**Задание.**

Реализовать CRUD-операции по выданной теме с веб-интерфейсом на основе Hibernate JTA + СУБД MySQL согласно концепции DB First.

Требования:

* Разделение представления, контроллеров и модели
* Реализация Data Access Object – класса
* Использование интерфейса DAO или Data Source

Рекомендации:

* Реализовать сначала модель
* Протестировать модель на юнит-тестах (например)
* Реализовать сервлеты и view (рекомендуется использовать bootstrap)
* См. пример с урока от 10.10.2022

**Варианты заданий.**

1. Бабунин А. А. – Библиотечный каталог

2. Боровушкин Д. И. – Каталог виниловых пластинок

3. Гущин Л. П. – Художественная галерея

4. Дурнев А. А. – Фермерский автопарк

5. Исоков А. А. – Каталог переведенных копий книг

6. Кузнецов А. К – Учет животных в зоопарке

7. Курлыгин Р. С. – Студенческие списки поступления

8. Курлыгина В. С. – Справочник ВВП разных стран

9. Майдан М. К. – Каталог горных вершин мира

10. Найденов А. Е. – Справочник по микропроцессорам

11. Пестова О. Ю. – Учетная книга денежных операций

12. Поминов В. С. – Каталог автомобилей

13. Скуратов Н. С. – Модельный ряд мирового авиапарка транспортной авиации

14. Тымкив А. С. – Список курортов мира

Auto Antity

package CarEntity;  
  
import jakarta.persistence.\*;  
  
@Entity  
@Table(name = "auto", schema = "db\_auto", catalog = "")  
public class AutoEntity {  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 @Id  
 @Column(name = "id")  
 private Long id;  
 @Basic  
 @Column(name = "manufacturer")  
 private String manufacturer;  
 @Basic  
 @Column(name = "model")  
 private String model;  
 @Basic  
 @Column(name = "year")  
 private Integer year;  
 @Basic  
 @Column(name = "price")  
 private Double price;  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getManufacturer() {  
 return manufacturer;  
 }  
  
 public void setManufacturer(String manufacturer) {  
 this.manufacturer = manufacturer;  
 }  
  
 public String getModel() {  
 return model;  
 }  
  
 public void setModel(String model) {  
 this.model = model;  
 }  
  
 public Integer getYear() {  
 return year;  
 }  
  
 public void setYear(Integer year) {  
 this.year = year;  
 }  
  
 public Double getPrice() {  
 return price;  
 }  
  
 public void setPrice(Double price) {  
 this.price = price;  
 }  
  
 public void updateFields(AutoEntity src) {  
 manufacturer = src.manufacturer;  
 model = src.model;  
 year = src.year;  
 price = src.price;  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
  
 AutoEntity that = (AutoEntity) o;  
  
 if (id != null ? !id.equals(that.id) : that.id != null) return false;  
 if (manufacturer != null ? !manufacturer.equals(that.manufacturer) : that.manufacturer != null) return false;  
 if (model != null ? !model.equals(that.model) : that.model != null) return false;  
 if (year != null ? !year.equals(that.year) : that.year != null) return false;  
 if (price != null ? !price.equals(that.price) : that.price != null) return false;  
  
 return true;  
 }  
  
 @Override  
 public int hashCode() {  
 int result = id != null ? id.hashCode() : 0;  
 result = 31 \* result + (manufacturer != null ? manufacturer.hashCode() : 0);  
 result = 31 \* result + (model != null ? model.hashCode() : 0);  
 result = 31 \* result + (year != null ? year.hashCode() : 0);  
 result = 31 \* result + (price != null ? price.hashCode() : 0);  
 return result;  
 }  
}

Control Servlet

package com.example.farm\_vehicle\_crud\_hibernate;  
  
import CarEntity.AutoEntity;  
import DAO.CarDAO;  
import jakarta.servlet.\*;  
import jakarta.servlet.http.\*;  
import jakarta.servlet.annotation.\*;  
  
import java.io.IOException;  
import java.sql.SQLException;  
import java.util.List;  
  
@WebServlet("/")  
public class ControlServlet extends HttpServlet {  
 private static final long *serialVersionUID* = 1L;  
 private CarDAO carDAO;  
  
 @Override  
 public void init() {  
 carDAO = new CarDAO();  
  
 }  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response)  
 throws ServletException, IOException {  
 String action = request.getServletPath();  
 try {  
 switch (action) {  
 case "/new":  
 showNewForm(request, response);  
 break;  
 case "/insert":  
 insertCar(request, response);  
 break;  
 case "/delete":  
 deleteCar(request, response);  
 break;  
 case "/edit":  
 showEditForm(request, response);  
 break;  
 case "/update":  
 updateCar(request, response);  
 break;  
 default:  
 listCars(request, response);  
 break;  
 }  
 } catch (SQLException ex) {  
 throw new ServletException(ex);  
 }  
 }  
  
 @Override  
 protected void doPost(HttpServletRequest request, HttpServletResponse response)  
 throws ServletException, IOException {  
 doGet(request, response);  
 }  
  
 private void listCars(HttpServletRequest request, HttpServletResponse response)  
 throws SQLException, IOException, ServletException {  
 List<AutoEntity> listCars = carDAO.getAllAuto();  
 request.setAttribute("listCars", listCars);  
 RequestDispatcher dispatcher = request.getRequestDispatcher("CarList.jsp");  
 dispatcher.forward(request, response);  
 }  
  
 private void showNewForm(HttpServletRequest request, HttpServletResponse response)  
 throws ServletException, IOException {  
 RequestDispatcher dispatcher = request.getRequestDispatcher("CarForm.jsp");  
 dispatcher.forward(request, response);  
 }  
  
 private void showEditForm(HttpServletRequest request, HttpServletResponse response)  
 throws SQLException, ServletException, IOException {  
// int id = Integer.parseInt(request.getParameter("id"));  
 Long id = Long.*parseLong*(request.getParameter("id"));  
 AutoEntity existingCar = carDAO.getAutoById(id);  
 RequestDispatcher dispatcher = request.getRequestDispatcher("CarForm.jsp");  
 request.setAttribute("car", existingCar);  
 dispatcher.forward(request, response);  
 }  
  
 private void insertCar(HttpServletRequest request, HttpServletResponse response)  
 throws SQLException, IOException {  
 String manufacturer = request.getParameter("nameCar");  
 String model = request.getParameter("modelF");  
 Integer year = Integer.*valueOf*(request.getParameter("yearF"));  
 Double price = Double.*valueOf*(request.getParameter("priceF"));  
  
 AutoEntity newOrder = new AutoEntity();  
 newOrder.setManufacturer(manufacturer);  
 newOrder.setModel(model);  
 newOrder.setYear(year);  
 newOrder.setPrice(price);  
 carDAO.insertAuto(newOrder);  
 response.sendRedirect("list");  
 }  
  
 private void updateCar(HttpServletRequest request, HttpServletResponse response)  
 throws SQLException, IOException {  
 Long id = Long.*parseLong*(request.getParameter("id"));  
 String manufacturer = request.getParameter("nameCar");  
 String model = request.getParameter("modelF");  
 Integer year = Integer.*valueOf*(request.getParameter("yearF"));  
 Double price = Double.*valueOf*(request.getParameter("priceF"));  
  
 AutoEntity newOrder = carDAO.getAutoById(id);  
 newOrder.setManufacturer(manufacturer);  
 newOrder.setModel(model);  
 newOrder.setYear(year);  
 newOrder.setPrice(price);  
 carDAO.updateAuto(newOrder);  
 response.sendRedirect("list");  
 }  
  
 private void deleteCar(HttpServletRequest request, HttpServletResponse response)  
 throws SQLException, IOException {  
 Long id = Long.*parseLong*(request.getParameter("id"));  
 carDAO.deleteAutoById(id);  
 response.sendRedirect("list");  
 }  
}

CarDAO

package DAO;  
  
import CarEntity.AutoEntity;  
import jakarta.persistence.EntityManager;  
import jakarta.persistence.EntityManagerFactory;  
import jakarta.persistence.EntityTransaction;  
import jakarta.persistence.Persistence;  
  
import java.util.List;  
  
public class CarDAO implements IControlOrder {  
  
 @Override  
 public void insertAuto(AutoEntity order) {  
 // 1. создаем фабрику  
 EntityManagerFactory entityManagerFactory = Persistence.*createEntityManagerFactory*("default");  
 // 2. manager  
 EntityManager entityManager = entityManagerFactory.createEntityManager();  
 // 3. объект транзакции  
 EntityTransaction transaction = entityManager.getTransaction();  
 try {  
 // выполнение самой операции создания записи в БД  
 transaction.begin();  
 entityManager.persist(order); // создание новой записи на основе объекта  
 transaction.commit();  
 } finally {  
 // все закрыть и откатить транзакцию, если нужно  
 if (transaction.isActive()) {  
 transaction.rollback();  
 }  
 entityManager.close();  
 entityManagerFactory.close();  
 }  
 }  
  
 @Override  
 public AutoEntity getAutoById(Long id) {  
 EntityManagerFactory entityManagerFactory = Persistence.*createEntityManagerFactory*("default");  
 EntityManager entityManager = entityManagerFactory.createEntityManager();  
 EntityTransaction transaction = entityManager.getTransaction();  
 // переменная для хранения объекта-результата поиска  
 AutoEntity result = null;  
 try {  
 transaction.begin();  
 // сама операция  
 result = entityManager.find(AutoEntity.class, id);  
 transaction.commit();  
 } finally {  
 if (transaction.isActive()) {  
 transaction.rollback();  
 }  
 entityManager.close();  
 entityManagerFactory.close();  
 }  
 return result; // вернуть результат  
 }  
  
 @Override  
 public List<AutoEntity> getAllAuto() {  
 EntityManagerFactory entityManagerFactory = Persistence.*createEntityManagerFactory*("default");  
 EntityManager entityManager = entityManagerFactory.createEntityManager();  
 EntityTransaction transaction = entityManager.getTransaction();  
 // список результатов  
 List<AutoEntity> auto = null;  
  
 try {  
 transaction.begin();  
 // сама операция  
 auto = entityManager.createQuery("SELECT e FROM AutoEntity e").getResultList();  
 transaction.commit();  
 } finally {  
 if (transaction.isActive()) {  
 transaction.rollback();  
 }  
 entityManager.close();  
 entityManagerFactory.close();  
 }  
 return auto;  
 }  
  
 @Override  
 public void updateAuto(AutoEntity car) {  
 EntityManagerFactory entityManagerFactory = Persistence.*createEntityManagerFactory*("default");  
 EntityManager entityManager = entityManagerFactory.createEntityManager();  
 EntityTransaction transaction = entityManager.getTransaction();  
 try {  
 transaction.begin();  
 // 1. получаю обновляемый объект по Id  
 AutoEntity updated = entityManager.find(AutoEntity.class, car.getId());  
 // 2. обновляю поля обновляемого объекта  
 updated.updateFields(car);  
 transaction.commit();  
 } finally {  
 if (transaction.isActive()) {  
 transaction.rollback();  
 }  
 entityManager.close();  
 entityManagerFactory.close();  
 }  
 }  
  
 @Override  
 public void deleteAutoById(Long id) {  
 EntityManagerFactory entityManagerFactory = Persistence.*createEntityManagerFactory*("default");  
 EntityManager entityManager = entityManagerFactory.createEntityManager();  
 EntityTransaction transaction = entityManager.getTransaction();  
  
 try {  
 transaction.begin();  
 // сама операция  
 // 1. получить удаляемый объект  
 AutoEntity deleted = entityManager.find(AutoEntity.class, id);  
 // 2. удалить  
 entityManager.remove(deleted);  
 transaction.commit();  
 } finally {  
 if (transaction.isActive()) {  
 transaction.rollback();  
 }  
 entityManager.close();  
 entityManagerFactory.close();  
 }  
 }  
}

IController

package DAO;  
  
import CarEntity.AutoEntity;  
  
import java.util.List;  
  
public interface IControlOrder {  
 void insertAuto(AutoEntity order); // добавить  
 AutoEntity getAutoById(Long id); // получить по id  
  
 List<AutoEntity> getAllAuto(); // получить все  
  
 void updateAuto(AutoEntity car); // обновить данные  
  
 void deleteAutoById(Long id); // удалить по id  
}

CarDAO Test

package DAO;  
  
import CarEntity.AutoEntity;  
import org.junit.jupiter.api.Test;  
  
import java.util.List;  
  
import static org.junit.jupiter.api.Assertions.\*;  
  
class CarDAOTest {  
  
 @Test  
 void insertAuto() {  
 AutoEntity car = new AutoEntity();  
 car.setManufacturer("Шишига");  
 car.setModel("133");  
 car.setYear(1990);  
 car.setPrice(77000.0);  
 }  
  
 @Test  
 void getAutoById() {  
 AutoEntity result = new CarDAO().getAutoById(13L);  
 System.*out*.println(result);  
 }  
  
 @Test  
 void getAllAuto() {  
 List<AutoEntity> cars = new CarDAO().getAllAuto();  
 for (AutoEntity car : cars) {  
 System.*out*.println(car);  
 }  
 }  
  
 @Test  
 void updateAuto() {  
 AutoEntity result = new CarDAO().getAutoById(13L);  
 result.setYear(2000);  
 System.*out*.println(result);  
 }  
  
 @Test  
 void deleteAutoById() {  
 new CarDAO().deleteAutoById(21L);  
 }  
}

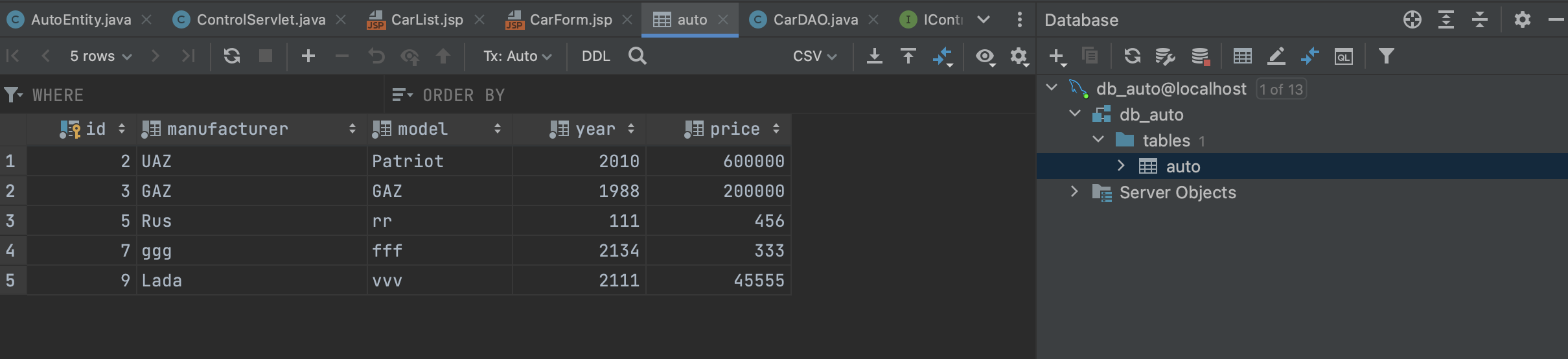
Carlist.jsp

<%--  
 Created by IntelliJ IDEA.  
 User: aleksandr  
 Date: 16.10.2022  
 Time: 13:35  
 To change this template use File | Settings | File Templates.  
--%>  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ **page** contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <title>Каталог автомобилей</title>  
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet"  
 integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">  
 <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/umd/popper.min.js"  
 integrity="sha384-IQsoLXl5PILFhosVNubq5LC7Qb9DXgDA9i+tQ8Zj3iwWAwPtgFTxbJ8NT4GN1R8p"  
 crossorigin="anonymous"></script>  
 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.min.js"  
 integrity="sha384-cVKIPhGWiC2Al4u+LWgxfKTRIcfu0JTxR+EQDz/bgldoEyl4H0zUF0QKbrJ0EcQF"  
 crossorigin="anonymous"></script>  
</head>  
<body>  
<header>  
<%-- <nav class="navbar navbar-expand-md navbar-dark"--%>  
<%-- style="background-color: gray">--%>  
<%-- <div>--%>  
<%-- <a href="<%=request.getContextPath()%>/list" class="navbar-brand">На главную</a>--%>  
<%-- </div>--%>  
<%-- </nav>--%>  
</header>  
<br>  
  
<div class="row">  
  
 <div class="container">  
 <h3 class="text-center">Каталог фермерских автомобилей</h3>  
<%-- <div class="container text-left" style="text-align: center">--%>  
<%-- <a href="<%=request.getContextPath()%>/new" class="btn btn-success">Добавить авто в каталог</a>--%>  
<%-- </div>--%>  
 <table class="table table-bordered">  
 <thead>  
 <tr>  
 <th>ID</th>  
 <th>Производитель</th>  
 <th>Модель</th>  
 <th>Год выпуска</th>  
 <th>Цена</th>  
 <th>Выберите действие</th>  
 </tr>  
 </thead>  
 <tbody>  
  
 <**c:forEach** var="car" items="${listCars}">  
  
 <tr>  
 <td><**c:out** value="${car.id}"/></td>  
 <td><**c:out** value="${car.manufacturer}"/></td>  
 <td><**c:out** value="${car.model}"/></td>  
 <td><**c:out** value="${car.year}"/></td>  
 <td><**c:out** value="${car.price}"/></td>  
 <td><a href="edit?id=<**c:out** value='${car.id}' />">Изменить</a>  
 &nbsp;&nbsp;&nbsp;&nbsp; <a  
 href="delete?id=<**c:out** value='${car.id}' />">Удалить</a></td>  
 </tr>  
 </**c:forEach**>  
 </tbody>  
 </table>  
 </div>  
</div>  
<div class="container text-left" style="text-align: center">  
  
 <a href="<%=request.getContextPath()%>/new" class="btn btn-outline-success">Добавить авто в каталог</a>  
</div>  
</body>  
</html>

CarForm.jsp

<%--  
 Created by IntelliJ IDEA.  
 User: aleksandr  
 Date: 16.10.2022  
 Time: 13:32  
 To change this template use File | Settings | File Templates.  
--%>  
  
<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<%@ **page** contentType="text/html;charset=UTF-8" language="java" %>  
<html>  
<head>  
 <title>Каталог автомобилей</title>  
 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet"  
 integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">  
 <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/umd/popper.min.js"  
 integrity="sha384-IQsoLXl5PILFhosVNubq5LC7Qb9DXgDA9i+tQ8Zj3iwWAwPtgFTxbJ8NT4GN1R8p"  
 crossorigin="anonymous"></script>  
 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.min.js"  
 integrity="sha384-cVKIPhGWiC2Al4u+LWgxfKTRIcfu0JTxR+EQDz/bgldoEyl4H0zUF0QKbrJ0EcQF"  
 crossorigin="anonymous"></script>  
</head>  
<body>  
<header>  
 <nav class="navbar navbar-expand-md navbar-dark"  
 style="background-color: gray">  
 <div>  
 <a href="<%=request.getContextPath()%>/list" class="navbar-brand">Главная</a>  
 </div>  
 </nav>  
</header>  
<br>  
<div class="container col-md-5">  
 <div class="card">  
 <div class="card-body">  
 <**c:if** test="${car != null}">  
 <form action="update" method="post">  
 </**c:if**>  
 <**c:if** test="${car == null}">  
 <form action="insert" method="post">  
 </**c:if**>  
  
 <caption>  
 <h2>  
 <**c:if** test="${car != null}">  
 Изменить данные  
 </**c:if**>  
 <**c:if** test="${car == null}">  
 Добавить автомобиль  
 </**c:if**>  
 </h2>  
 </caption>  
  
 <**c:if** test="${car != null}">  
 <input type="hidden" name="id" value="<**c:out** value='${car.id}' />"/>  
 </**c:if**>  
  
 <fieldset class="form-group">  
 <label>Производитель</label> <input type="text"  
 value="<**c:out** value='${car.manufacturer}' />"  
 class="form-control"  
 name="nameCar" required="required">  
 </fieldset>  
  
 <fieldset class="form-group">  
 <label>Модель</label> <input type="text"  
 value="<**c:out** value='${car.model}' />" class="form-control"  
 name="modelF">  
 </fieldset>  
  
 <fieldset class="form-group">  
 <label>Год выпуска</label> <input type="text"  
 value="<**c:out** value='${car.year}' />" class="form-control"  
 name="yearF">  
 </fieldset>  
 <fieldset class="form-group">  
 <label>Цена</label> <input type="text"  
 value="<**c:out** value='${car.price}' />" class="form-control"  
 name="priceF">  
 </fieldset>  
  
 <button type="submit" class="btn btn-outline-success">Сохранить</button>  
 </form>  
 </div>  
 </div>  
</div>  
</body>  
</html>

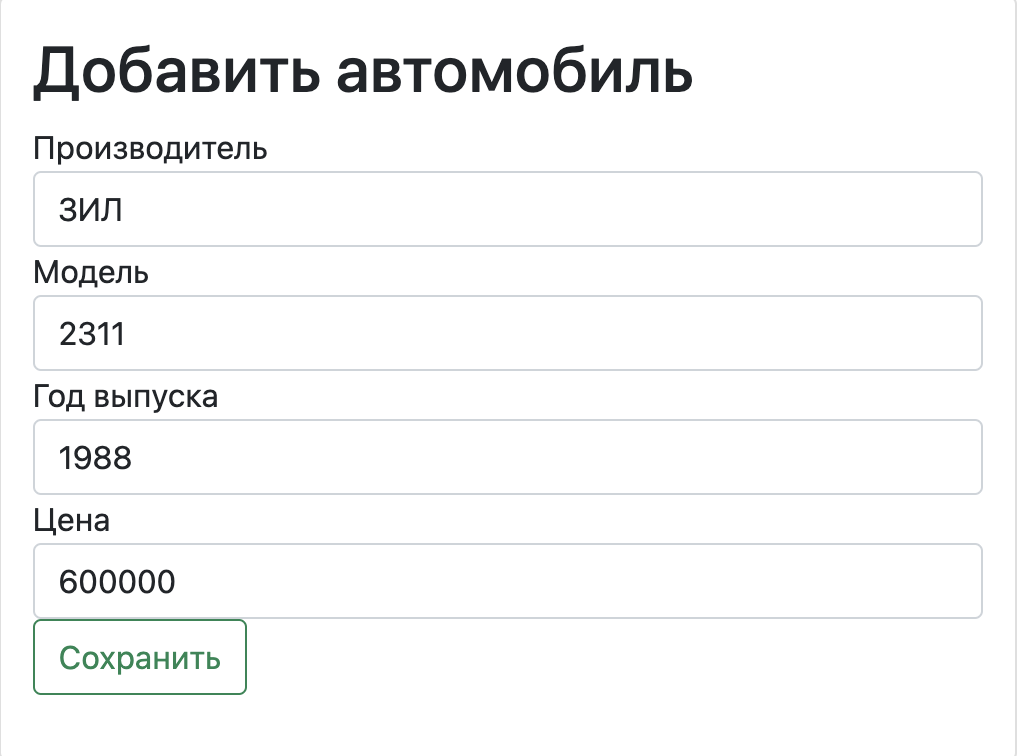
Table

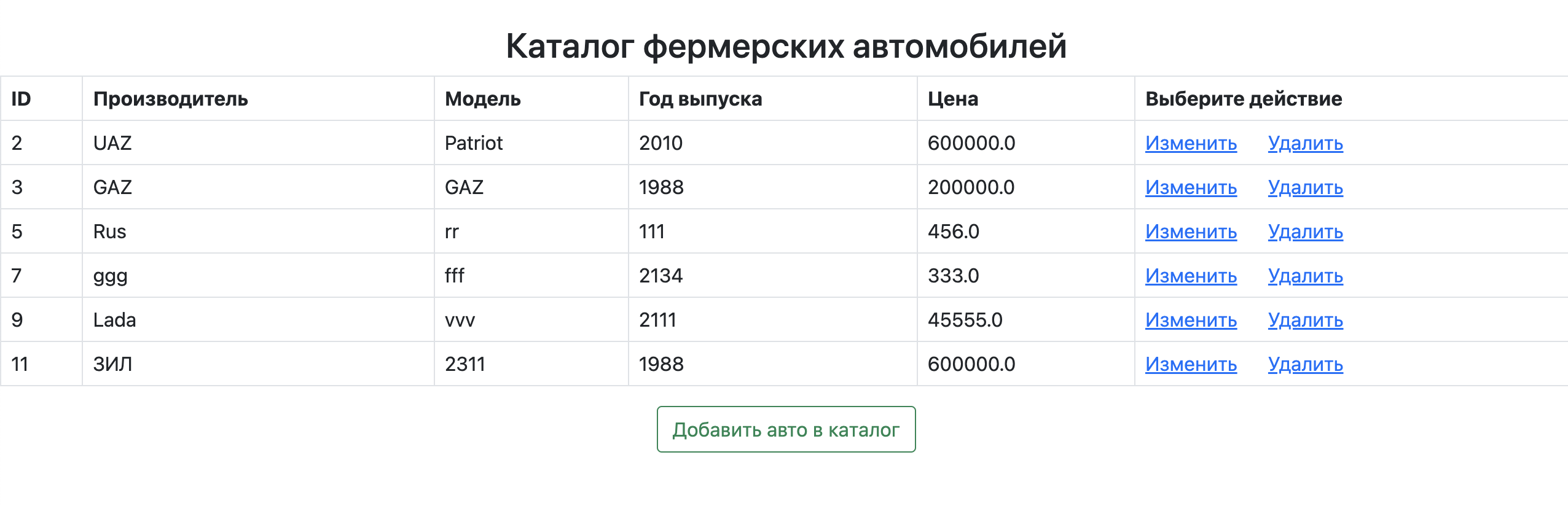


Test

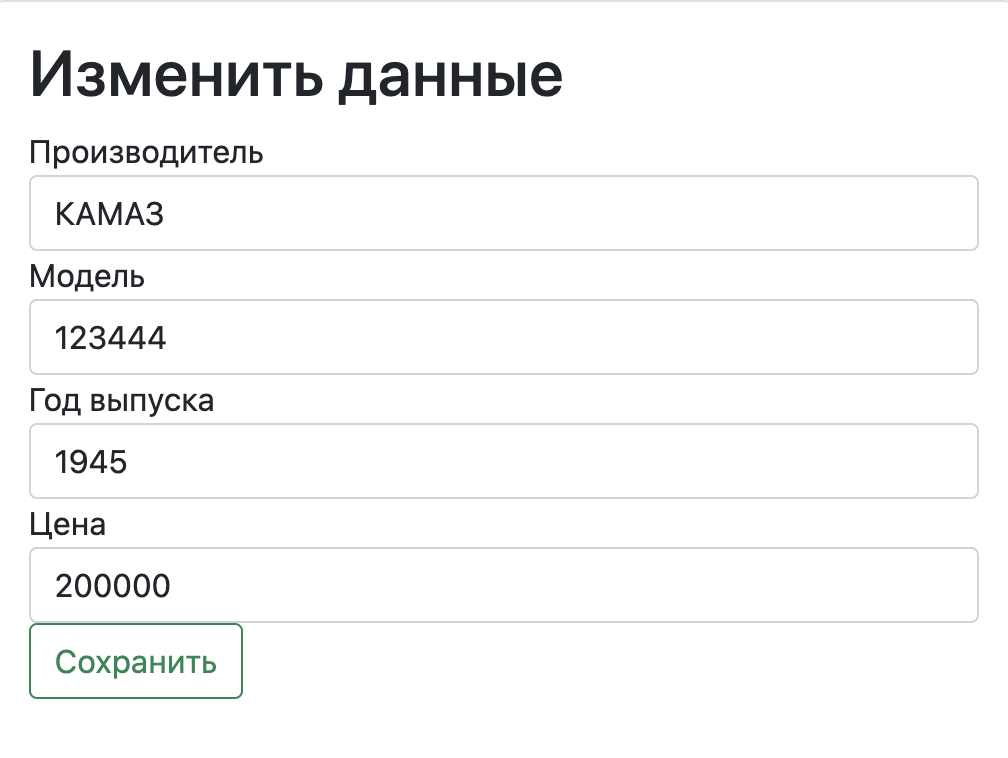


Add new car





Change Parameters





Delete

