Sistem de Recomandare și Vânzare de Mașini

Student: Jula Mădălina-Veronica

Grupa: 30233

Echipa: Buta Rada, Jula Mădălina-Veronica, Lungu Andrada-Cristiana

Cuprins

Rezumat

În dezvoltarea aplicației vom folosi o arhitectura împărțita între back-end și front-end fiecare ruland pe câte un port. Clienții vor fii cei care utilizează aplicația iar serverul va fii cel care manageriază bază de date si răspunde clienților.

Fiecare user va avea propria lor accesibilitate la program, singurul user cu accesibilitate completa asupra programului, va fi administratorul.

Userul va putea căuta în funcție de preț o masină, iar aplicația îi va afișa mașinile disponibile in ordine descrescătoare în funție de un rating (notare) pe care l-au dat anumiți clienți.

Se va face conexiunea la server, de unde, pe baza acestui filtru se va face o repartizare cu masinile aflate in stoc.

Toate mașinile disponibile pot fi vizualizate in showroom, iar daca cineva e interesat de vreo mașină acesta se poate loga în contul de utilizator și o poate achiziționa.

De asemenea, clienții care sunt înregistrați pot oferi anumite note mașinilor disponibile, astfel încât să ajute alți utilizatori să aleagă mai ușor mașina dorită.

A.Miniproiect

Implementare:

Controllerul:

```
package org.carsworld.controllers;


import org.carsworld.data.CarRepository;
import org.carsworld.models.Car;

public class CarController {


public String displayAllEvents(Model model) {
    model.addAttribute("title","All cars");
    model.addAttribute("cars", carRepository.findAll());
    return "cars/index";
}


public String processCreateCarForm(@ModelAttribute Car newCar){
    carRepository.save(newCar);
    return "cars/create";
}
```

Data:

```
package org.carsworld.data;

import org.carsworld.models.Car;
import org.springframework.data.repository.CrudRepository;
import org.springframework.stereotype.Repository;

@Repository
public interface CarRepository extends CrudRepository<Car, Integer> {

}
```

Model:

```
package org.carsworld.models;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import java.util.Objects;
@Entity
public class Car {
    @Id
    @GeneratedValue
    private int id;
```

```
    private static int nextId=1;
    private String company;
    private String model;
    private String color;
    private long rolledKilometers;
    private int year;
    private String fuel;
    private String gearbox;
    private int price;
    private double grade;

    public double getGrade() {
        return grade;
    }

    public void setGrade(double grade) {
        this.grade = grade;
    }

    public void setId(int id) {
        this.id = id;
    }
    public String getFuel() {
        return fuel;
    }

    public void setFuel(String fuel) {
        this.fuel = fuel;
    }

    public String getGearbox() {
        return gearbox;
    }

    public void setGearbox(String gearbox) {
        this.gearbox = gearbox;
    }

    public int getPrice() {
        return price;
    }

    public void setPrice(int price) {
        this.price = price;
    }


    public Car(String company, String model,String color,long
rolledKiometers,int year,int price,String gearbox,String fuel,double
grade){

        this.company=company;
        this.model=model;
        this.color=color;
        this.rolledKilometers=rolledKilometers;
        this.year=year;
        this.price=price;
        this.fuel=fuel;
        this.gearbox=gearbox;
        this.grade=grade;
```

```
     }

    public Car(){}
    public String getCompany() {
        return company;
    }

    public void setCompany(String company) {
        this.company = company;
    }

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model= model;
    }

    public int getId() {
        return id;
    }

    public String getColor() {
        return color;
    }

    public void setColor(String color) {
        this.color = color;
    }

    public long getRolledKilometers() {
        return rolledKilometers;
    }

    public void setRolledKilometers(long rolledKilometers) {
        this.rolledKilometers = rolledKilometers;
    }

    @Override //2 cars with same id but different names and descriptions
are not considered the same objects
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Car car = (Car) o;
        return id == car.id;
    }

    public int getYear() {
        return year;
    }

    public void setYear(int year) {
        this.year = year;
    }

    @Override
    public int hashCode() {
        return Objects.hash(id);
    }
```

```
}
```

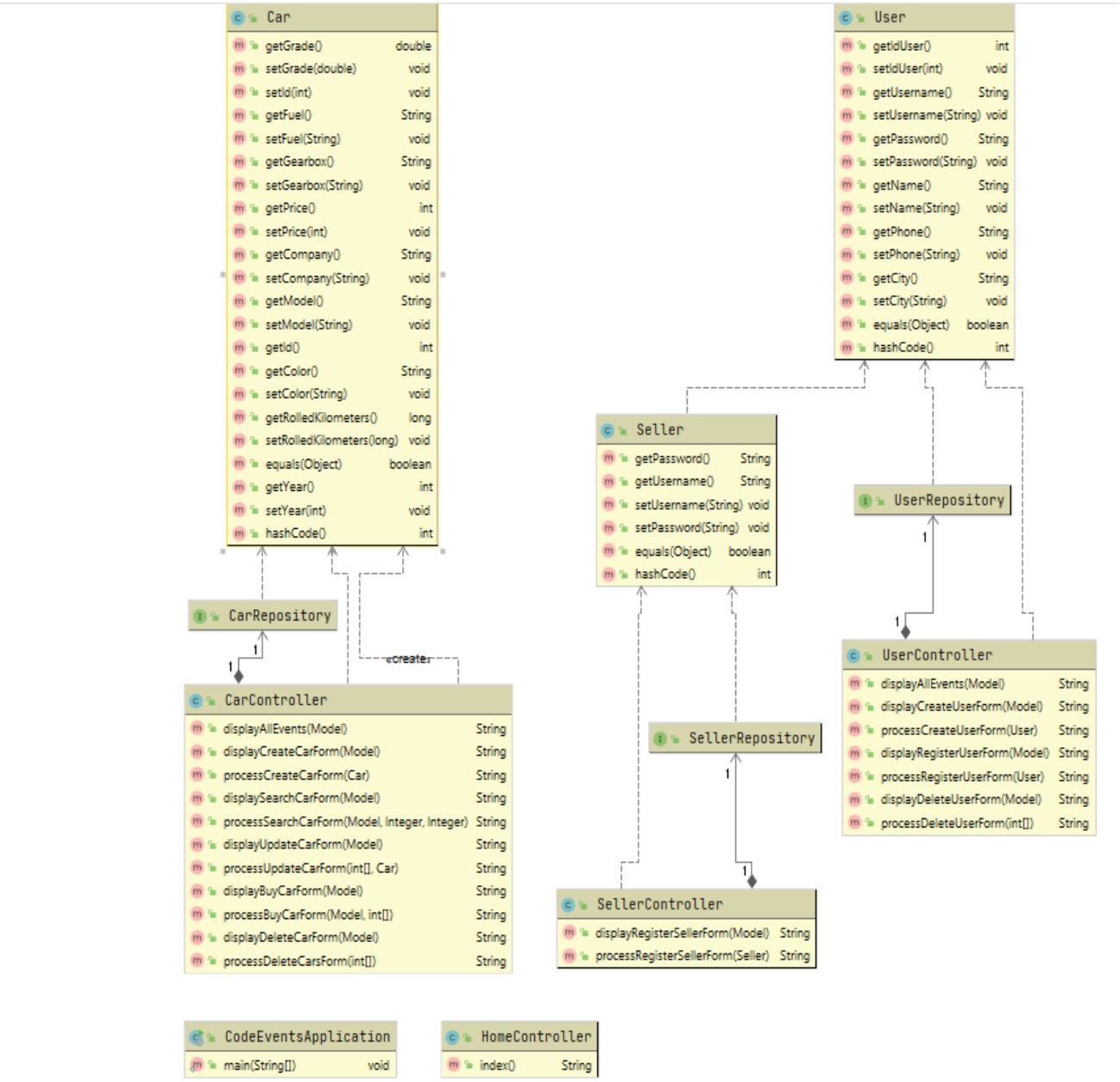
View:

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org/">
<head>

<meta charset="IS0-8859-1">
 <title>CautMasina.ro</title>
    <style>

        .font-36{

            font-size: 24px;

        }

    </style>
</head>
```

B. Proiectul Final

Design:

Diagrama UML:



Modelarea Cazurilor de Utilizare

Sistem de recomandare si vanzare de masini

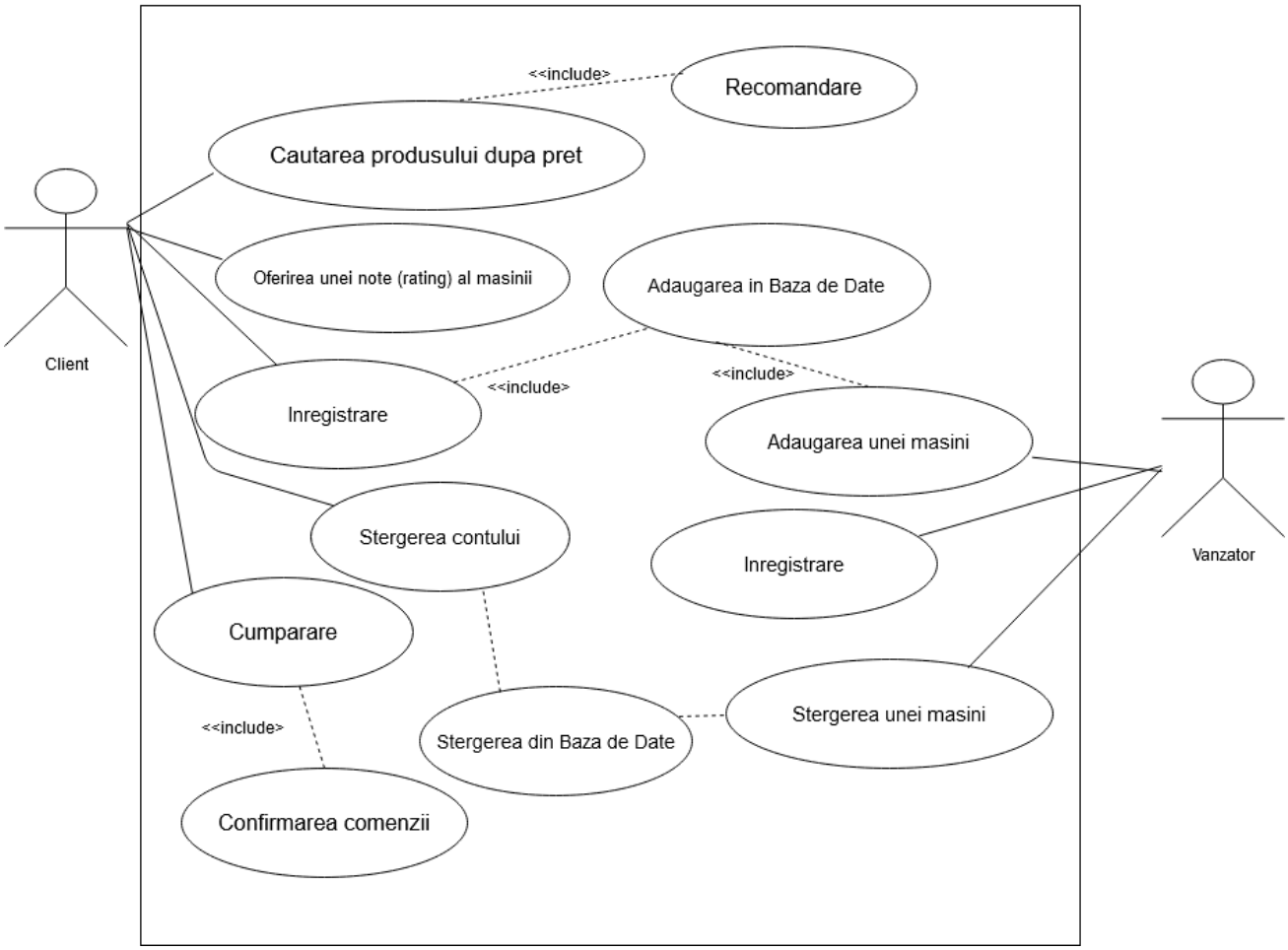
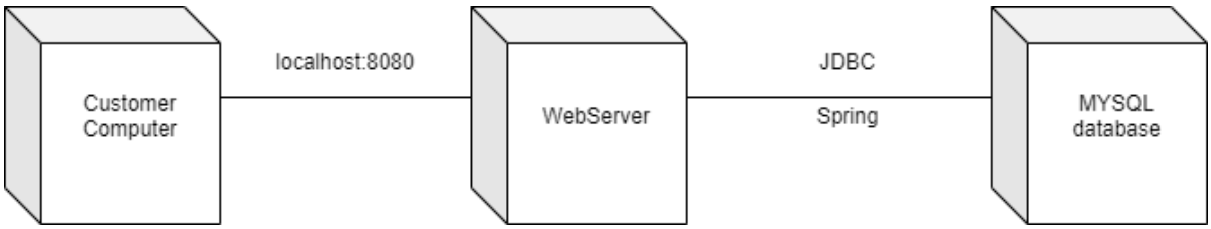


Diagrama Deployment:



Spring

Spring oferă un model cuprinzător de programare și configurare pentru aplicații de întreprindere moderne bazate pe Java - pe orice tip de platformă de implementare. Un element cheie al Spring este suportul infrastructural la nivel de aplicație: Spring se concentrează pe „instalarea” aplicațiilor de întreprindere, astfel încât echipele să se poată concentra pe logica de afaceri la nivel de aplicație, fără legături inutile cu medii de implementare specifice.

Thymeleaf

Thymeleaf este un motor modern de șabloane Java pentru server, atât pentru medii web, cât și pentru medii independente.Scopul principal al Thymeleaf este de a aduce șabloane naturale elegante în fluxul dvs. de lucru de dezvoltare - HTML care poate fi afișat corect în browsere și, de asemenea, poate funcționa ca prototipuri statice, permițând o colaborare mai puternică în echipele de dezvoltare.Cu module pentru Spring Framework, o serie de integrări cu instrumentele dvs. preferate și posibilitatea de a vă conecta la propriile funcționalități, Thymeleaf este ideal pentru dezvoltarea web modernă HTML5 JVM - deși poate face mult mai mult.

AspectJ

AspectJ se referă la un stil de declarare a unor aspecte ca și clase obișnuite de Java declarate cu annotații speciale. Stilul acestui aspect a fost introdus odată cu ieșirea pe piață a AspectJ 5. Spring interpretează aceleași annotații utilizând o biblioteca suplinită de către AspectJ pentru potrivirea și analiza Pointcut-urilor. Deși executabilul AOP este pur Spring, acesta nu folosește nicio dependență față de compilatorul AspectJ.

Acest aspect se poate configura atât prin suport Java, cât și prin suport XML. În acest proiect am folosit configuratorul Java.

Implementare:

```
package org.carsworld.controllers;

import com.mysql.jdbc.SocketFactoryWrapper;
import org.aspectj.lang.annotation.Around;
import org.aspectj.lang.annotation.Before;
import org.aspectj.util.UtilClassLoader;
import org.carsworld.data.CarRepository;
import org.carsworld.models.Car;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;

import java.util.ArrayList;
import java.util.Iterator;

@Controller
@RequestMapping("cars")

public class CarController {

    private final Class<SocketFactoryWrapper> aClass =
SocketFactoryWrapper.class;
    @Autowired
    private CarRepository carRepository;

    @Around("index")
    @GetMapping("index")
    public String displayAllEvents(Model model) {
        model.addAttribute("title","All cars");
        model.addAttribute("cars", carRepository.findAll());
        return "cars/index";
    }
    @Around("create")
    @GetMapping("create")
    public String displayCreateCarForm(Model model){
        model.addAttribute("title","Create car");
        return "cars/create";
    }

    @PostMapping("create")
    public String processCreateCarForm(@ModelAttribute Car newCar){
        carRepository.save(newCar);
        return "cars/create";
    }

    @Around("search")
    @GetMapping("search")
    public String displaySearchCarForm(Model model){
        model.addAttribute("title","Search");
        return "cars/search";
    }
    @PostMapping("search")
    public String processSearchCarForm(Model model,@RequestParam(required =
false) Integer price,@RequestParam(required = false) Integer price2) throws
InterruptedException {

        ArrayList<Car> carsList = new ArrayList<Car>();
        Car carsList2[] = new Car[50];
```

```
        int carsNr = 0;
        Iterator cars = carRepository.findAll().iterator();
        for (Iterator it = cars; it.hasNext(); ) {
            Car c = (Car) it.next();
            if (c.getPrice() >= price && c.getPrice() <= price2) {
                carsList2[carsNr] = c;
                carsNr++;
                carRepository.save(c);
            }
        }
        for (int i = 0; i < carsNr - 1; i++)
            for (int j = i + 1; j < carsNr; j++)
                if (carsList2[i].getGrade() < carsList2[j].getGrade()) {
                    Car aux = new Car();
                    aux = carsList2[i];
                    carsList2[i] = carsList2[j];
                    carsList2[j] = aux;
                }
        for (int i = 0; i < carsNr; i++) {
            carsList.add(carsList2[i]);
        }
        model.addAttribute("cars", carsList);

        return "cars/search";
    }

    @Around("update")
    @GetMapping("update")
    public String displayUpdateCarForm(Model model){
        model.addAttribute("title","Create car");
        model.addAttribute("cars",carRepository.findAll());
        return "cars/update";
    }

    @Around("update")
    @PostMapping("update")
    public String processUpdateCarForm(@RequestParam(required = false)
int[] carIds ,@ModelAttribute Car newCar) {
        Iterator cars=carRepository.findAll().iterator();
        for (Iterator it = cars; it.hasNext(); ) {
            Car c = (Car) it.next();
            for (int id : carIds) {
                System.out.println(c.getId()+" " + c.getCompany()+ " "+id);
                if(id == c.getId()){
                    c.setGrade((newCar.getGrade()+c.getGrade())/2);
                    carRepository.save(c);
                }
            }


        }
        return "cars/update";
    }

    @Around("buy")
    @GetMapping("buy")
    public String displayBuyCarForm(Model model){
        model.addAttribute("title","Buy car");
        model.addAttribute("cars",carRepository.findAll());
        return "cars/buy";
    }
```

```
    @Around("buy")
    @PostMapping("buy")
    public String processBuyCarForm(Model model,@RequestParam(required =
false) int[] carIds ) throws InterruptedException {

        ArrayList<Car> carsList =new ArrayList<Car>();
        if (carIds != null) {
            for (int id : carIds) {
                for (Car car:carRepository.findAll())
                    if(car.getId()==id){
                        // System.out.println(car.getId());
                        carsList.add(car);
                        //   model.addAttribute("cars",carsList);
                    }

            }
        }
        model.addAttribute("cars",carsList);
        return "cars/buy";
    }

    @Around("delete")
    @GetMapping("delete")
    public String displayDeleteCarForm(Model model){
        model.addAttribute("title","Delete Cars");
        model.addAttribute("cars",carRepository.findAll());
        return "cars/delete";
    }

    @Around("delete")
    @PostMapping("delete")
    public String processDeleteCarsForm(@RequestParam(required = false)
int[] carIds )
    {    if (carIds != null) {
        for (int id : carIds) {
            carRepository.deleteById(id);

        }
    }
        return "cars/delete";

    }
}
```

```
package org.carsworld.controllers;

import org.aspectj.lang.annotation.Around;
import org.carsworld.data.CarRepository;
import org.carsworld.data.SellerRepository;
import org.carsworld.models.Car;
import org.carsworld.models.Seller;
import org.carsworld.models.User;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
```

```
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;

@Controller
@RequestMapping("seller")
public class SellerController {
    @Autowired
    private SellerRepository sellerRepository;

    @Around("register")
    @GetMapping("register")
    public String displayRegisterSellerForm(Model model){
        model.addAttribute("title","Register seller");
        return "seller/register";
    }

    @Around("register")
    @PostMapping("register")
    public String processRegisterSellerForm(@ModelAttribute Seller seller){

            if(seller.getUsername().equals("seller") &&
seller.getPassword().equals("seller")) {
                return "seller/choose";
            }

        return "seller/register";
    }


}
```

```
package org.carsworld.controllers;


import org.carsworld.data.UserRepository;
import org.carsworld.models.User;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;

@Controller
@RequestMapping("users")
public class UserController {
    @Autowired
    private UserRepository userRepository;

    @GetMapping("index")
    public String displayAllEvents(Model model) {
        model.addAttribute("title","All users");
        model.addAttribute("users", userRepository.findAll());
        return "users/index";
    }

    @GetMapping("create")
    public String displayCreateUserForm(Model model){
        model.addAttribute("title","Create user");
```

```
        return "users/create";
    }

    @PostMapping("create")
    public String processCreateUserForm(@ModelAttribute User newUser){
        userRepository.save(newUser);
        return "users/create";
    }


    @GetMapping("register")
    public String displayRegisterUserForm(Model model){
        model.addAttribute("title","Register user");
        return "users/register";
    }

    @PostMapping("register")
    public String processRegisterUserForm(@ModelAttribute User user){
        for (User user1:userRepository.findAll()) {
            if(user1.getUsername().equals(user.getUsername()) &&
user1.getPassword().equals(user.getPassword())){
                return "users/choose";
            }
        }
        return "users/register";
    }

    @GetMapping("delete")
    public String displayDeleteUserForm(Model model){
        model.addAttribute("title","Delete User");
        model.addAttribute("users",userRepository.findAll());
        return "users/delete";
    }

    @PostMapping("delete")
    public String processDeleteUserForm(@RequestParam(required = false)
int[] userIds )
    {    if (userIds != null) {
        for (int id : userIds) {
            userRepository.deleteById(id);

        }
    }
        return "users/delete";

    }
}
```

```
package org.carsworld.controllers;

import org.aspectj.lang.annotation.Aspect;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;

@Aspect
@Controller
@RequestMapping("/home")
public class HomeController {
```

```
    @GetMapping
    public String index(){
        return "index";
    }
}
```

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org/">
<head>
   <!-- <style>
        body {
            background-image: url("static/images/carr.jpg");
            background-repeat: no-repeat;
            background-attachment: fixed;
            background-size: cover;
        }
    </style>
-->
<meta charset="ISO-8859-1">
 <title>CautMasina.ro</title>
    <style>

        .font-36{

            font-size: 24px;

        }

    </style>
</head>

<body class="container"
      style="background: antiquewhite" >

<!--
<div class="font-36" align="center">
    <img th:src="@{images/car.jpg}" width="400" />
</div>
-->

<div class="font-36" align="right" style="font-family: 'Century Gothic'">
         <a href="users/create">  Sign in  </a>
</div>
<div class="font-36" align="right" style="font-family: 'Century Gothic'">
             <a href="users/register">  Log in  </a>
</div>

<div class="font-36" align="center">
    <img th:src="@{images/logo.webp}" width="100" />
</div>

<div class="font-36" align="center"
     style="color: cornflowerblue; font-family: 'Arial Black'">
    <h1 href="/home"> CautMasina.ro</h1>
</div>
```

```
<div class="font-36" align="left" style="font-family: 'Century Gothic'">
     <a href="seller/register">  Ai o masina de vanzare? </a>
</div>

<div class="font-36" align="left" style="font-family: 'Century Gothic'">
     <a href="users/index">  Aici urmaresti toti userii </a>
</div>


 <div class="font-36" align="left" style="font-family: 'Century Gothic'">
     <a href="users/delete">  Daca vrei sa iti stergi contul, intra aici
</a>
 </div>

<!--
 <div class="font-36" align="left" style="font-family: 'Century Gothic'">
     <a href="cars/update">  Update</a>
 </div>
-->

<p>

</p>
 <div class="row">
     <div class="column">
         <img th:src="@{images/car1.jpeg}" width="400"/>
         <img th:src="@{images/car.jpg}" width="400"/>
         <img th:src="@{images/car2.jpeg}" width="400"/>
     </div>
     <!--
     <div class="column">
         <img th:src="@{images/car.jpg}" width="400"/>
     </div>
     <div class="column">
         <img th:src="@{images/car2.jpeg}" width="400"/>
     </div>-->
</div>


<div class="font-36" align="center"
     style="color: crimson; font-family: 'Arial Black'">
    <cite> <h1 href="/home"> Gaseste masina visurilor tale! </h1> </cite>
</div>


<div class="font-36" align="center" style="font-family: Elephant; font-
size:30px"; >
    <a href="cars/index">  Showroom de masini </a>
</div>

</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org/">
<head>
    <meta charset="IS0-8859-1">
    <title> Inregistreaza-te ca user </title>
    <style>
```

```
        .font-36{

            font-size: 24px;

        }

    </style>
<body class="container"
      style="background: antiquewhite" >

<div align="center" style="color: cornflowerblue">
    <h1 href="/users/register"> Inregistreaza-te ca user </h1>
</div>
<form method="post" >
<div class="form-group font-36" align ="center" >
    <label>Username
        <input type="text" name="username" class="form-control">
    </label>
</div>

<div class="form-group font-36" align="center">
    <label>Password
        <input type="text" name="password" class="form-control">
    </label>
</div>

<div class="form-group font-36" align="center">
    <input type="submit" value = "Register" class="btn btn-success">
</div>
</form>
</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org/">
<head>
    <meta charset="IS0-8859-1">
    <title>Create user</title>
<style>

    .font-36{

        font-size: 24px;

    }

</style>
<body class="container"
      style="background: antiquewhite" >

<div align="center" style="color: cornflowerblue">
    <h1 href="/users/create">Creaza un cont de user</h1>

</div>

<form method="post" >
    <div class="form-group font-36" align="center" >
        <label>Username
            <input type="text" name="username" class="form-control">
        </label>
```

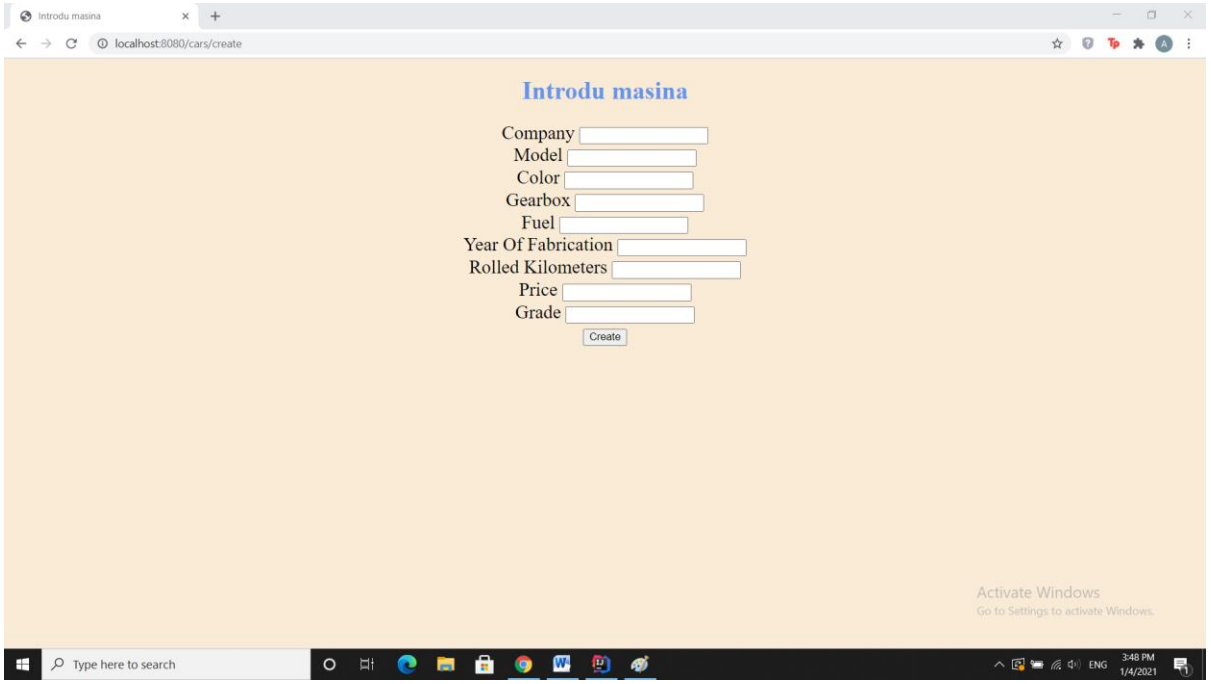
```
    </div>
    <div class="form-group font-36" align="center">
        <label>Password
            <input type="text" name="password" class="form-control">
        </label>
    </div>
    <div class="form-group font-36" align="center">
        <label>Name
            <input type="text" name="name" class="form-control">
        </label>
    </div>
    <div class="form-group font-36" align="center">
        <label>Phone
            <input type="text" name="phone" class="form-control">
        </label>
    </div>
    <div class="form-group font-36" align="center">
        <label>City
            <input type="text" name="city" class="form-control">
        </label>
    </div>
    <div class="form-group font-36" align="center">
        <input type="submit" value = "Create" class="btn btn-success">
    </div>

</form>
</body>
</html>
```

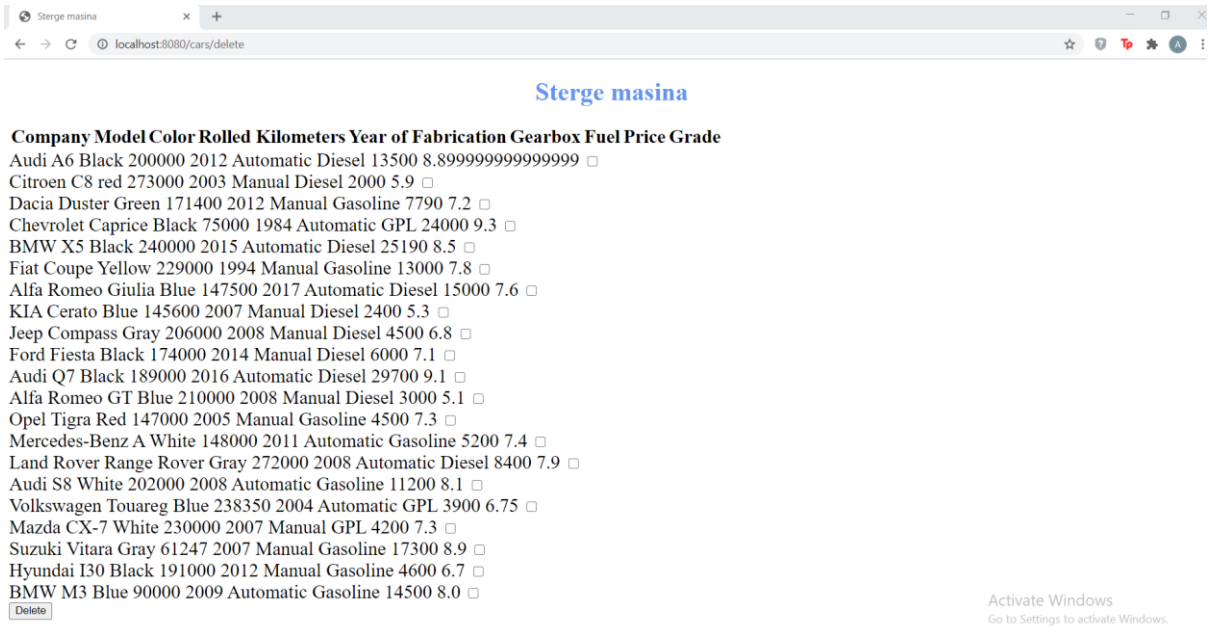
```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org/">
<head>
    <meta charset="IS0-8859-1">
    <title> Inregistreaza-te ca vanzator </title>
    <style>

        .font-36{

            font-size: 24px;

        }

    </style>
<body class="container"
      style="background: antiquewhite" >

<div align="center" style="color: cornflowerblue">
    <h1 href="/seller/register"> Inregistreaza-te ca vanzator </h1>
</div>
<form method="post" >
    <div class="form-group font-36" align ="center" >
        <label>Username
            <input type="text" name="username" class="form-control">
        </label>
    </div>

    <div class="form-group font-36" align="center">
        <label>Password
            <input type="text" name="password" class="form-control">
        </label>
```

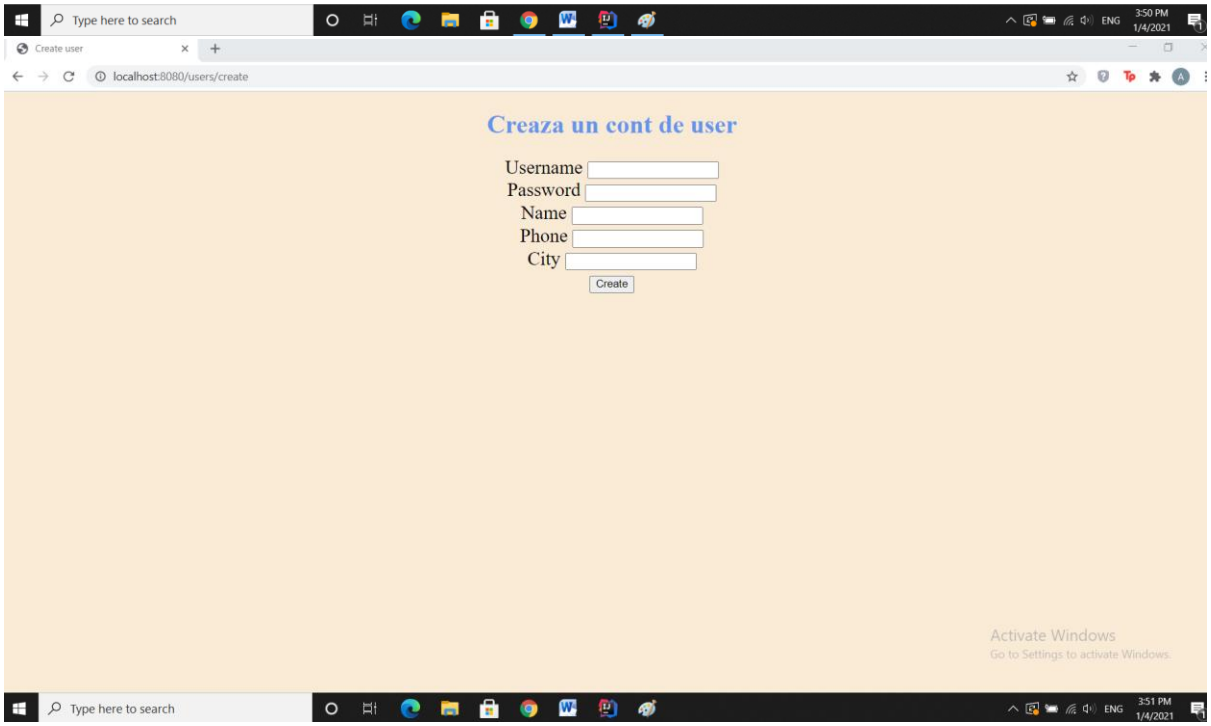
```
    </div>

    <div class="form-group" align="center">
        <input type="submit" value = "Register" class="btn btn-success">
    </div>
</form>
</body>
</html>
```

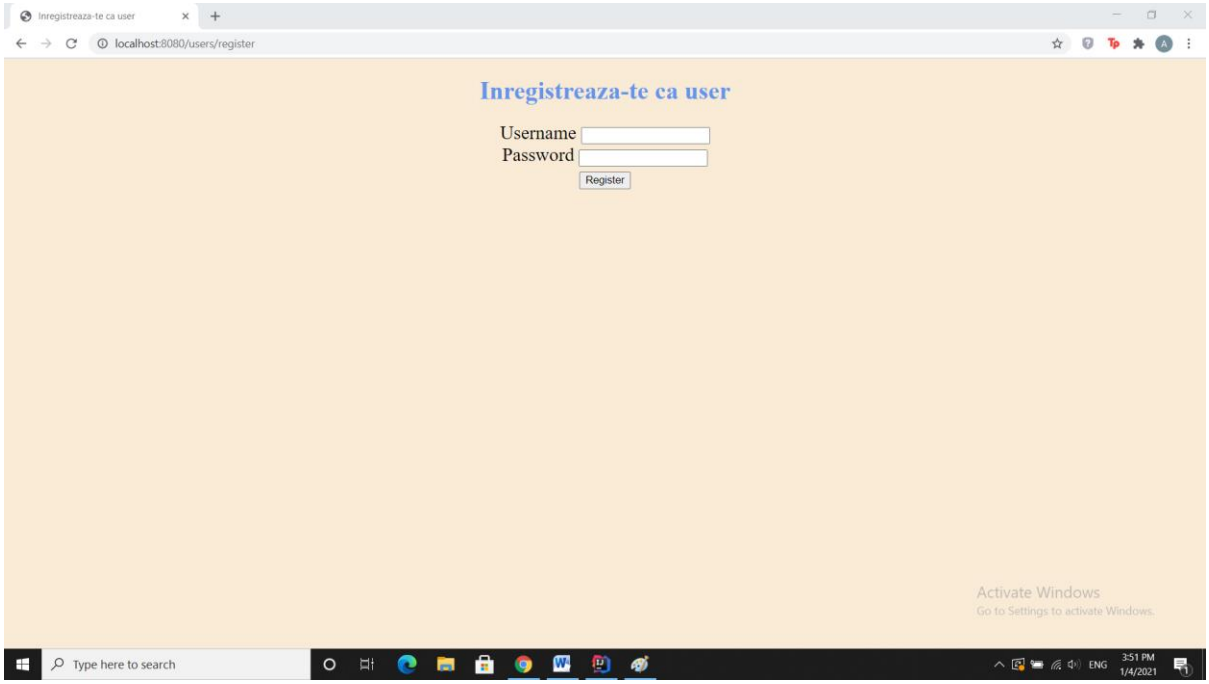
```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org/">
<head>
    <meta charset="IS0-8859-1">
    <title>Create/Delete car</title>
    <style>

        .font-36{

            font-size: 24px;

        }

    </style>
<body class="container"
      style="background: antiquewhite" >

<div class="font-36" align="center">
    <h1 href="home"></h1>
</div>

<div class="font-36" align="center" >
    <a href="/cars/create">Click here to create a car</a>
</div>

<div class="font-36" align="center">
    <a href="/cars/delete">Click here to delete a car</a>
</div>


</body>
</html>
```

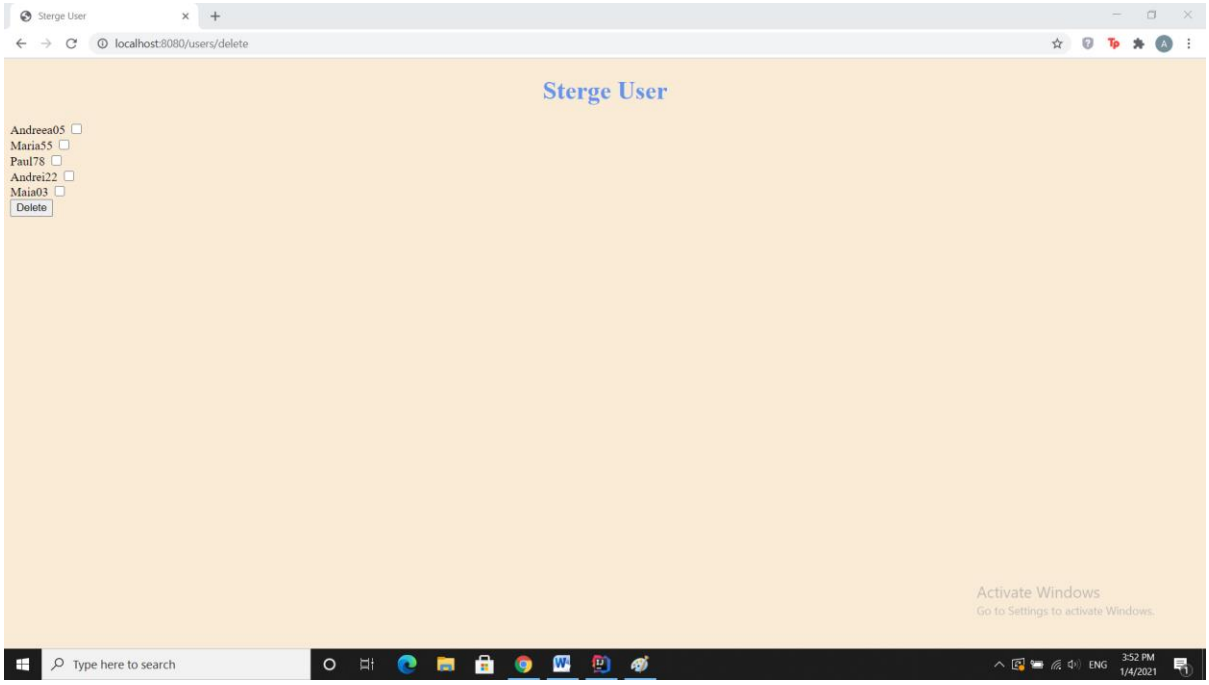

CautMasina.ro
localhost:8080
Sign in
Log in
CautMasina.ro
Ai o masina de vanzare?
Aici urmaresti toti userii
Daca vrei sa iti stergi contul, intra aici
Gaseste masina visurilor tale!
Showroom de masini
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
3:46 PM
1/4/2021

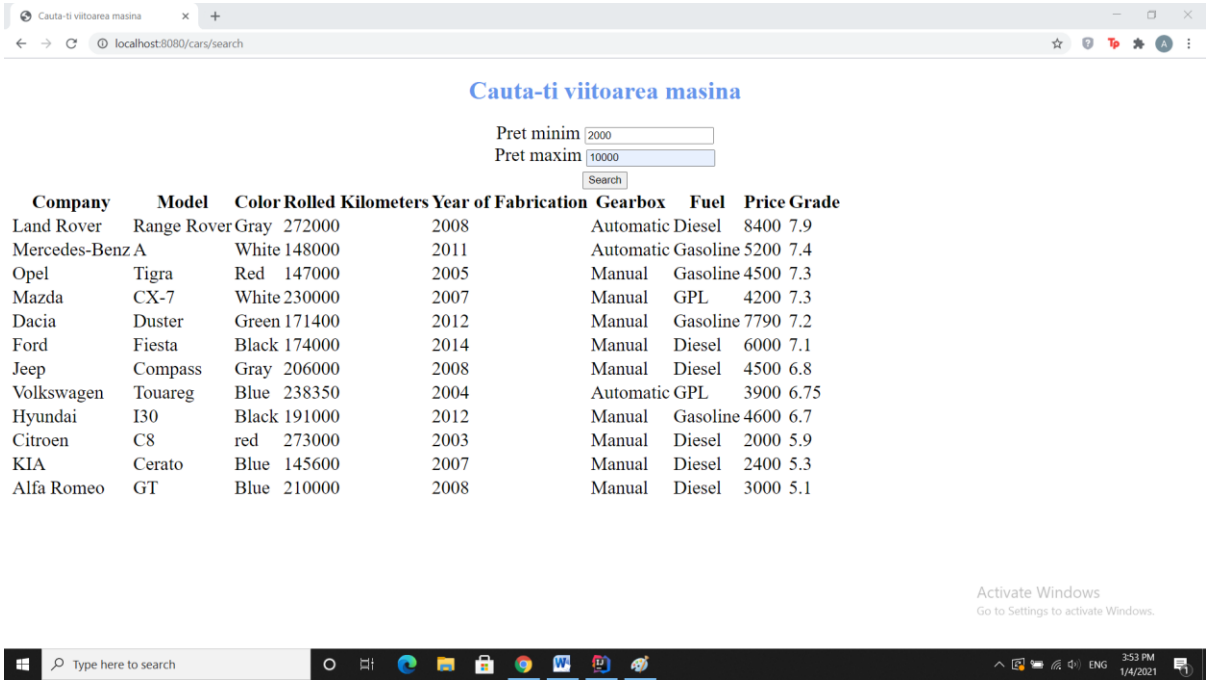

Introdu masina
localhost:8080/cars/create
Introdu masina
Company
Model
Color
Gearbox
Fuel
Year Of Fabrication
Rolled Kilometers
Price
Grade
Create
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
3:48 PM
1/4/2021

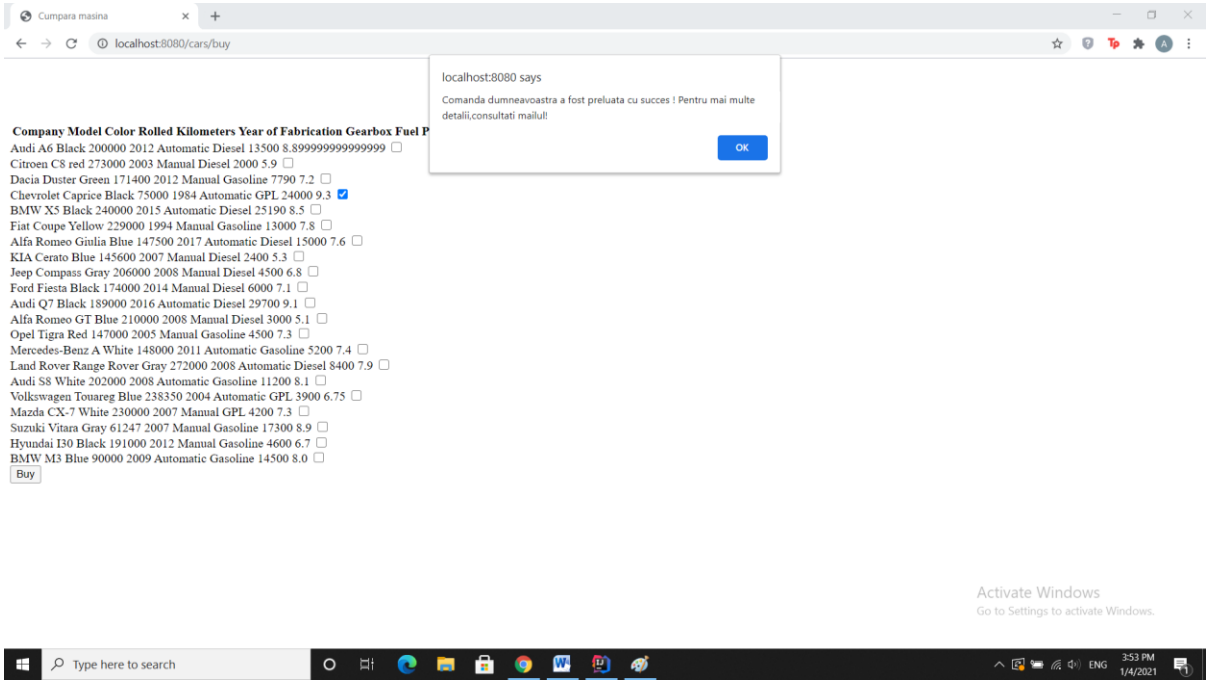

Sterge masina
localhost:8080/cars/delete
Sterge masina
Company Model Color Rolled Kilometers Year of Fabrication Gearbox Fuel Price Grade
Audi A6 Black 200000 2012 Automatic Diesel 13500 8.899999999999999
Citroen C8 red 273000 2003 Manual Diesel 2000 5.9
Dacia Duster Green 171400 2012 Manual Gasoline 7790 7.2
Chevrolet Caprice Black 75000 1984 Automatic GPL 24000 9.3
BMW X5 Black 240000 2015 Automatic Diesel 25190 8.5
Fiat Coupe Yellow 229000 1994 Manual Gasoline 13000 7.8
Alfa Romeo Giulia Blue 147500 2017 Automatic Diesel 15000 7.6
KIA Cerato Blue 145600 2007 Manual Diesel 2400 5.3
Jeep Compass Gray 206000 2008 Manual Diesel 4500 6.8
Ford Fiesta Black 174000 2014 Manual Diesel 6000 7.1
Audi Q7 Black 189000 2016 Automatic Diesel 29700 9.1
Alfa Romeo GT Blue 210000 2008 Manual Diesel 3000 5.1
Opel Tigra Red 147000 2005 Manual Gasoline 4500 7.3
Mercedes-Benz A White 148000 2011 Automatic Gasoline 5200 7.4
Land Rover Range Rover Gray 272000 2008 Automatic Diesel 8400 7.9
Audi S8 White 202000 2008 Automatic Gasoline 11200 8.1
Volkswagen Touareg Blue 238350 2004 Automatic GPL 3900 6.75
Mazda CX-7 White 230000 2007 Manual GPL 4200 7.3
Suzuki Vitara Gray 61247 2007 Manual Gasoline 17300 8.9
Hyundai I30 Black 191000 2012 Manual Gasoline 4600 6.7
BMW M3 Blue 90000 2009 Automatic Gasoline 14500 8.0
Delete
Activate Windows
Go to Settings to activate Windows.

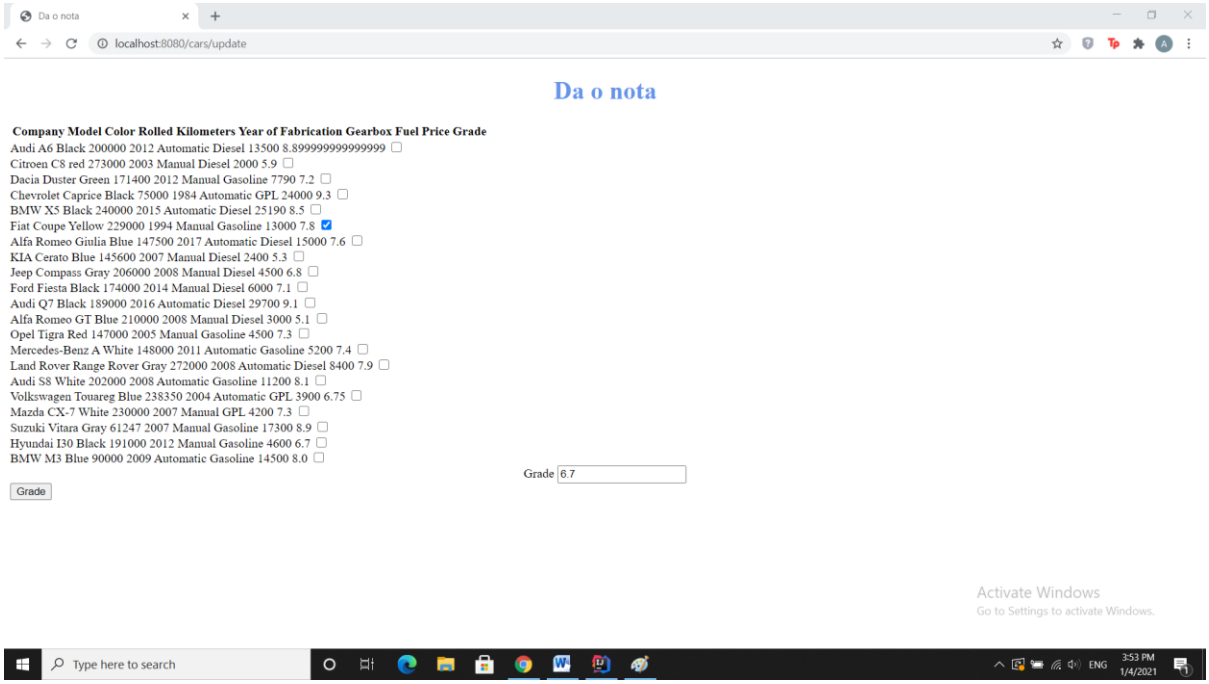

Type here to search
ENG
3:50 PM
1/4/2021
Create user
localhost:8080/users/create
Creaza un cont de user
Username
Password
Name
Phone
City
Create
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
3:51 PM
1/4/2021

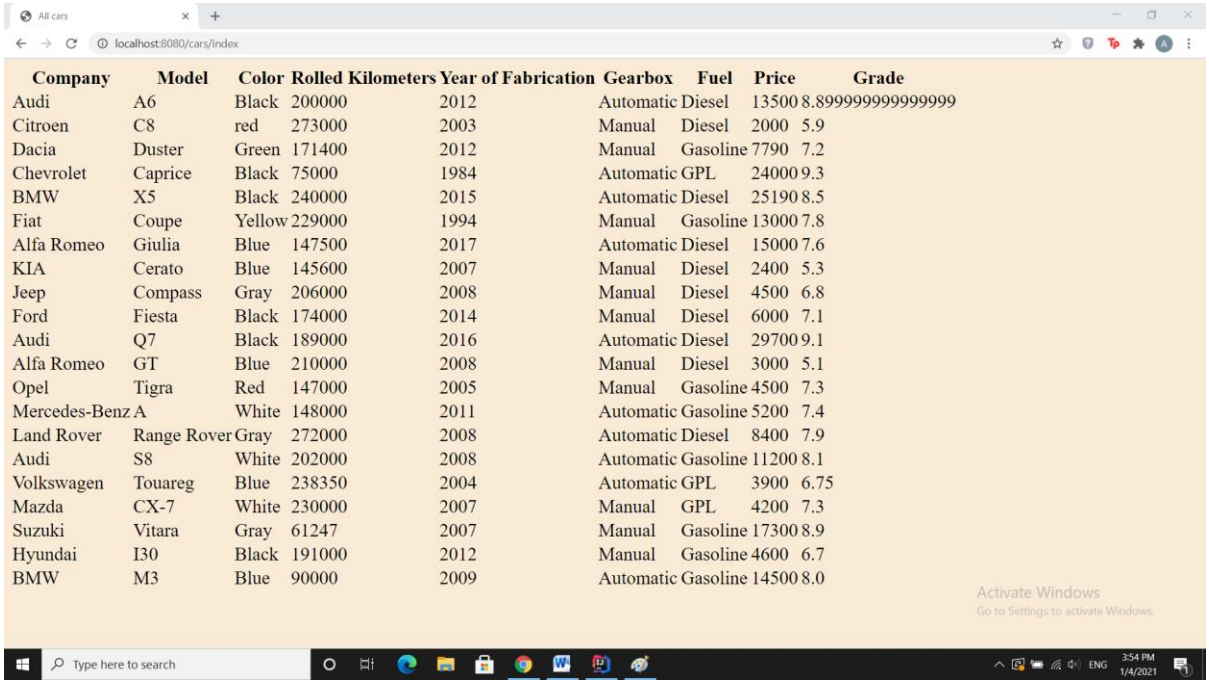

Inregistreaza-te ca user
localhost:8080/users/register
Inregistreaza-te ca user
Username
Password
Register
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
3:51 PM
1/4/2021


Sterge User
localhost:8080/users/delete
Sterge User
Andreea05
Maria55
Paul78
Andrei22
Maia03
Delete
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
3:52 PM
1/4/2021


Cauta-ti viitoarea masina
localhost:8080/cars/search
Cauta-ti viitoarea masina
Pret minim 2000
Pret maxim 10000
Search
Company Model Color Rolled Kilometers Year of Fabrication Gearbox Fuel Price Grade
Land Rover Range Rover Gray 272000 2008 Automatic Diesel 8400 7.9
Mercedes-Benz A White 148000 2011 Automatic Gasoline 5200 7.4
Opel Tigra Red 147000 2005 Manual Gasoline 4500 7.3
Mazda CX-7 White 230000 2007 Manual GPL 4200 7.3
Dacia Duster Green 171400 2012 Manual Gasoline 7790 7.2
Ford Fiesta Black 174000 2014 Manual Diesel 6000 7.1
Jeep Compass Gray 206000 2008 Manual Diesel 4500 6.8
Volkswagen Touareg Blue 238350 2004 Automatic GPL 3900 6.75
Hyundai I30 Black 191000 2012 Manual Gasoline 4600 6.7
Citroen C8 red 273000 2003 Manual Diesel 2000 5.9
KIA Cerato Blue 145600 2007 Manual Diesel 2400 5.3
Alfa Romeo GT Blue 210000 2008 Manual Diesel 3000 5.1
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
3:53 PM
1/4/2021


Cumpara masina
localhost:8080/cars/buy
localhost:8080 says
Comanda dumneavoastra a fost preluata cu succes ! Pentru mai multe detalii,consultati mailul!
OK
Company Model Color Rolled Kilometers Year of Fabrication Gearbox Fuel P
Audi A6 Black 200000 2012 Automatic Diesel 13500 8.899999999999999
Citroen C8 red 273000 2003 Manual Diesel 2000 5.9
Dacia Duster Green 171400 2012 Manual Gasoline 7790 7.2
Chevrolet Caprice Black 75000 1984 Automatic GPL 24000 9.3
BMW X5 Black 240000 2015 Automatic Diesel 25190 8.5
Fiat Coupe Yellow 229000 1994 Manual Gasoline 13000 7.8
Alfa Romeo Giulia Blue 147500 2017 Automatic Diesel 15000 7.6
KIA Cerato Blue 145600 2007 Manual Diesel 2400 5.3
Jeep Compass Gray 206000 2008 Manual Diesel 4500 6.8
Ford Fiesta Black 174000 2014 Manual Diesel 6000 7.1
Audi Q7 Black 189000 2016 Automatic Diesel 29700 9.1
Alfa Romeo GT Blue 210000 2008 Manual Diesel 3000 5.1
Opel Tigra Red 147000 2005 Manual Gasoline 4500 7.3
Mercedes-Benz A White 148000 2011 Automatic Gasoline 5200 7.4
Land Rover Range Rover Gray 272000 2008 Automatic Diesel 8400 7.9
Audi S8 White 202000 2008 Automatic Gasoline 11200 8.1
Volkswagen Touareg Blue 238350 2004 Automatic GPL 3900 6.75
Mazda CX-7 White 230000 2007 Manual GPL 4200 7.3
Suzuki Vitara Gray 61247 2007 Manual Gasoline 17300 8.9
Hyundai I30 Black 191000 2012 Manual Gasoline 4600 6.7
BMW M3 Blue 90000 2009 Automatic Gasoline 14500 8.0
Buy
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
3:53 PM
1/4/2021


Da o nota
localhost:8080/cars/update
Da o nota
Company Model Color Rolled Kilometers Year of Fabrication Gearbox Fuel Price Grade
Audi A6 Black 200000 2012 Automatic Diesel 13500 8.899999999999999
Citroen C8 red 273000 2003 Manual Diesel 2000 5.9
Dacia Duster Green 171400 2012 Manual Gasoline 7790 7.2
Chevrolet Caprice Black 75000 1984 Automatic GPL 24000 9.3
BMW X5 Black 240000 2015 Automatic Diesel 25190 8.5
Fiat Coupe Yellow 229000 1994 Manual Gasoline 13000 7.8
Alfa Romeo Giulia Blue 147500 2017 Automatic Diesel 15000 7.6
KIA Cerato Blue 145600 2007 Manual Diesel 2400 5.3
Jeep Compass Gray 206000 2008 Manual Diesel 4500 6.8
Ford Fiesta Black 174000 2014 Manual Diesel 6000 7.1
Audi Q7 Black 189000 2016 Automatic Diesel 29700 9.1
Alfa Romeo GT Blue 210000 2008 Manual Diesel 3000 5.1
Opel Tigra Red 147000 2005 Manual Gasoline 4500 7.3
Mercedes-Benz A White 148000 2011 Automatic Gasoline 5200 7.4
Land Rover Range Rover Gray 272000 2008 Automatic Diesel 8400 7.9
Audi S8 White 202000 2008 Automatic Gasoline 11200 8.1
Volkswagen Touareg Blue 238350 2004 Automatic GPL 3900 6.75
Mazda CX-7 White 230000 2007 Manual GPL 4200 7.3
Suzuki Vitara Gray 61247 2007 Manual Gasoline 17300 8.9
Hyundai I30 Black 191000 2012 Manual Gasoline 4600 6.7
BMW M3 Blue 90000 2009 Automatic Gasoline 14500 8.0
Grade 6.7
Grade
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
3:53 PM
1/4/2021


All cars
localhost:8080/cars/index
Company Model Color Rolled Kilometers Year of Fabrication Gearbox Fuel Price Grade
Audi A6 Black 200000 2012 Automatic Diesel 13500 8.899999999999999
Citroen C8 red 273000 2003 Manual Diesel 2000 5.9
Dacia Duster Green 171400 2012 Manual Gasoline 7790 7.2
Chevrolet Caprice Black 75000 1984 Automatic GPL 24000 9.3
BMW X5 Black 240000 2015 Automatic Diesel 25190 8.5
Fiat Coupe Yellow 229000 1994 Manual Gasoline 13000 7.8
Alfa Romeo Giulia Blue 147500 2017 Automatic Diesel 15000 7.6
KIA Cerato Blue 145600 2007 Manual Diesel 2400 5.3
Jeep Compass Gray 206000 2008 Manual Diesel 4500 6.8
Ford Fiesta Black 174000 2014 Manual Diesel 6000 7.1
Audi Q7 Black 189000 2016 Automatic Diesel 29700 9.1
Alfa Romeo GT Blue 210000 2008 Manual Diesel 3000 5.1
Opel Tigra Red 147000 2005 Manual Gasoline 4500 7.3
Mercedes-Benz A White 148000 2011 Automatic Gasoline 5200 7.4
Land Rover Range Rover Gray 272000 2008 Automatic Diesel 8400 7.9
Audi S8 White 202000 2008 Automatic Gasoline 11200 8.1
Volkswagen Touareg Blue 238350 2004 Automatic GPL 3900 6.75
Mazda CX-7 White 230000 2007 Manual GPL 4200 7.3
Suzuki Vitara Gray 61247 2007 Manual Gasoline 17300 8.9
Hyundai I30 Black 191000 2012 Manual Gasoline 4600 6.7
BMW M3 Blue 90000 2009 Automatic Gasoline 14500 8.0
Activate Windows
Go to Settings to activate Windows.
Type here to search
ENG
3:54 PM
1/4/2021

C. Bibliografie

1. https://docs.spring.io/spring-framework/docs/4.3.15.RELEASE/spring-framework-reference/html/aop.html
2. https://www.youtube.com/watch?v=Og9Fyew8ltQ&t=543s
3. https://www.youtube.com/watch?v=3Wby8Dax4Vc
4. https://stackoverflow.com/
5. http://www.drogoreanu.ro/tutorials/html.php
6. https://www.w3schools.com/html/
7. https://www.jetbrains.com/help/idea/aspectj.html
8. http://www.cs.ubbcluj.ro/~gabis/DocDiplome/SistemeDeRecomandare/Recommender_systems_handbook.pdf