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)**

Кафедра информационных систем и программной инженерии

**Лабораторная работа № 02**

**по дисциплине**

**««Распределенные программные системы»**

Выполнил:

ст. гр. ПРИ-117

Хлызова В.Г.

Принял:

Трифонов Д.А.

Владимир, 2020

ЦЕЛЬ РАБОТЫ

Освоить создание сессионных компонентов EJB 3.1 и использование сессионных компонентов в веб-приложении с помощью CDI, а также получить навыки работы со средой разработки NetBeans.

ЗАДАНИЕ

1. Разработать Java EE-приложение, состоящее из EJB-модуля и модуля веб-приложения.

2. Разработать сессионные компоненты всех видов: без состояния, с состоянием, синглтон.

3. Разработать (в целом) не менее одного удаленного и не менее одного локального бизнес-интерфейса для компонентов.

4. Использовать представление без интерфейса для одного из компонентов.

5. Использовать инъекцию зависимостей CDI.

6. Использовать не менее трех различных контекстов для размещения экземпляров компонентов, в том числе обязательно контекст диалога (с организацией длительного диалога).

7. Реализовать взаимодействие между компонентами путем инициирования и обработки событий.

8. Разместить не менее одного сессионного компонента в модуле веб-приложения.

9. Разработать обработчик события ЖЦ и обработчик вызова бизнес-метода. Один из них должен быть методом класса компонента, а второй – методом отдельного класса-перехватчика.

10. Показать успешную установку Java EE-приложения и выполнение заданной функции в веб-приложении.

ВЫПОЛНЕНИЕ РАБОТЫ

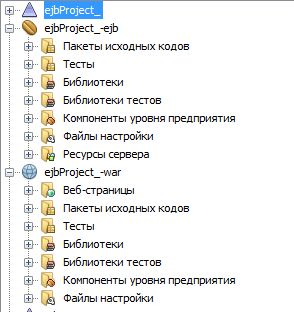


Рисунок 1. Общая структура проекта.

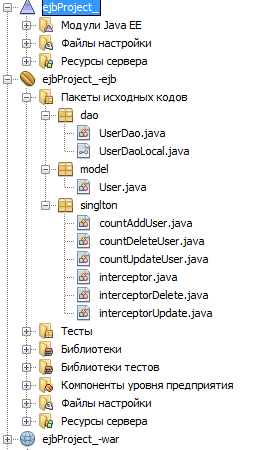


Рисунок 2. Содержимое EJB-модуля.

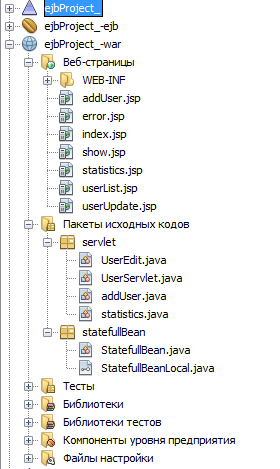


Рисунок 3. Содержимое WEB-модуля.

В проекте EJB-модуля создаем класс, представляющий данные из таблицы базы данных

.

package model;

import java.io.Serializable;

import java.util.Date;

public class User implements Serializable {

private int userID;

private String login;

private String password;

private String email;

private Date bDate;

private String role;

public User(int id, String login, String password, String email, Date date, String role){

userID=id;

this.login=login;

this.email=email;

this.password=password;

bDate = date;

this.role = role;

}

public User() {

}

public int getUserID() {

return userID;

}

public void setUserID(int UserID) {

this.userID = UserID;

}

public String getLogin() {

return login;

}

public void setLogin(String Login) {

this.login = Login;

}

public String getPassword() {

return password;

}

public void setPassword(String Password) {

this.password = Password;

}

public String getEmail() {

return email;

}

public void setEmail(String Email) {

this.email = Email;

}

public Date getbDate() {

return bDate;

}

public void setbDate(Date bDate) {

this.bDate = bDate;

}

public String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

}

Создаем DAO-класс и помечаем его аннотацией @Stateless, что сделает его классом сессионного компонента без состояния. Переносим инициализацию полей url, user и password в метод обработки событий ЖЦ @PostConstruct. Помечаем методы вставки, изменения и удаления аннотацией @Interceptors и указываем класс перехватчика.

package dao;

import java.util.ArrayList;

import java.util.List;

import javax.ejb.Stateless;

import model.User;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.annotation.PostConstruct;

import javax.interceptor.Interceptors;

import singlton.interceptor;

import singlton.interceptorDelete;

import singlton.interceptorUpdate;

@Stateless

public class UserDao implements UserDaoLocal {

private static String url;

private static String username;

private static String password;

private String query;

@PostConstruct

private void init(){

url = "jdbc:mysql://localhost:3306/buymypoem?"

+ "useSSL=false&useUnicode=true&serverTimezone=Europe/Moscow";

username = "root";

password = "";

}

@Override

public List<User> getAllUser() {

List<User> userList = new ArrayList<>();

query = "select\*from user;";

try {

Class.forName("com.mysql.cj.jdbc.Driver");

} catch (ClassNotFoundException ex) {

Logger.getLogger(UserDao.class.getName()).log(Level.SEVERE, null, ex);

}

try (

Connection con = DriverManager.getConnection(url, username, password);

Statement stmt = con.createStatement(); ResultSet rs = stmt.executeQuery(query);) {

while (rs.next()) {

User user = new User(rs.getInt("UserId"), rs.getString("login"),

rs.getString("password"), rs.getString("email"),

rs.getDate("birthdate"), rs.getString("role"));

userList.add(user);

}

} catch (SQLException sqlEx) {

Logger.getLogger(UserDao.class.getName()).log(Level.SEVERE, null, sqlEx);

}

return userList;

}

@Override

public boolean Delete(int id) {

query = "delete from user where userId=" + id +";";

try (Connection con = DriverManager.getConnection(url, username, password)) {

if(con.prepareStatement(query).executeUpdate()>0) return true;

}catch (SQLException sqlEx){

Logger.getLogger(UserDao.class.getName()).log(Level.SEVERE, null, sqlEx);

}

return false;

}

@Override

@Interceptors(interceptorDelete.class)

public User getById(int id) {

User user = null;

query = "select\*from user where userId = " + id + ";";

try {

Class.forName("com.mysql.cj.jdbc.Driver");

} catch (ClassNotFoundException ex) {

Logger.getLogger(UserDao.class.getName()).log(Level.SEVERE, null, ex);

}

try (

Connection con = DriverManager.getConnection(url, username, password);

Statement stmt = con.createStatement(); ResultSet rs = stmt.executeQuery(query);) {

while (rs.next()) {

user = new User(rs.getInt("UserId"), rs.getString("login"),

rs.getString("password"), rs.getString("email"),

rs.getDate("birthdate"), rs.getString("role"));

}

} catch (SQLException sqlEx) {

Logger.getLogger(UserDao.class.getName()).log(Level.SEVERE, null, sqlEx);

}

return user;

}

@Override

@Interceptors(interceptorUpdate.class)

public boolean editUser(User user) {

query = "update user set login='"+ user.getLogin() + "', "

+ "password='"+ user.getPassword() + "', "

+ "email='"+ user.getEmail() + "' "

+ "where userID=" +user.getUserID() + ";";

try {

Class.forName("com.mysql.cj.jdbc.Driver");

} catch (ClassNotFoundException ex) {

Logger.getLogger(UserDao.class.getName()).log(Level.SEVERE, null, ex);

}

try (Connection con = DriverManager.getConnection(url, username, password)) {

if(con.prepareStatement(query).executeUpdate()>0)return true;

}catch (SQLException sqlEx){

Logger.getLogger(UserDao.class.getName()).log(Level.SEVERE, null, sqlEx);

}

return false;

}

@Override

@Interceptors(interceptor.class)

public boolean insertUser(User user) {

query = "insert into user (login, email, password, birthdate) " +

"values (?, ?, ?, ?);";

try {

Class.forName("com.mysql.cj.jdbc.Driver");

} catch (ClassNotFoundException ex) {

Logger.getLogger(UserDao.class.getName()).log(Level.SEVERE, null, ex);

}

try(Connection con = DriverManager.getConnection(url, username, password)){

PreparedStatement pStmt = con.prepareStatement(query);

pStmt.setString(1, user.getLogin());

pStmt.setString(2, user.getEmail());

pStmt.setString(3, user.getPassword());

pStmt.setDate(4,new java.sql.Date(user.getbDate().getTime()));

if (pStmt.executeUpdate()>0) return true;

}catch (SQLException sqlEx){

Logger.getLogger(UserDao.class.getName()).log(Level.SEVERE, null, sqlEx);

}

return false;

}

}

Для DAO-класса создаем интерфейс, помечаем его аннотацией @Remote и помещаем в него открытые методы DAO-класса.

package dao;

import java.util.List;

import javax.ejb.Remote;

import model.User;

@Remote

public interface UserDaoLocal {

public List<User> getAllUser();

public boolean Delete(int id);

public User getById(int id) ;

public boolean editUser(User user);

public boolean insertUser(User user);

}

В WEB-модуле создаем сессионный компонент с состоянием, помечаем класс аннотацией @Stateful и @SessionScoped(чтобы поместить компонент в область видимости сеанса)

package statefullBean;

import java.io.Serializable;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.PostConstruct;

import javax.ejb.Stateful;

import javax.enterprise.context.SessionScoped;

@Stateful

@SessionScoped

public class StatefullBean implements StatefullBeanLocal, Serializable {

private List<String> loginList = new ArrayList<>();;

@PostConstruct

private void init(){

loginList = new ArrayList<>();

}

@Override

public void addLogin(String login){

loginList.add(login);

}

@Override

public List<String> returnList() {

return loginList;

}

}

Для созданного компонента с состоянием создаем локальный интерфейс, помечаем аннотацией @Local.

package statefullBean;

import java.util.List;

import javax.ejb.Local;

@Local

public interface StatefullBeanLocal {

public void addLogin(String login);

public List<String> returnList();

}

В проекте EJB-модуля создаём сессионный компонент типа singleton. В данной лабораторной он используется для сбора статистики выполнения функций удаления, изменения и добавления записей в веб-приложении. Ниже представлен код компонента ориентированный на функцию добавления новой записи. Компоненты на удаление и обновление выглядят аналогично.

package singlton;

import javax.annotation.PostConstruct;

import javax.ejb.Singleton;

@Singleton

public class countAddUser {

int count;

@PostConstruct

public void init() {

count = 0;

}

public int getCount() {

return count;

}

public void plus(){

count++;

}

}

Создаем класс перехватчик для созданного сессионного компонента типа singleton. Классы перехватчики для компонентов на удаление и обновление выглядят аналогично.

package singlton;

import java.io.Serializable;

import javax.ejb.EJB;

import javax.interceptor.AroundInvoke;

import javax.interceptor.InvocationContext;

public class interceptor implements Serializable{

@EJB

countAddUser caUser;

@AroundInvoke

public Object addInter(InvocationContext context) throws Exception{

caUser.plus();

return context.proceed();

}

}

Помещаем в WEB-модуль сервлеты и создаем jsp странички. Содержание web.xml:

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

<web-app version="3.1" xmlns="http://xmlns.jcp.org/xml/ns/javaee"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee

http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd">

<servlet>

<servlet-name>UserServlet</servlet-name>

<servlet-class>servlet.UserServlet</servlet-class>

</servlet>

<servlet>

<servlet-name>UserEdit</servlet-name>

<servlet-class>servlet.UserEdit</servlet-class>

</servlet>

<servlet>

<servlet-name>addUser</servlet-name>

<servlet-class>servlet.addUser</servlet-class>

</servlet>

<servlet>

<servlet-name>statistics</servlet-name>

<servlet-class>servlet.statistics</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>UserServlet</servlet-name>

<url-pattern>/user</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>UserEdit</servlet-name>

<url-pattern>/userEdit</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>addUser</servlet-name>

<url-pattern>/addUser</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>statistics</servlet-name>

<url-pattern>/statistics</url-pattern>

</servlet-mapping>

<session-config>

<session-timeout>

30

</session-timeout>

</session-config>

<welcome-file-list>

<welcome-file>index.jsp</welcome-file>

</welcome-file-list>

</web-app>

Список всех пользователей и удаление

package servlet;

import dao.UserDaoLocal;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.List;

import javax.ejb.EJB;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import model.User;

public class UserServlet extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

try (PrintWriter out = response.getWriter()) {

out.println("<!DOCTYPE html>");

out.println("<html>");

out.println("<head>");

out.println("<title>Servlet UserServlet</title>");

out.println("</head>");

out.println("<body>");

out.println("<h1>Servlet UserServlet at " + request.getContextPath() + "</h1>");

out.println("</body>");

out.println("</html>");

}

}

@EJB

private UserDaoLocal userDao;

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

try{

List<User> userList = userDao.getAllUser();

request.setAttribute("userList", userList);

RequestDispatcher requestDispatcher = request.getRequestDispatcher("userList.jsp");

requestDispatcher.forward(request, response);

}catch (Exception e) {

throw new ServletException(e.getMessage());

}

}

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

RequestDispatcher dispatcher;

int id = Integer.parseInt(request.getParameter("id"));

userDao.Delete(id);

dispatcher=request.getRequestDispatcher("index.jsp");

dispatcher.forward(request,response);

}

@Override

public String getServletInfo() {

return "Short description";

}

}

Страничка со списком всех пользователей

<%@ page contentType="text/html;charset=UTF-8" language="java" %>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Список пользователей</title>

<style>

tr:nth-child(odd){

background: antiquewhite;

color: black;

}

table{

border-spacing:0;

width: 80%;

height: 100%;

font-size: 20px;

}

.btn{

background: none;

color: black;

border: 0;

font-size: 20px;

}

</style>

</head>

<body>

<table align="center">

<tr>

<td>логин</td>

<td>пароль</td>

<td>email</td>

<td>дата рождения</td>

<td>роль в системе</td>

<td></td>

<td></td>

</tr>

<c:forEach items="${userList}" var="user">

<tr>

<td>${user.getLogin()}</td>

<td>${user.getPassword()}</td>

<td>${user.getEmail()}</td>

<td>${user.getbDate()}</td>

<td>${user.getRole()}</td>

<td>

<form action="userEdit" method="get">

<input type="hidden" name="id" value="${user.getUserID()}">

<input class="btn" type="submit" value="редактировать">

</form>

</td>

<td><form action="user" method="post">

<input type="hidden" name="id" value="${user.getUserID()}">

<input class="btn" type="submit" value="удалить">

</form>

</td>

</tr>

</c:forEach>

</table>

</body>

</html>

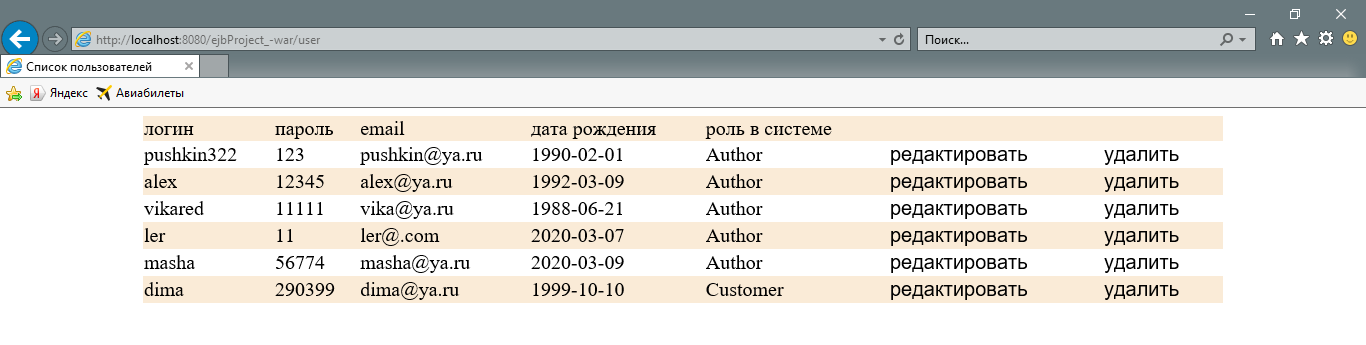


Рисунок 4. Страница со списком всех пользователей.

Добавление нового пользователя

package servlet;

import dao.UserDaoLocal;

import java.io.IOException;

import java.io.PrintWriter;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.List;

import javax.ejb.EJB;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import model.User;

import statefullBean.StatefullBeanLocal;

public class addUser extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

try (PrintWriter out = response.getWriter()) {

out.println("<!DOCTYPE html>");

out.println("<html>");

out.println("<head>");

out.println("<title>Servlet addUser</title>");

out.println("</head>");

out.println("<body>");

out.println("<h1>Servlet addUser at " + request.getContextPath() + "</h1>");

out.println("</body>");

out.println("</html>");

}

}

@EJB

private StatefullBeanLocal slBean;

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

String go = request.getParameter("go");

if (go.equals("add")){

RequestDispatcher requestDispatcher = request.getRequestDispatcher("addUser.jsp");

requestDispatcher.forward(request, response);

}else{

// В компонент с стоянием записываются логины добавленных пользователей и выводятся на странице show.jsp

List<String> ls = slBean.returnList();

request.setAttribute(“userList”, ls);

RequestDispatcher requestDispatcher = request.getRequestDispatcher(“show.jsp”);

requestDispatcher.forward(request, response);

}

}

@EJB

private UserDaoLocal userDao;

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

if (!(request.getParameter(“password”).equals(request.getParameter(“password2”)))){

request.setAttribute(“error”, “Пароли не совпадают”);

RequestDispatcher requestDispatcher = request.getRequestDispatcher(“error.jsp”);

requestDispatcher.forward(request, response);

}

User user = new User();

user.setLogin(request.getParameter(“login”));

user.setEmail(request.getParameter(“email”));

user.setPassword(request.getParameter(“password”));

SimpleDateFormat format = new SimpleDateFormat(“yyyy-MM-dd”);

String bdate = request.getParameter(“bdate”);

try {

Date date =format.parse(bdate);

user.setbDate(date);

} catch (ParseException e) {

e.printStackTrace();

}

if (userDao.insertUser(user)){

slBean.addLogin(request.getParameter(“login”));

}

RequestDispatcher requestDispatcher = request.getRequestDispatcher(“index.jsp”);

requestDispatcher.forward(request, response);

}

@Override

public String getServletInfo() {

return «Short description»;

}

}

Страница добавления

<%@page contentType=”text/html” pageEncoding=”UTF-8”%>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv=”Content-Type” content=”text/html; charset=UTF-8”>

<title>Создание нового пользователя</title>

</head>

<body>

<form action=”addUser” method=”post”>

<table>

<tr>

<td>Логин</td>

<td><input required type=”text” name=”login” /></td>

</tr>

<tr>

<td>Email</td>

<td><input required type=”text” name=”email” /></td>

</tr>

<tr>

<td>Дата рождения</td>

<td><input required type=”text” name=”bdate” /></td>

</tr>

<tr>

<td>Пароль</td>

<td><input required type=”password” name=”password” /></td>

</tr>

<tr>

<td>Повторите пароль</td>

<td><input required type=”password” name=”password2” /></td>

</tr>

<tr>

<td>

<input required type=”submit” value=”Сохранить”/>

</td></tr>

</table>

</form>

</body>

</html>

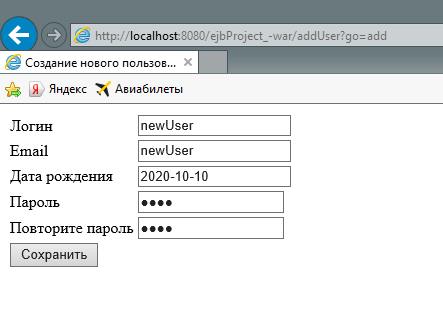


Рисунок 5. Страница добавления.

Страница со списком всех добавленных пользователей за сеанс

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>JSP Page</title>

</head>

<body>

<h3> Логины добавленных пользователей:&nbsp;

<c:forEach items="${userList}" var="user">

${user}&nbsp;

</c:forEach>

</h3>

</body>

</html>

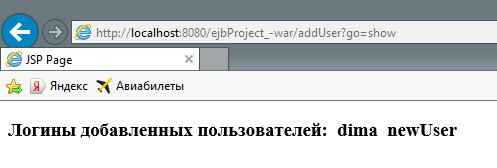


Рисунок 6. Страница с добавленными пользователями.

Изменение пользователя

package servlet;

import dao.UserDaoLocal;

import java.io.IOException;

import java.io.PrintWriter;

import javax.ejb.EJB;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import model.User;

public class UserEdit extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType(“text/html;charset=UTF-8”);

try (PrintWriter out = response.getWriter()) {

out.println(“<!DOCTYPE html>”);

out.println(“<html>”);

out.println(“<head>”);

out.println(“<title>Servlet UserEdit</title>”);

out.println(“</head>”);

out.println(“<body>”);

out.println(“<h1>Servlet UserEdit at “ + request.getContextPath() + “</h1>”);

out.println(“</body>”);

out.println(“</html>”);

}

}

@EJB

private UserDaoLocal userDao;

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

try{

int id = Integer.parseInt(request.getParameter(“id”));

User user = userDao.getById(id);

request.setAttribute(“user”, user);

RequestDispatcher requestDispatcher = request.getRequestDispatcher(“userUpdate.jsp”);

requestDispatcher.forward(request, response);

}catch (Exception e) {

throw new ServletException(e.getMessage());

}

}

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

if (!(request.getParameter(“newPass”).equals(request.getParameter(“newPass2”)))){

request.setAttribute(“error”, “Пароли не совпадают”);

RequestDispatcher requestDispatcher = request.getRequestDispatcher(“error.jsp”);

requestDispatcher.forward(request, response);

}

int id = Integer.parseInt(request.getParameter(“id”));

User user = userDao.getById(id);

if (!(user.getPassword().equals(request.getParameter(“pass1”)))){

request.setAttribute(“error”, “Старый пароль не совпал”);

RequestDispatcher requestDispatcher = request.getRequestDispatcher(“error.jsp”);

requestDispatcher.forward(request, response);

}

user.setLogin(request.getParameter(“newLogin”));

user.setEmail(request.getParameter(“newEmail”));

user.setPassword(request.getParameter(“newPass”));

userDao.editUser(user);

RequestDispatcher requestDispatcher = request.getRequestDispatcher(“index.jsp”);

requestDispatcher.forward(request, response);

}

@Override

public String getServletInfo() {

return “Short description”;

}

}

Страница изменения

<%@page contentType=”text/html” pageEncoding=”UTF-8”%>

<%@taglib uri=”http://java.sun.com/jsp/jstl/core” prefix=”c” %>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv=”Content-Type” content=”text/html; charset=UTF-8”>

<title>Редактировать</title>

<style>

</style>

</head>

<body>

<form action=”userEdit” method=”post”>

<input type=”hidden” name=”id” value=”${user.getUserID()}”>

<table>

<tr> <td>${user.getLogin()}</td>

<td><input required type=”text” name=”newLogin” /></td>

</tr>

<tr> <td>${user.getEmail()}</td>

<td><input required type=”text” name=”newEmail” /></td>

</tr>

<tr> <td>Введите старый пароль</td>

<td><input required type=”password” name=”pass1” /></td>

</tr>

<tr> <td>Введите новый пароль</td>

<td><input required type=”password” name=”newPass” /></td>

</tr>

<tr><td>Повторите пароль</td>

<td><input required type=”password” name=”newPass2” /></td>

</tr>

<tr> <td colspan=”2”><input type=”submit” value=”сохранить”></td>

</tr></table>

</form>

</body>

</html>

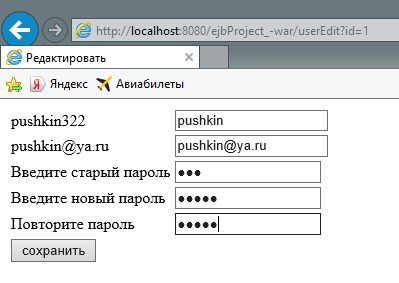


Рисунок 7. Страница с изменением данных пользователя.

Статистика

package servlet;

import java.io.IOException;

import java.io.PrintWriter;

import javax.ejb.EJB;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import ingleton.countAddUser;

import ingleton.countDeleteUser;

import ingleton.countUpdateUser;

public class statistics extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType(“text/html;charset=UTF-8”);

try (PrintWriter out = response.getWriter()) {

out.println(“<!DOCTYPE html>”);

out.println(“<html>”);

out.println(“<head>”);

out.println(“<title>Servlet statistics</title>”);

out.println(“</head>”);

out.println(“<body>”);

out.println(“<h1>Servlet statistics at “ + request.getContextPath() + “</h1>”);

out.println(“</body>”);

out.println(“</html>”);

}

}

@EJB

private countDeleteUser cdUser;

@EJB

private countUpdateUser cuUser;

@EJB

private countAddUser caUser;

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

request.setAttribute(“delete”, cdUser.getCount());

request.setAttribute(“update”, cuUser.getCount());

request.setAttribute(“add”, caUser.getCount());

RequestDispatcher requestDispatcher = request.getRequestDispatcher(“statistics.jsp”);

requestDispatcher.forward(request, response);

}

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

@Override

public String getServletInfo() {

return “Short description”;

}

}

Страничка со статистикой

<%@page contentType=”text/html” pageEncoding=”UTF-8”%>

<%@taglib uri=”http://java.sun.com/jsp/jstl/core” prefix=”c” %>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv=”Content-Type” content=”text/html; charset=UTF-8”>

<title>JSP Page</title>

</head>

<body>

<h3>${add} – кол-во добавленых пользователей</h3>

<h3>${delete} – кол-во удаленных пользователей</h3>

<h3>${update} – кол-во обновлений</h3>

</body>

</html>

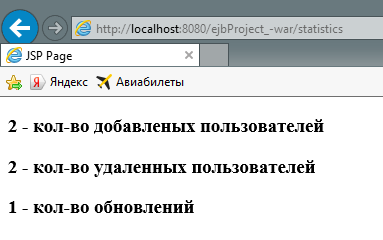


Рисунок 8. Страница статистики.

|  |
| --- |
|  |

ВЫВОД

В ходе выполнения лабораторной работы было освоено создание сессионных компонентов EJB 3.1 и использование сессионных компонентов в веб-приложении с помощью CDI, а также получены навыки работы со средой разработки NetBeans.