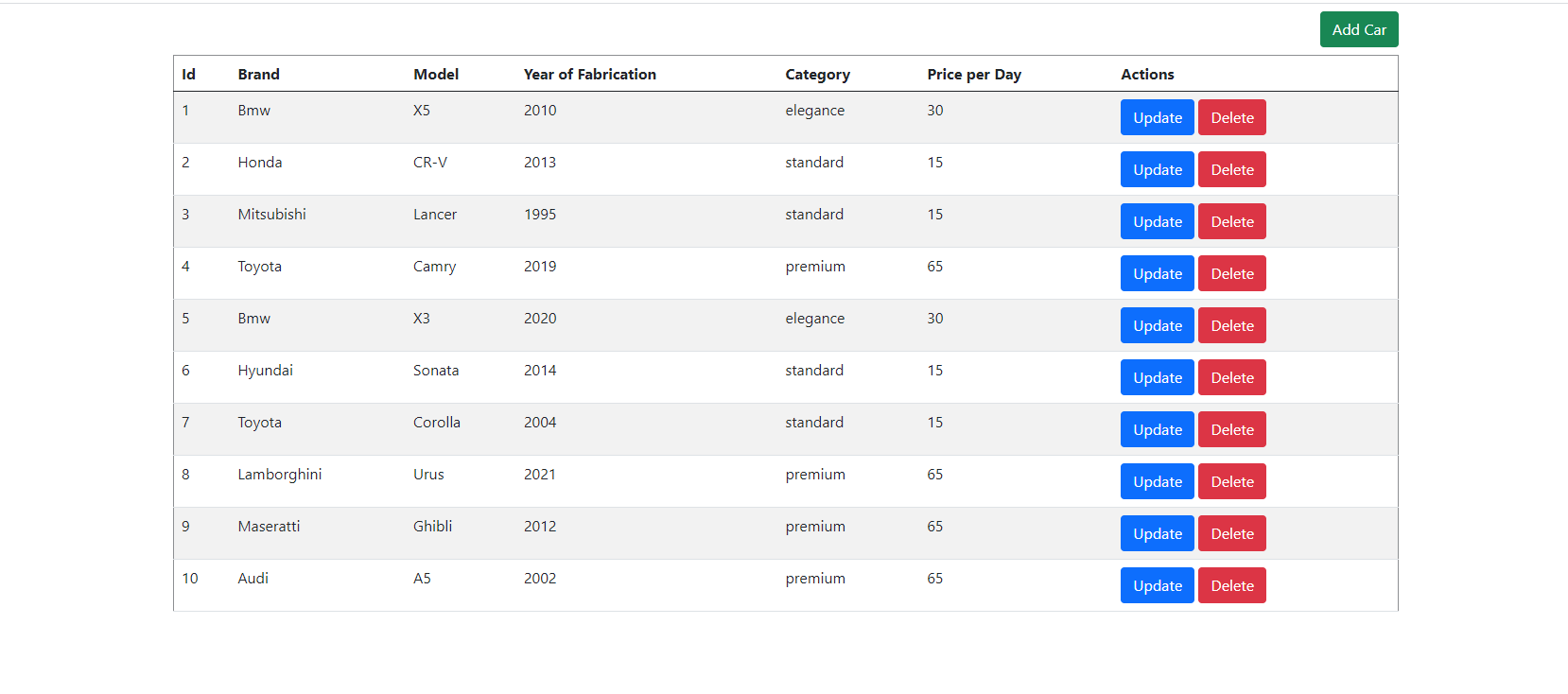
Fiecare mapare returneaza un string (fisierul html) din care se compileaza codul html (cu pluginul thymeleaf) si este afisat utilizatorului. De asemenea, am folosit stilizarea formulalelor cu ajutorul bootstrap.



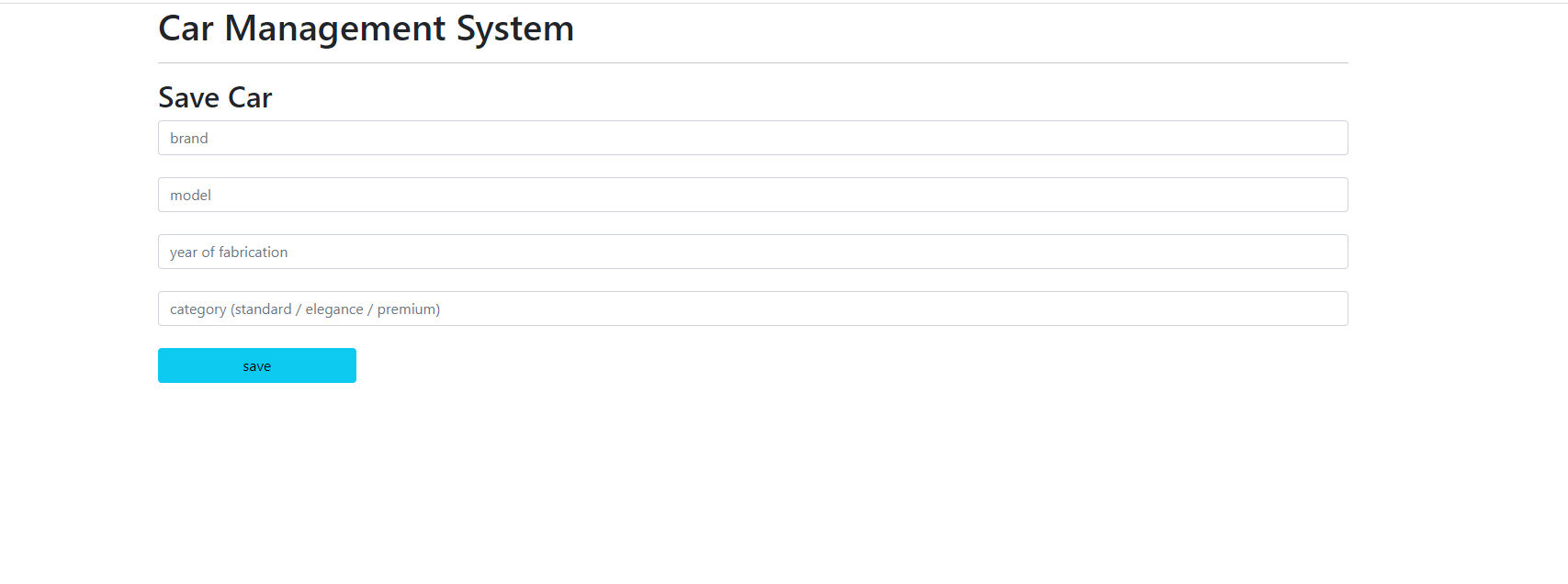


Daca se introduce datele corecte din tabelul User, se redirectioneaza pe pagina index.html, altfel iti face reload la pagina login pentru a introduce datele din nou.

Pagina index.html



Din ce putem observa avem un Read de date dintr-un tabel, astfel putem efectua diverse operatiuni, precum sa adaugam o noua masina, sa facem update pe date iar daca vrem, s-o stergem.



In screenshotul de sus este actiunea unui Create, in care prin intermediul metodei Jpa (save()), salvam masina in db.

Din ce putem observa nu avem campul pentru a introduce pretul zilei, pentru ca prin intermediul categoriei masinii (standard/elegance/premium) se determina pretul masinii.

Triggerul este before INSERT dar si before UPDATE ca in cazul in care facem update pe masina sa nu se reseteze valoarea pretului masinii.

Butonul DELETE din interfata principal apeleaza in spate metoda (Delete()) din JpaCrud, iar in urma apasarii pe buton, se face reload la pagina (“:redirect:/”) cu db-ul updatat.

Daca vrem, putem sa facem Update pe o masina, astfel in urma clickuirii pe butonul UPDATE suntem redirectionati catre pagina de update a masinii



Informatiile ne sunt atasate deja in acest form datorita faptului ca am facut acel select in baza de id-ul masinii pe care am introdus in path-ul URL-ului

<http://localhost:8080/showFormUpdate/1>

unde 1 este id-ul masinii.

Functii, Triggers, Proceduri Stocate

Toate aceste scripturi sunt in fisierul SGBD scripts, insotite de un comentariu in care descriu succinct functionalitatea acestor functii.

Concluzie

Acest proiect a fost o adevarata provocare pentru ca intalnindu-ma printe primele dati cu framework-ul Spring, am facut foarte mult research pentru a-mi indeplini taskurile. Un proiect fullstack ce urmeaza sa fie dezvoltat si adaugat in portofoliul personal.