Министерство образования и науки Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение науки и высшего образования**

«Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

**Институт информационных технологий и радиотехники (ИИТР)**

**Лабораторная работа № 3**

**по дисциплине**

**«Распределённые программные системы»**

Выполнила:

ст. гр. ИСТ-117

Шумейко Д.С.

Принял:

Макаров С.А.

Владимир, 2020

ЦЕЛЬ РАБОТЫ

Освоить создание классов сущностей, в том числе путем обратного проектирования реляционной базы данных средствами NetBeans 5.5 и работу с менеджером сущностей, и получить навыки работы со средой разработки.

Ссылка на репозиторий с лабораторными работами:

<https://github.com/daryashumeyko/rps_6semestr>

ВЫПОЛНЕНИЕ РАБОТЫ

Класс сущности – Person:

package Entity;  
  
import java.io.Serializable;  
import javax.persistence.Basic;  
import javax.persistence.Column;  
import javax.persistence.Entity;  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;  
import javax.persistence.JoinColumn;  
import javax.persistence.ManyToOne;  
import javax.persistence.NamedQueries;  
import javax.persistence.NamedQuery;  
import javax.persistence.Table;  
import javax.validation.constraints.NotNull;  
import javax.validation.constraints.Size;  
  
@Entity  
@Table(  
 name = "person"  
)  
@NamedQueries({@NamedQuery(  
 name = "Person.findAll",  
 query = "SELECT p FROM Person p"  
), @NamedQuery(  
 name = "Person.findById",  
 query = "SELECT p FROM Person p WHERE p.id = :id"  
), @NamedQuery(  
 name = "Person.findByFirstName",  
 query = "SELECT p FROM Person p WHERE p.firstName = :firstName"  
), @NamedQuery(  
 name = "Person.findByLastName",  
 query = "SELECT p FROM Person p WHERE p.lastName = :lastName"  
)})  
public class Person implements Serializable {  
 private static final long serialVersionUID = 1L;  
 @Id  
 @Basic(  
 optional = false  
 )  
 @NotNull  
 @Column(  
 name = "id"  
 )  
 @GeneratedValue(  
 strategy = GenerationType.AUTO  
 )  
 private Long id;  
 @Size(  
 max = 255  
 )  
 @Column(  
 name = "first\_name"  
 )  
 private String firstName;  
 @Size(  
 max = 255  
 )  
 @Column(  
 name = "last\_name"  
 )  
 private String lastName;  
 @JoinColumn(  
 name = "driving\_license\_id",  
 referencedColumnName = "id"  
 )  
 @ManyToOne  
 private DrivingLicense drivingLicenseId;  
  
 public Person() {  
 }  
  
 public Person(Long id) {  
 this.id = id;  
 }  
  
 public Long getId() {  
 return this.id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getFirstName() {  
 return this.firstName;  
 }  
  
 public void setFirstName(String firstName) {  
 this.firstName = firstName;  
 }  
  
 public String getLastName() {  
 return this.lastName;  
 }  
  
 public void setLastName(String lastName) {  
 this.lastName = lastName;  
 }  
  
 public DrivingLicense getDrivingLicenseId() {  
 return this.drivingLicenseId;  
 }  
  
 public void setDrivingLicenseId(DrivingLicense drivingLicenseId) {  
 this.drivingLicenseId = drivingLicenseId;  
 }  
  
 public int hashCode() {  
 int hash = 0;  
 int hash = hash + (this.id != null ? this.id.hashCode() : 0);  
 return hash;  
 }  
  
 public boolean equals(Object object) {  
 if (!(object instanceof Person)) {  
 return false;  
 } else {  
 Person other = (Person)object;  
 return (this.id != null || other.id == null) && (this.id == null || this.id.equals(other.id));  
 }  
 }  
  
 public String toString() {  
 return "Entity.Person[ id=" + this.id + " ]";  
 }  
}

Класс сущности – DrivingLicence:

package Entity;  
  
import java.io.Serializable;  
import java.util.Collection;  
import java.util.Date;  
import javax.persistence.Basic;  
import javax.persistence.Column;  
import javax.persistence.Entity;  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;  
import javax.persistence.ManyToMany;  
import javax.persistence.NamedQueries;  
import javax.persistence.NamedQuery;  
import javax.persistence.OneToMany;  
import javax.persistence.Table;  
import javax.persistence.Temporal;  
import javax.persistence.TemporalType;  
import javax.validation.constraints.NotNull;  
import javax.xml.bind.annotation.XmlTransient;  
  
@Entity  
@Table(  
 name = "driving\_license"  
)  
@NamedQueries({@NamedQuery(  
 name = "DrivingLicense.findAll",  
 query = "SELECT d FROM DrivingLicense d"  
), @NamedQuery(  
 name = "DrivingLicense.findById",  
 query = "SELECT d FROM DrivingLicense d WHERE d.id = :id"  
), @NamedQuery(  
 name = "DrivingLicense.findByExpireDate",  
 query = "SELECT d FROM DrivingLicense d WHERE d.expireDate = :expireDate"  
), @NamedQuery(  
 name = "DrivingLicense.findByIssueDate",  
 query = "SELECT d FROM DrivingLicense d WHERE d.issueDate = :issueDate"  
)})  
public class DrivingLicense implements Serializable {  
 private static final long serialVersionUID = 1L;  
 @Id  
 @Basic(  
 optional = false  
 )  
 @NotNull  
 @Column(  
 name = "id"  
 )  
 @GeneratedValue(  
 strategy = GenerationType.AUTO  
 )  
 private Long id;  
 @Column(  
 name = "expire\_date"  
 )  
 @Temporal(TemporalType.DATE)  
 private Date expireDate;  
 @Column(  
 name = "issue\_date"  
 )  
 @Temporal(TemporalType.DATE)  
 private Date issueDate;  
 @ManyToMany(  
 mappedBy = "drivingLicenseCollection"  
 )  
 private Collection<DrivingLicenseCategory> drivingLicenseCategoryCollection;  
 @OneToMany(  
 mappedBy = "drivingLicenseId"  
 )  
 private Collection<Person> personCollection;  
  
 public DrivingLicense() {  
 }  
  
 public DrivingLicense(Long id) {  
 this.id = id;  
 }  
  
 public Long getId() {  
 return this.id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public Date getExpireDate() {  
 return this.expireDate;  
 }  
  
 public void setExpireDate(Date expireDate) {  
 this.expireDate = expireDate;  
 }  
  
 public Date getIssueDate() {  
 return this.issueDate;  
 }  
  
 public void setIssueDate(Date issueDate) {  
 this.issueDate = issueDate;  
 }  
  
 @XmlTransient  
 public Collection<DrivingLicenseCategory> getDrivingLicenseCategoryCollection() {  
 return this.drivingLicenseCategoryCollection;  
 }  
  
 public void setDrivingLicenseCategoryCollection(Collection<DrivingLicenseCategory> drivingLicenseCategoryCollection) {  
 this.drivingLicenseCategoryCollection = drivingLicenseCategoryCollection;  
 }  
  
 @XmlTransient  
 public Collection<Person> getPersonCollection() {  
 return this.personCollection;  
 }  
  
 public void setPersonCollection(Collection<Person> personCollection) {  
 this.personCollection = personCollection;  
 }  
  
 public int hashCode() {  
 int hash = 0;  
 int hash = hash + (this.id != null ? this.id.hashCode() : 0);  
 return hash;  
 }  
  
 public boolean equals(Object object) {  
 if (!(object instanceof DrivingLicense)) {  
 return false;  
 } else {  
 DrivingLicense other = (DrivingLicense)object;  
 return (this.id != null || other.id == null) && (this.id == null || this.id.equals(other.id));  
 }  
 }  
  
 public String toString() {  
 return "Entity.DrivingLicense[ id=" + this.id + " ]";  
 }  
}

Класс сущности – DrivingLicenceCategory:

package Entity;  
  
import java.io.Serializable;  
import java.util.Collection;  
import javax.persistence.Basic;  
import javax.persistence.Column;  
import javax.persistence.Entity;  
import javax.persistence.GeneratedValue;  
import javax.persistence.GenerationType;  
import javax.persistence.Id;  
import javax.persistence.JoinColumn;  
import javax.persistence.JoinTable;  
import javax.persistence.ManyToMany;  
import javax.persistence.NamedQueries;  
import javax.persistence.NamedQuery;  
import javax.persistence.Table;  
import javax.validation.constraints.NotNull;  
import javax.validation.constraints.Size;  
import javax.xml.bind.annotation.XmlTransient;  
  
@Entity  
@Table(  
 name = "driving\_license\_category"  
)  
@NamedQueries({@NamedQuery(  
 name = "DrivingLicenseCategory.findAll",  
 query = "SELECT d FROM DrivingLicenseCategory d"  
), @NamedQuery(  
 name = "DrivingLicenseCategory.findById",  
 query = "SELECT d FROM DrivingLicenseCategory d WHERE d.id = :id"  
), @NamedQuery(  
 name = "DrivingLicenseCategory.findByCategoryName",  
 query = "SELECT d FROM DrivingLicenseCategory d WHERE d.categoryName = :categoryName"  
)})  
public class DrivingLicenseCategory implements Serializable {  
 private static final long serialVersionUID = 1L;  
 @Id  
 @Basic(  
 optional = false  
 )  
 @NotNull  
 @Column(  
 name = "id"  
 )  
 @GeneratedValue(  
 strategy = GenerationType.AUTO  
 )  
 private Long id;  
 @Size(  
 max = 255  
 )  
 @Column(  
 name = "category\_name"  
 )  
 private String categoryName;  
 @JoinTable(  
 name = "license\_category",  
 joinColumns = {@JoinColumn(  
 name = "category\_id",  
 referencedColumnName = "id"  
)},  
 inverseJoinColumns = {@JoinColumn(  
 name = "license\_id",  
 referencedColumnName = "id"  
)}  
 )  
 @ManyToMany  
 private Collection<DrivingLicense> drivingLicenseCollection;  
  
 public DrivingLicenseCategory() {  
 }  
  
 public DrivingLicenseCategory(Long id) {  
 this.id = id;  
 }  
  
 public Long getId() {  
 return this.id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getCategoryName() {  
 return this.categoryName;  
 }  
  
 public void setCategoryName(String categoryName) {  
 this.categoryName = categoryName;  
 }  
  
 @XmlTransient  
 public Collection<DrivingLicense> getDrivingLicenseCollection() {  
 return this.drivingLicenseCollection;  
 }  
  
 public void setDrivingLicenseCollection(Collection<DrivingLicense> drivingLicenseCollection) {  
 this.drivingLicenseCollection = drivingLicenseCollection;  
 }  
  
 public int hashCode() {  
 int hash = 0;  
 int hash = hash + (this.id != null ? this.id.hashCode() : 0);  
 return hash;  
 }  
  
 public boolean equals(Object object) {  
 if (!(object instanceof DrivingLicenseCategory)) {  
 return false;  
 } else {  
 DrivingLicenseCategory other = (DrivingLicenseCategory)object;  
 return (this.id != null || other.id == null) && (this.id == null || this.id.equals(other.id));  
 }  
 }  
  
 public String toString() {  
 return "Entity.DrivingLicenseCategory[ id=" + this.id + " ]";  
 }  
}

Локальный интерфейс компонента Person:

package Facade;  
  
import Entity.Person;  
import java.util.List;  
import javax.ejb.Local;  
  
@Local  
public interface PersonFacadeLocal {  
 void create(Person var1);  
  
 void edit(Person var1);  
  
 void remove(Person var1);  
  
 Person find(Object var1);  
  
 List<Person> findAll();  
  
 Person findById(Long var1);  
  
 List<Person> findRange(int[] var1);  
  
 int count();  
}

Локальный интерфейс компонента DrivingLicence:

package Facade;  
  
import Entity.DrivingLicense;  
import java.util.List;  
import javax.ejb.Local;  
  
@Local  
public interface DrivingLicenseFacadeLocal {  
 void create(DrivingLicense var1);  
  
 void edit(DrivingLicense var1);  
  
 void remove(DrivingLicense var1);  
  
 DrivingLicense find(Object var1);  
  
 List<DrivingLicense> findAll();  
  
 DrivingLicense findById(Long var1);  
  
 List<DrivingLicense> findRange(int[] var1);  
  
 int count();  
}

Локальный интерфейс компонента DrivingLicenceCategory:

package Facade;  
  
import Entity.DrivingLicenseCategory;  
import java.util.List;  
import javax.ejb.Local;  
  
@Local  
public interface DrivingLicenseCategoryFacadeLocal {  
 void create(DrivingLicenseCategory var1);  
  
 void edit(DrivingLicenseCategory var1);  
  
 void remove(DrivingLicenseCategory var1);  
  
 DrivingLicenseCategory find(Object var1);  
  
 List<DrivingLicenseCategory> findAll();  
  
 DrivingLicenseCategory findById(Long var1);  
  
 List<DrivingLicenseCategory> findRange(int[] var1);  
  
 int count();  
}

Сессионный компонент Person без сохранения состояния:

package Facade;  
  
import Entity.Person;  
import javax.ejb.Stateless;  
import javax.persistence.EntityManager;  
import javax.persistence.PersistenceContext;  
  
@Stateless  
public class PersonFacade extends AbstractFacade<Person> implements PersonFacadeLocal {  
 @PersistenceContext(  
 unitName = "Lab2-ejbPU"  
 )  
 private EntityManager em;  
  
 protected EntityManager getEntityManager() {  
 return this.em;  
 }  
  
 public PersonFacade() {  
 super(Person.class);  
 }  
  
 public Person findById(Long id) {  
 return (Person)this.em.find(Person.class, id);  
 }  
}

Сессионный компонент DrivingLicence без сохранения состояния:

package Facade;  
  
import Entity.DrivingLicense;  
import javax.ejb.Stateless;  
import javax.persistence.EntityManager;  
import javax.persistence.PersistenceContext;  
  
@Stateless  
public class DrivingLicenseFacade extends AbstractFacade<DrivingLicense> implements DrivingLicenseFacadeLocal {  
 @PersistenceContext(  
 unitName = "Lab2-ejbPU"  
 )  
 private EntityManager em;  
  
 protected EntityManager getEntityManager() {  
 return this.em;  
 }  
  
 public DrivingLicenseFacade() {  
 super(DrivingLicense.class);  
 }  
  
 public DrivingLicense findById(Long id) {  
 return (DrivingLicense)this.em.find(DrivingLicense.class, id);  
 }  
}

Сессионный компонент DrivingLicenceCategory без сохранения состояния:

package Facade;  
  
import Entity.DrivingLicenseCategory;  
import javax.ejb.Stateless;  
import javax.persistence.EntityManager;  
import javax.persistence.PersistenceContext;  
  
@Stateless  
public class DrivingLicenseCategoryFacade extends AbstractFacade<DrivingLicenseCategory> implements DrivingLicenseCategoryFacadeLocal {  
 @PersistenceContext(  
 unitName = "Lab2-ejbPU"  
 )  
 private EntityManager em;  
  
 protected EntityManager getEntityManager() {  
 return this.em;  
 }  
  
 public DrivingLicenseCategoryFacade() {  
 super(DrivingLicenseCategory.class);  
 }  
  
 public DrivingLicenseCategory findById(Long id) {  
 return (DrivingLicenseCategory)this.em.find(DrivingLicenseCategory.class, id);  
 }  
}

PersonBean.class:

package Beans;  
  
import Entity.Person;  
import Facade.DrivingLicenseFacadeLocal;  
import Facade.PersonFacadeLocal;  
import java.io.Serializable;  
import java.util.Collection;  
import javax.ejb.EJB;  
import javax.enterprise.context.RequestScoped;  
import javax.inject.Named;  
  
@Named("personBean")  
@RequestScoped  
public class PersonBean implements Serializable {  
 @EJB  
 PersonFacadeLocal personFacadeLocal;  
 @EJB  
 DrivingLicenseFacadeLocal drivingLicenseFacadeLocal;  
 private Person person = new Person();  
 private Long selectedDrivingLicense;  
  
 public PersonBean() {  
 }  
  
 public Person getPerson() {  
 return this.person;  
 }  
  
 public void setPerson(Person person) {  
 this.person = person;  
 }  
  
 public Long getSelectedDrivingLicense() {  
 return this.selectedDrivingLicense;  
 }  
  
 public void setSelectedDrivingLicense(Long selectedDrivingLicense) {  
 this.selectedDrivingLicense = selectedDrivingLicense;  
 }  
  
 public String create() {  
 this.person.setDrivingLicenseId(this.drivingLicenseFacadeLocal.findById(this.selectedDrivingLicense));  
 this.personFacadeLocal.create(this.person);  
 return "personList";  
 }  
  
 public String remove() {  
 this.person = this.personFacadeLocal.findById(this.person.getId());  
 this.personFacadeLocal.remove(this.person);  
 return "personList";  
 }  
  
 public Collection<Person> getAll() {  
 return this.personFacadeLocal.findAll();  
 }  
}

DrivingLicenseBean.class:

package Beans;  
  
import Entity.DrivingLicense;  
import Entity.DrivingLicenseCategory;  
import Facade.DrivingLicenseCategoryFacadeLocal;  
import Facade.DrivingLicenseFacadeLocal;  
import java.io.Serializable;  
import java.util.ArrayList;  
import java.util.Collection;  
import java.util.Date;  
import javax.ejb.EJB;  
import javax.enterprise.context.RequestScoped;  
import javax.inject.Named;  
  
@Named("drivingLicenseBean")  
@RequestScoped  
public class DrivingLicenseBean implements Serializable {  
 @EJB  
 DrivingLicenseFacadeLocal drivingLicenseFacadeLocal;  
 @EJB  
 DrivingLicenseCategoryFacadeLocal drivingLicenseCategoryFacadeLocal;  
 private DrivingLicense license = new DrivingLicense();  
 private Long categoryToAddId;  
  
 public DrivingLicenseBean() {  
 }  
  
 public DrivingLicense getLicense() {  
 return this.license;  
 }  
  
 public void setLicense(DrivingLicense license) {  
 this.license = license;  
 }  
  
 public Long getCategoryToAddId() {  
 return this.categoryToAddId;  
 }  
  
 public void setCategoryToAddId(Long categoryToAddId) {  
 this.categoryToAddId = categoryToAddId;  
 }  
  
 public String create() {  
 Date issDate = new Date();  
 this.license.setIssueDate(issDate);  
 Date expDate = new Date();  
 expDate.setYear(expDate.getYear() + 10);  
 this.license.setExpireDate(expDate);  
 DrivingLicenseCategory category = this.drivingLicenseCategoryFacadeLocal.find(this.categoryToAddId);  
 Collection<DrivingLicenseCategory> coll = new ArrayList();  
 coll.add(category);  
 this.license.setDrivingLicenseCategoryCollection(coll);  
 this.drivingLicenseFacadeLocal.create(this.license);  
 return "licenseList";  
 }  
  
 public String remove() {  
 this.license = this.drivingLicenseFacadeLocal.findById(this.license.getId());  
 this.drivingLicenseFacadeLocal.remove(this.license);  
 return "licenseList";  
 }  
  
 public String addCategory() {  
 this.license = this.drivingLicenseFacadeLocal.findById(this.license.getId());  
 DrivingLicenseCategory category = this.drivingLicenseCategoryFacadeLocal.findById(this.categoryToAddId);  
 this.license.getDrivingLicenseCategoryCollection().add(category);  
 this.drivingLicenseFacadeLocal.edit(this.license);  
 return "licenseList";  
 }  
  
 public Collection<DrivingLicense> getAll() {  
 return this.drivingLicenseFacadeLocal.findAll();  
 }  
}

DrivingLicenseCategory.class:

package Beans;  
  
import Entity.DrivingLicenseCategory;  
import Facade.DrivingLicenseCategoryFacadeLocal;  
import java.io.Serializable;  
import java.util.Collection;  
import javax.ejb.EJB;  
import javax.enterprise.context.RequestScoped;  
import javax.inject.Named;  
  
@Named("licenseCategoryBean")  
@RequestScoped  
public class DrivingLicenseCategoryBean implements Serializable {  
 @EJB  
 DrivingLicenseCategoryFacadeLocal drivingLicenseCategoryFacadeLocal;  
 DrivingLicenseCategory licenseCategory = new DrivingLicenseCategory();  
  
 public DrivingLicenseCategoryBean() {  
 }  
  
 public DrivingLicenseCategory getLicenseCategory() {  
 return this.licenseCategory;  
 }  
  
 public void setLicenseCategory(DrivingLicenseCategory licenseCategory) {  
 this.licenseCategory = licenseCategory;  
 }  
  
 public String create() {  
 this.drivingLicenseCategoryFacadeLocal.create(this.licenseCategory);  
 return "categoryList";  
 }  
  
 public String remove() {  
 this.drivingLicenseCategoryFacadeLocal.remove(this.licenseCategory);  
 return "categoryList";  
 }  
  
 public Collection<DrivingLicenseCategory> getAll() {  
 return this.drivingLicenseCategoryFacadeLocal.findAll();  
 }  
}

Представления:

/DrivingLicenseCategory/list.xhtml:

<?xml version="1.0" encoding="UTF-8"?>  
  
<!DOCTYPE html>  
<html xmlns="http://www.w3.org/1999/xhtml"  
 xmlns:h="http://xmlns.jcp.org/jsf/html"  
 xmlns:f="http://xmlns.jcp.org/jsf/core">  
 <h:head>  
 <title>List</title>  
 </h:head>  
 <h:body>  
 <f:view>  
 <h:form>  
 <h:dataTable value="#{licenseCategoryBean.all}" var="item">  
 <h:column>  
 <f:facet name="header">  
 Id  
 </f:facet>  
 <h:outputText value="#{item.id}"/>  
 </h:column>  
 <h:column>  
 <f:facet name="header">  
 Name  
 </f:facet>  
 <h:outputText value="#{item.categoryName}"/>  
 </h:column>  
 <h:column>  
 <f:facet name="header">  
 Driving licenses  
 </f:facet>  
 <h:dataTable value="#{item.drivingLicenseCollection}" var="license">  
 <h:column>  
 <h:outputText value="#{license.driver}"/>  
 </h:column>  
 </h:dataTable>  
 </h:column>  
 </h:dataTable>  
 </h:form>  
 </f:view>  
 <br/>  
 <h:link outcome="/DrivingLicenseCategory/create" value="Add new" />  
 <br/>  
 <br/>  
 <h:link outcome="/index" value="Back to index" />  
 </h:body>  
</html>

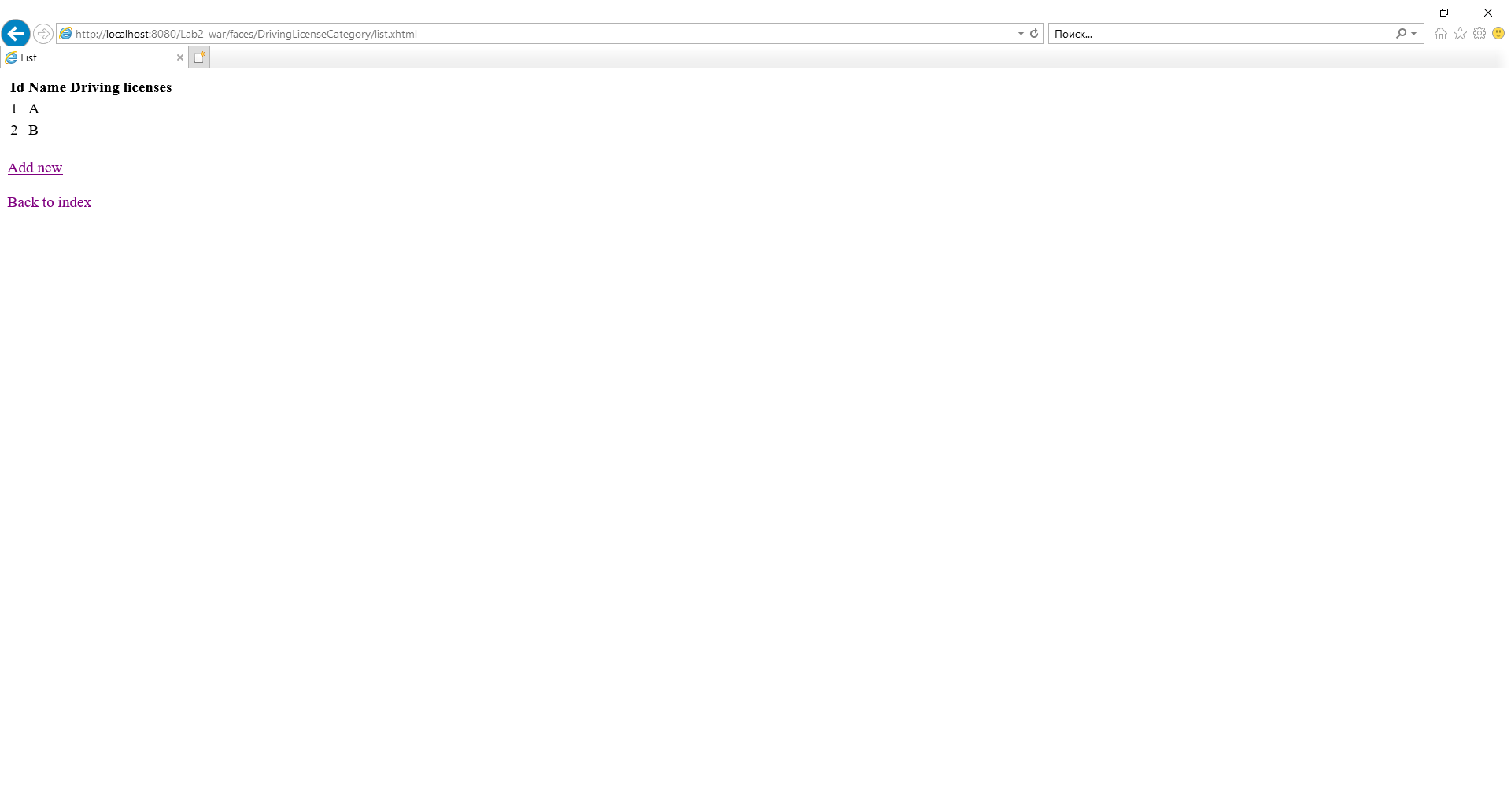


Рис.1.

/DrivingLicenseCategory/create.xhtml:

<?xml version="1.0" encoding="UTF-8"?>  
  
<!DOCTYPE html>  
 <html xmlns="http://www.w3.org/1999/xhtml"  
 xmlns:h="http://xmlns.jcp.org/jsf/html">  
 <h:head>  
 <title>Create license category</title>  
 </h:head>  
 <h:body>  
 <h:form>  
 <table border="1">  
 <tbody>  
 <tr>  
 <td>Name</td>  
 <td><h:inputText value="#{licenseCategoryBean.licenseCategory.categoryName}"/></td>  
 </tr>  
 </tbody>  
 </table>  
 <h:commandButton action="#{licenseCategoryBean.create}" value="Create"/>  
 </h:form>  
 </h:body>  
</html>

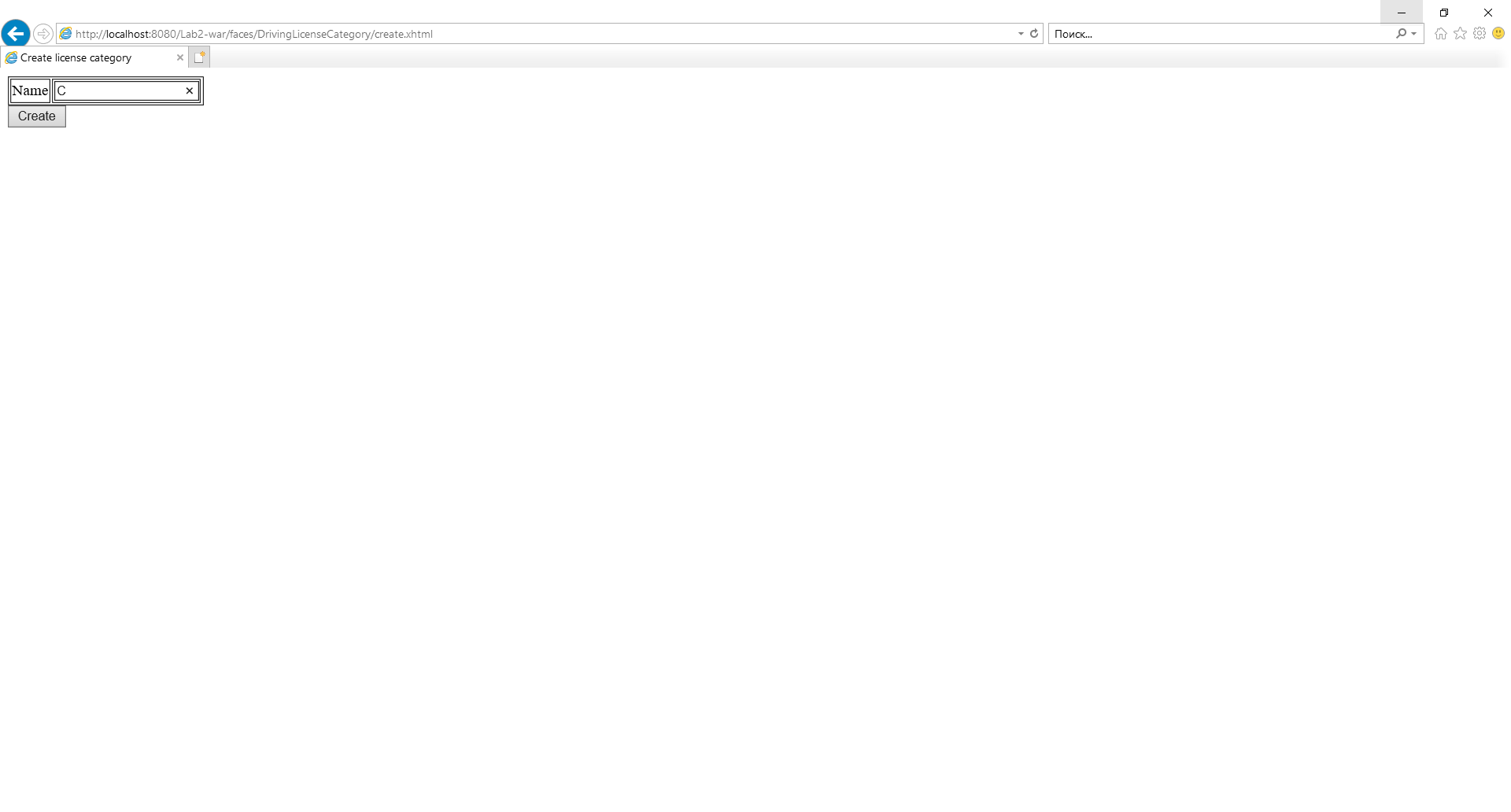


Рис.2.

Результат добавления.

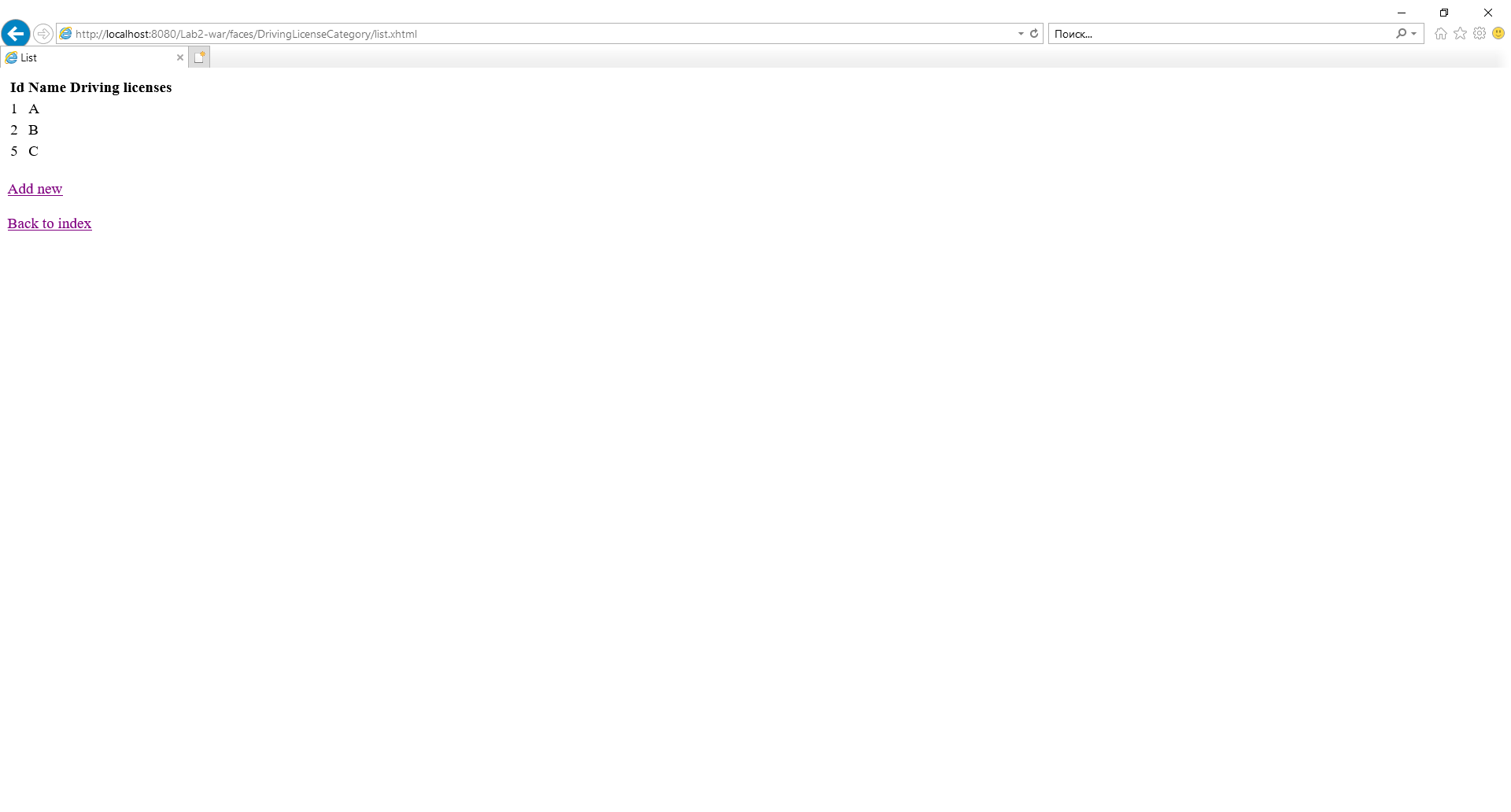


Рис.3.

/DrivingLicense/create.xhtml:

<?xml version="1.0" encoding="UTF-8"?>  
  
<!DOCTYPE html>  
<html xmlns="http://www.w3.org/1999/xhtml"  
 xmlns:h="http://xmlns.jcp.org/jsf/html"  
 xmlns:f="http://xmlns.jcp.org/jsf/core">  
 <h:head>  
 <title>Add driving license</title>  
 </h:head>  
 <h:body>  
 <h:form>  
 <table border="1">  
 <tbody>  
 <tr>  
 <td>Driving license category</td>  
 <td>  
 <h:selectOneMenu value="#{drivingLicenseBean.categoryToAddId}">  
 <f:selectItems value="#{licenseCategoryBean.all}" var="cat"   
 itemLabel="#{cat.categoryName}" itemValue="#{cat.id}"/>  
 <f:convertNumber />  
 </h:selectOneMenu>  
 </td>  
 </tr>  
 </tbody>  
 </table>  
 <h:commandButton action="#{drivingLicenseBean.create}" value="Create"/>  
 </h:form>  
 </h:body>  
</html>

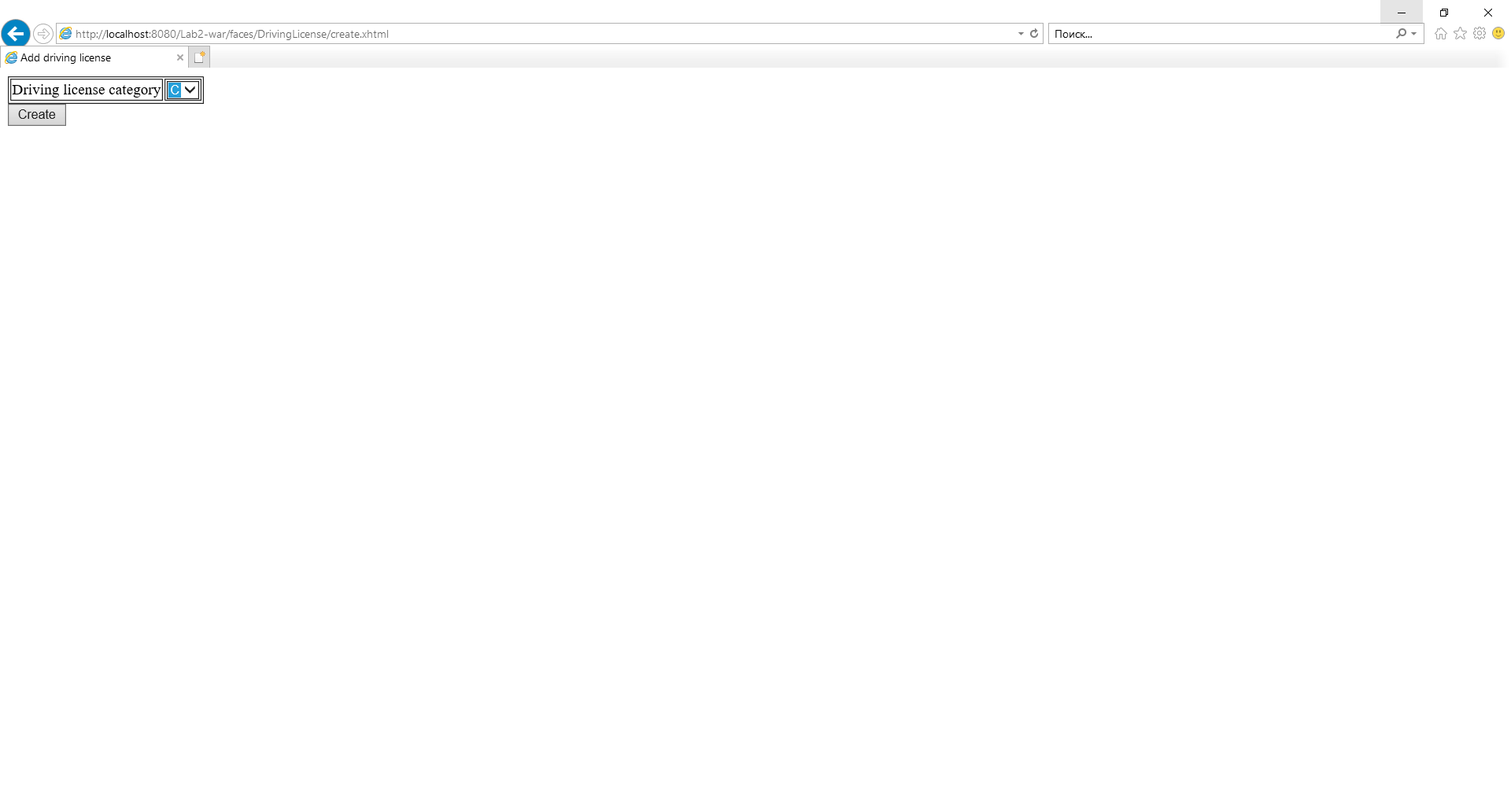


Рис.4.

/DrivingLicense/list.xhtml:

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html>  
<html xmlns="http://www.w3.org/1999/xhtml"  
 xmlns:h="http://xmlns.jcp.org/jsf/html"  
 xmlns:f="http://xmlns.jcp.org/jsf/core">  
 <h:head>  
 <title>List</title>  
 </h:head>  
 <h:body>  
 <f:view>  
 <h:form>  
 <h:dataTable value="#{drivingLicenseBean.all}" var="item">  
 <h:column>  
 <f:facet name="header">  
 Id  
 </f:facet>  
 <h:outputText value="#{item.id}"/>  
 </h:column>  
 <h:column>  
 <f:facet name="header">  
 Issue date  
 </f:facet>  
 <h:outputText value="#{item.issueDate}">  
 <f:convertDateTime pattern="d MMMM yyyy, EEEE" />  
 </h:outputText>  
 </h:column>  
 <h:column>  
 <f:facet name="header">  
 Expire date  
 </f:facet>  
 <h:outputText value="#{item.expireDate}">  
 <f:convertDateTime pattern="d MMMM yyyy, EEEE" />  
 </h:outputText>  
 </h:column>  
 <h:column>  
 <f:facet name="header">  
 Categories  
 </f:facet>  
 <h:dataTable value="#{item.drivingLicenseCategoryCollection}" var="category">  
 <h:column>  
 <h:outputText value="#{category.id}"/>  
 </h:column>  
 </h:dataTable>  
 </h:column>  
 </h:dataTable>  
 </h:form>  
 </f:view>  
 <h:link outcome="/DrivingLicense/create" value="Add new"/>  
 <br/>  
 <h:link outcome="/DrivingLicense/addCategory" value="Add category" />  
 <br/>  
 <h:link outcome="/DrivingLicense/remove" value="Remove" />  
 <br/>  
 <br/>  
 <h:link outcome="/index" value="Back to index" />  
 </h:body>  
</html>

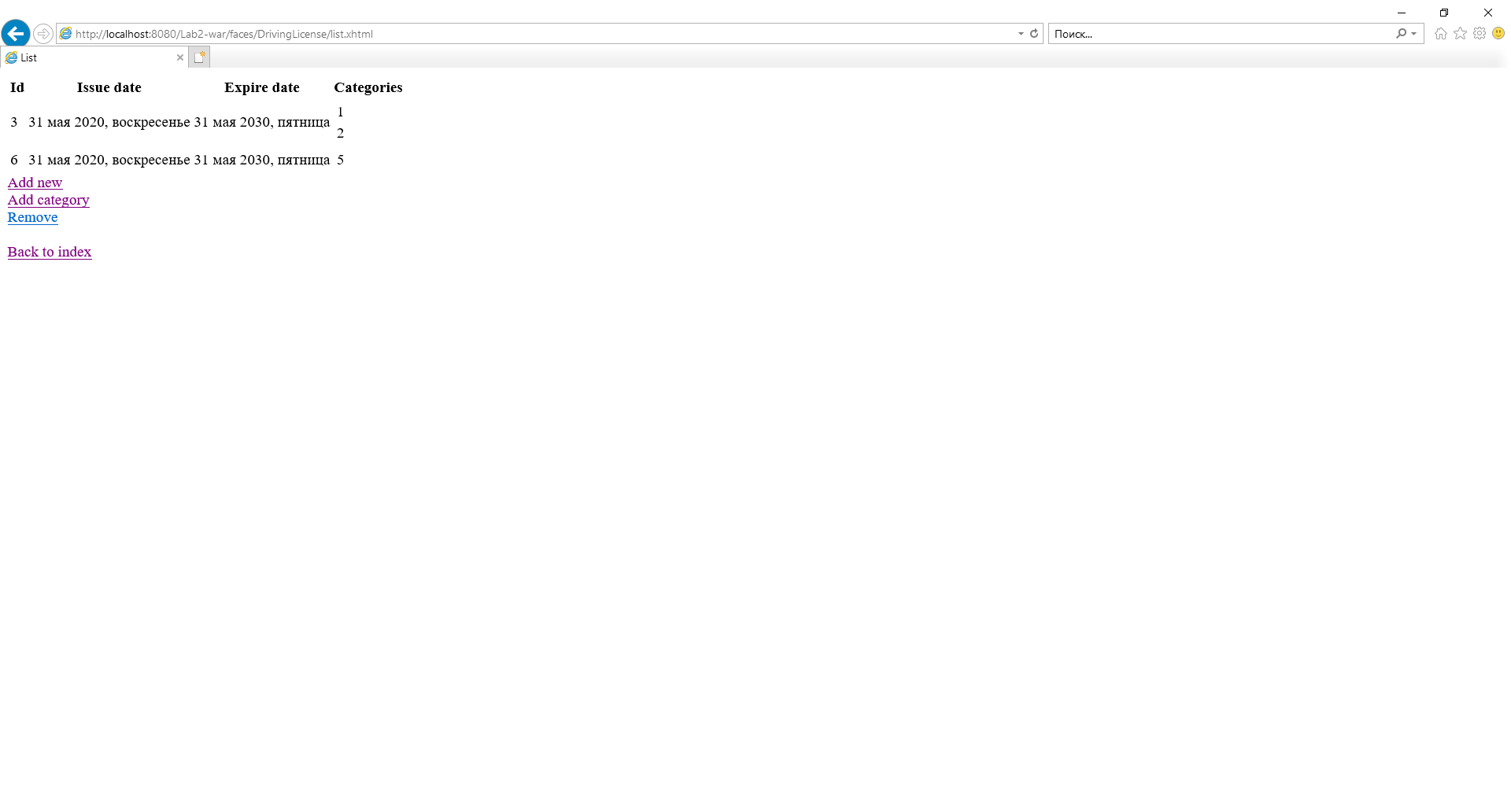


Рис.5.

/DrivingLicense/addCategory.xhtml:

<?xml version="1.0" encoding="UTF-8"?>  
  
<!DOCTYPE html>  
<html xmlns="http://www.w3.org/1999/xhtml"  
 xmlns:h="http://xmlns.jcp.org/jsf/html"  
 xmlns:f="http://xmlns.jcp.org/jsf/core">  
 <h:head>  
 <title>Add category</title>  
 </h:head>  
 <h:body>  
 <h:form>  
 <table border="1">  
 <tbody>  
 <tr>  
 <td>Driving license</td>  
 <td>  
 <h:selectOneMenu value="#{drivingLicenseBean.license.id}">  
 <f:selectItems value="#{drivingLicenseBean.all}" var="lic"  
 itemLabel="#{lic.id}" itemValue="#{lic.id}"/>  
 <f:convertNumber />  
 </h:selectOneMenu>  
 </td>  
 </tr>  
 <tr>  
 <td>Category to add</td>  
 <td>  
 <h:selectOneMenu value="#{drivingLicenseBean.categoryToAddId}">  
 <f:selectItems value="#{licenseCategoryBean.all}" var="cat"   
 itemLabel="#{cat.categoryName}" itemValue="#{cat.id}"/>  
 <f:convertNumber />  
 </h:selectOneMenu>  
 </td>  
 </tr>  
 </tbody>  
 </table>  
 <h:commandButton action="#{drivingLicenseBean.addCategory}" value="Add"/>  
 </h:form>  
 </h:body>  
</html>

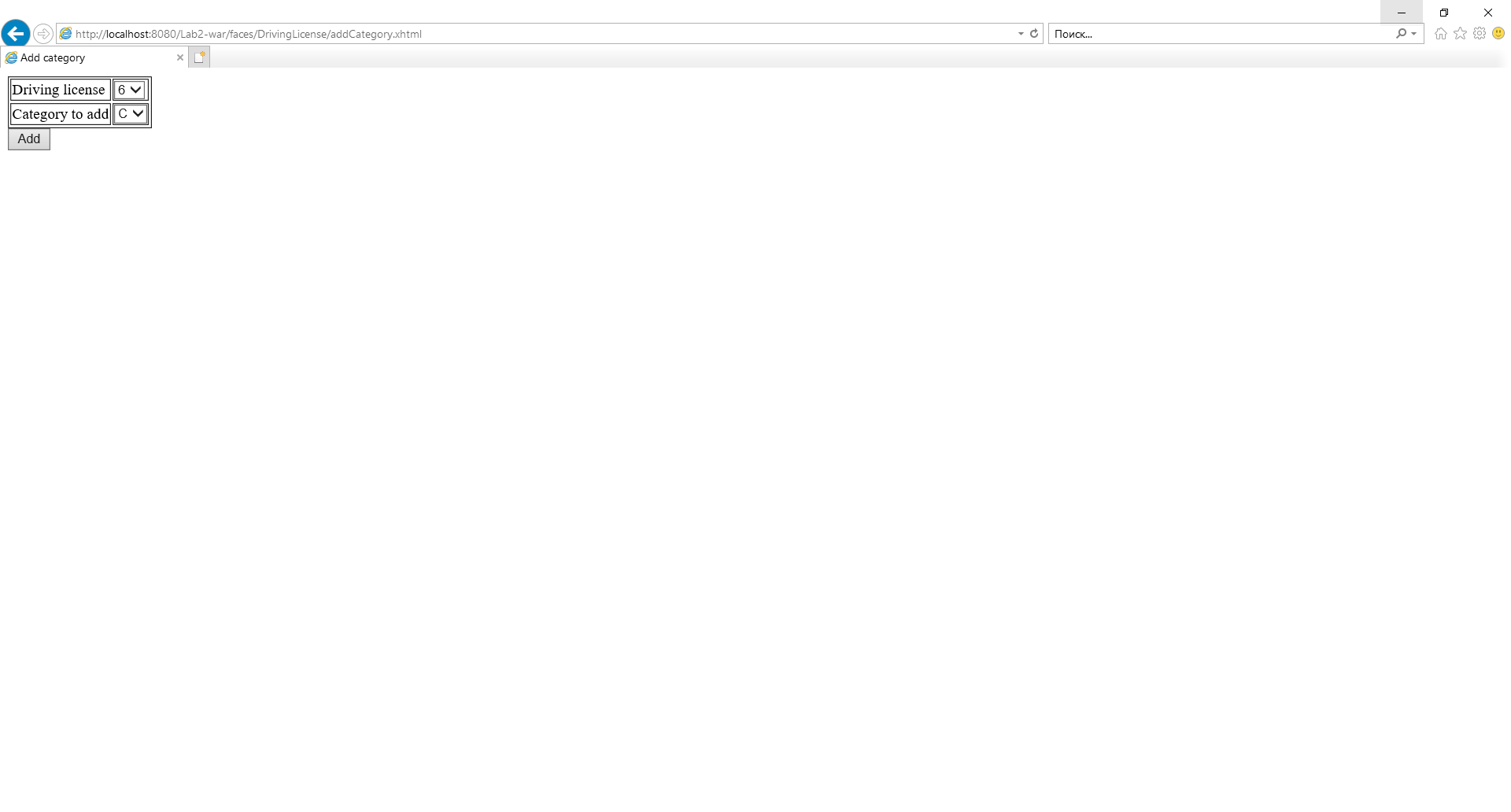


Рис.6.

/Person/create.xhtml:

<?xml version="1.0" encoding="UTF-8"?>  
  
<!DOCTYPE html>  
<html xmlns="http://www.w3.org/1999/xhtml"  
 xmlns:h="http://xmlns.jcp.org/jsf/html"  
 xmlns:f="http://xmlns.jcp.org/jsf/core">  
 <h:head>  
 <title>Add person</title>  
 </h:head>  
 <h:body>  
 <h:form>  
 <table border="1">  
 <tbody>  
 <tr>  
 <td>First name</td>  
 <td><h:inputText value="#{personBean.person.firstName}"/></td>  
 </tr>  
 <tr>  
 <td>Last name</td>  
 <td><h:inputText value="#{personBean.person.lastName}"/></td>  
 </tr>  
 <tr>  
 <td>Driving license</td>  
 <td>  
 <h:selectOneMenu value="#{personBean.selectedDrivingLicense}">  
 <f:selectItems value="#{drivingLicenseBean.all}" var="lic"  
 itemLabel="#{lic.id}" itemValue="#{lic.id}"/>  
 <f:convertNumber />  
 </h:selectOneMenu>  
 </td>  
 </tr>  
 </tbody>  
 </table>  
 <h:commandButton action="#{personBean.create}" value="Create"/>  
 </h:form>  
 </h:body>  
</html>

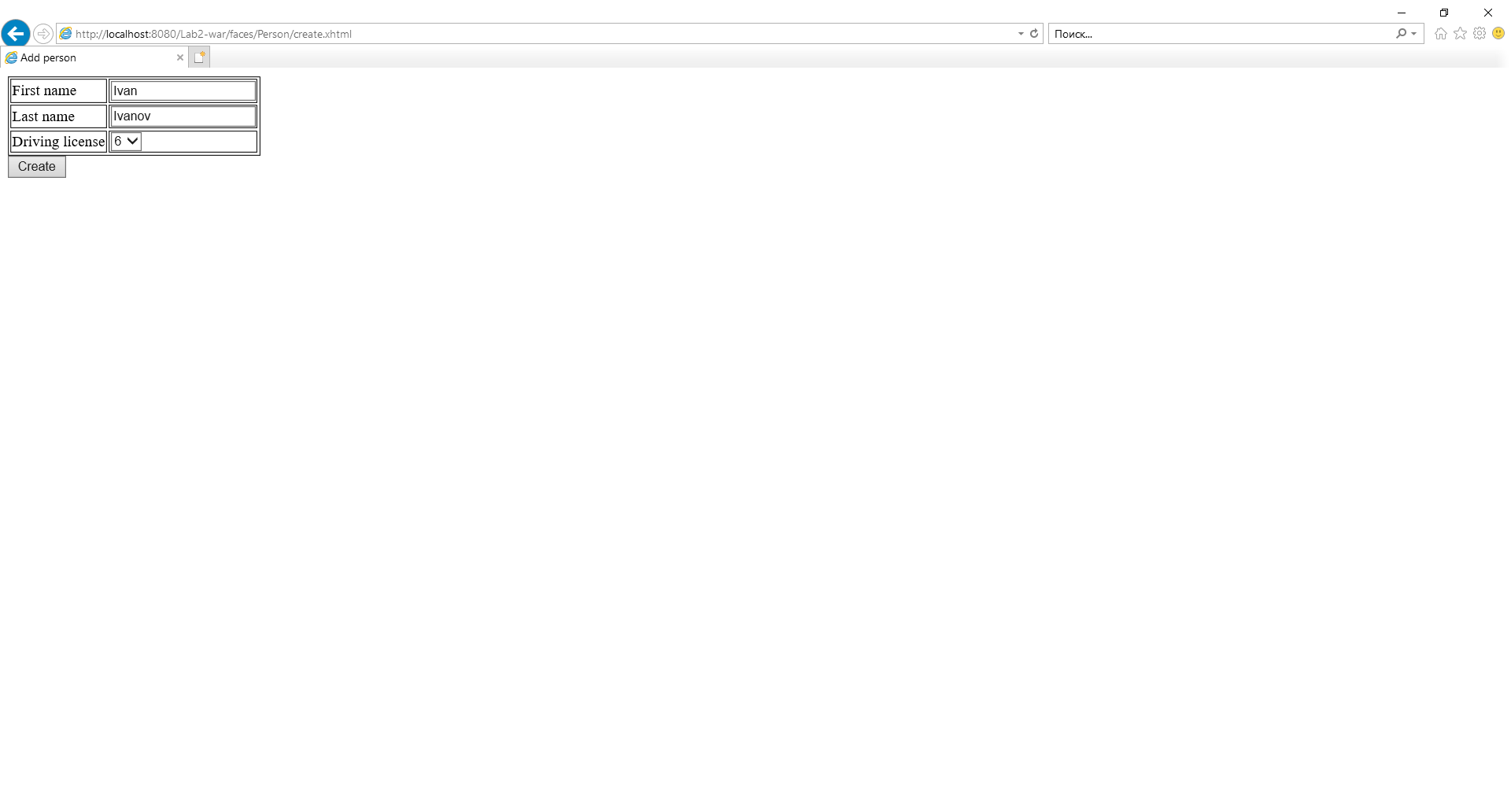


Рис.7.

/Person/list.xhtml:

<?xml version="1.0" encoding="UTF-8"?>  
  
<!DOCTYPE html>  
<html xmlns="http://www.w3.org/1999/xhtml"  
 xmlns:h="http://xmlns.jcp.org/jsf/html"  
 xmlns:f="http://xmlns.jcp.org/jsf/core">  
 <h:head>  
 <title>List</title>  
 </h:head>  
 <h:body>  
 <f:view>  
 <h:form>  
 <h:dataTable value="#{personBean.all}" var="item" border="1">  
 <h:column>  
 <f:facet name="header">  
 Id  
 </f:facet>  
 <h:outputText value="#{item.id}"/>  
 </h:column>  
 <h:column>  
 <f:facet name="header">  
 Name  
 </f:facet>  
 <h:outputText value="#{item.firstName}"/>  
 </h:column>  
 <h:column>  
 <f:facet name="header">  
 Surname  
 </f:facet>  
 <h:outputText value="#{item.lastName}"/>  
 </h:column>  
 <h:column>  
 <f:facet name="header">  
 Driving license  
 </f:facet>  
 <h:column>  
 <h:outputText value="#{item.drivingLicenseId.id}"/>  
 </h:column>  
 <h:column>  
 <h:outputText value="#{item.drivingLicenseId.issueDate}">  
 <f:convertDateTime pattern="d MMMM yyyy, EEEE" />  
 </h:outputText>  
 </h:column>  
 <h:column>  
 <h:outputText value="#{item.drivingLicenseId.expireDate}">  
 <f:convertDateTime pattern="d MMMM yyyy, EEEE" />  
 </h:outputText>  
 </h:column>  
 </h:column>  
   
 </h:dataTable>  
 </h:form>  
 </f:view>  
 <br/>  
 <h:link outcome="/Person/create" value="Add new" />  
 <br/>  
 <h:link outcome="/Person/remove" value="Remove" />  
 <br/>  
 <br/>  
 <h:link outcome="/index" value="Back to index" />  
 </h:body>  
</html>

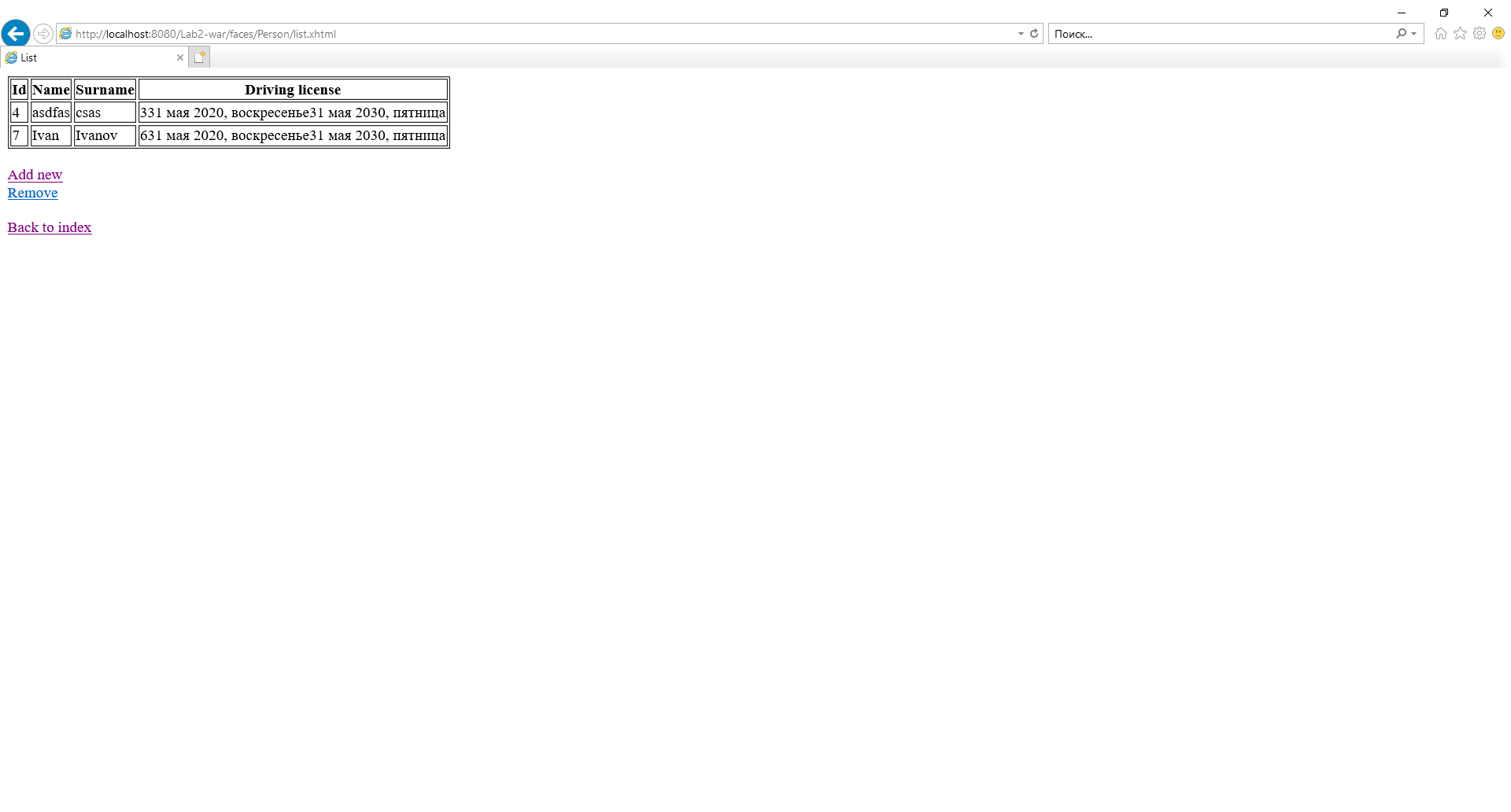


Рис.8.

/Person/remove.xhtml:

<?xml version="1.0" encoding="UTF-8"?>  
  
<!DOCTYPE html>  
<html xmlns="http://www.w3.org/1999/xhtml"  
 xmlns:h="http://xmlns.jcp.org/jsf/html"  
 xmlns:f="http://xmlns.jcp.org/jsf/core">  
 <h:head>  
 <title>Remove person</title>  
 </h:head>  
 <h:body>  
 <h:form>  
 <h:selectOneMenu value="#{personBean.person.id}">  
 <f:selectItems value="#{personBean.all}" var="person"  
 itemLabel="#{person}" itemValue="#{person.id}"/>  
 <f:convertNumber />  
 </h:selectOneMenu>  
 <h:commandButton action="#{personBean.remove}" value="Remove"/>  
 </h:form>  
 </h:body>  
</html>

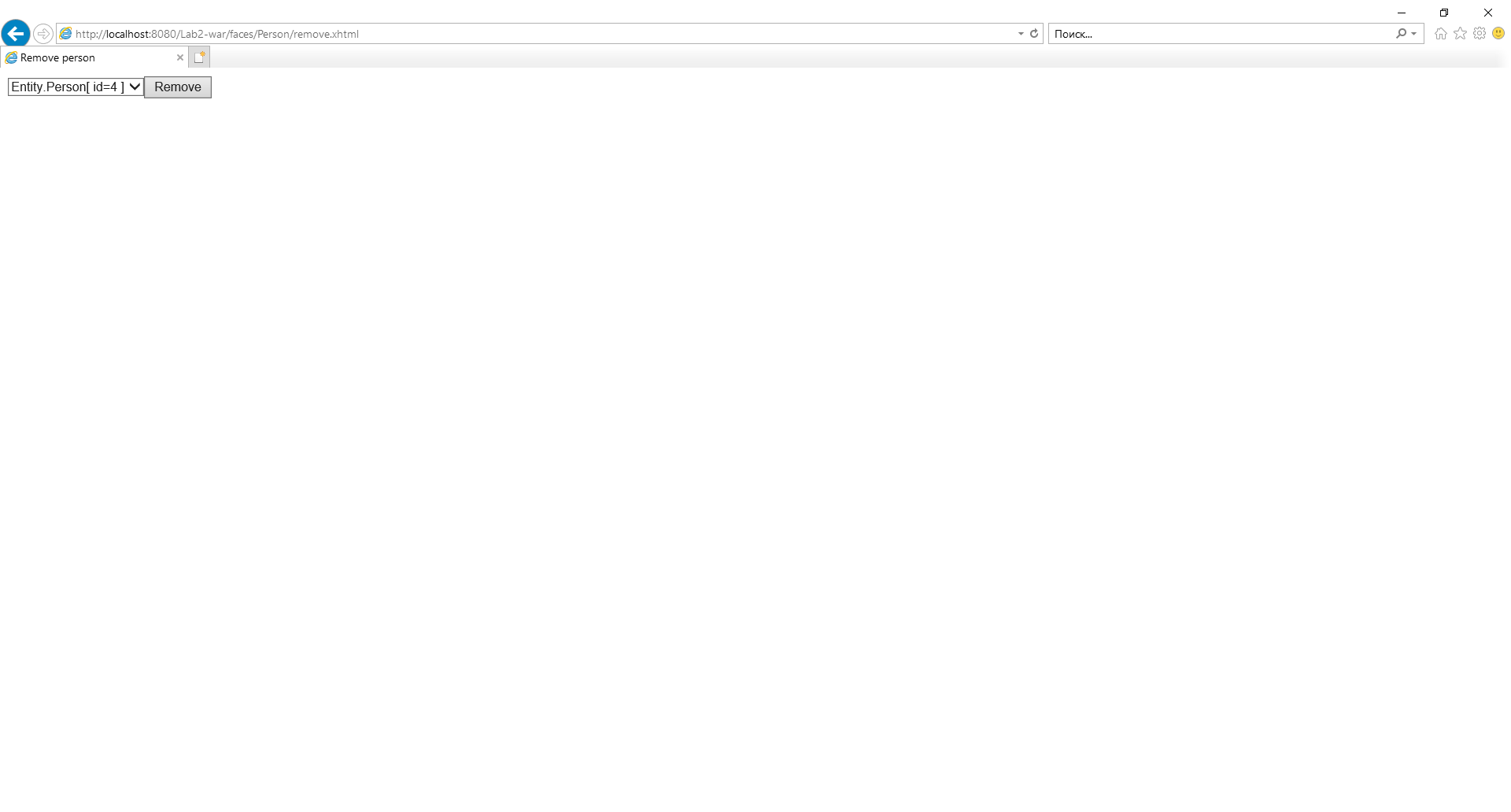


Рис.9.

Результат выполнения удаления.

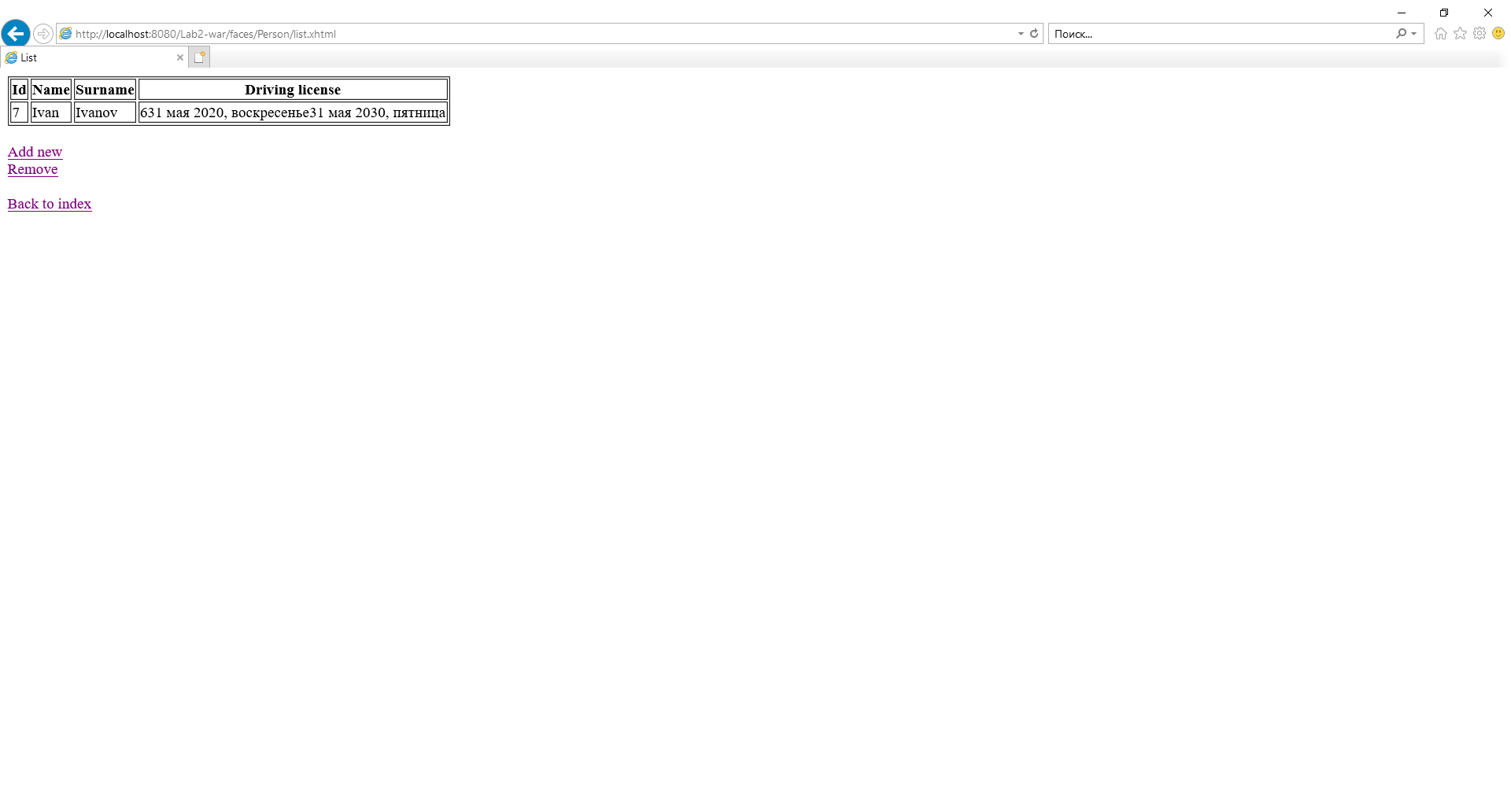


Рис.10.

ВЫВОДЫ

В ходе выполнения лабораторной работы было освоено создание классов сущностей, в том числе путем обратного проектирования реляционной базы данных средствами NetBeans 5.5, работа с менеджером сущностей и получены навыки работы со средой разработки.