МИНИСТЕРСТВО образования РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

УНИВЕРСИТЕТ информатики и радиоэлектроники»

Факультет компьютерного проектирования

Контрольная работа №2

По дисциплине: «Реализация приложения c использованием технологии JSF»

Тема: «Учет выплат по кредитам»

Выполнил студент 4-го курса факультета инновационного непрерывного образования Группа [894351](https://iis.bsuir.by/#/schedule;groupName=493551)

Галкин Илья Викторович

Зачетная книжка №6535021

Дата сдачи: 13.01.2022

Проверил: Унучек Е.Н.

 Минск 2022

**Оглавление**

[1. Краткие теоретические сведения 3](#_Toc92908491)

[2. Скриншоты работы программы 4](#_Toc92908492)

[3. Листинг программного кода 6](#_Toc92908493)

# Краткие теоретические сведения

JavaServer Faces (JSF) — это стандартная технология Java для создания веб-приложений. Разработка веб-интерфейсов на Java упрощается благодаря предоставлению компонентного подхода. Это разрешает доступ к данным и логике на стороне сервера. JavaServer Faces — это, по сути, документ XML, который обозначает формальные компоненты в форме рационального дерева. JSF API предлагает такие компоненты, как commandButton, ввод текста и т. д., а также помогает в управлении их состояниями. Компоненты JSF поддерживаются объектами Java и имеют несколько средств Java, включая базы данных, библиотеки тегов и удаленные API. Библиотеки тегов JSF добавляют компоненты на веб-страницы и связывают их с объектами на сервере.

JavaServer Faces привлекает разнообразную аудиторию разработчиков, поскольку «бизнес-разработчики» считают разработку JavaServer Faces простой из-за компонентов пользовательского интерфейса с помощью перетаскивания, а «системные дизайнеры» получают беспрецедентную гибкость программирования благодаря мощному API-интерфейсу JSF. JSF использует шаблон проектирования Model-View-Controller (MVC) в своей архитектуре, обеспечивая приложениям большую удобство сопровождения.

Основная идея платформы JavaServer Faces заключается в инкапсуляции (или обертывании) клиентских технологий, таких как HTML, CSS и JavaScript, что позволяет разработчикам или дизайнерам создавать веб-интерфейсы без особого взаимодействия с технологиями. Последняя версия JSF, JSF2, использует Facelets в качестве системы шаблонов по умолчанию.

# Скриншоты работы программы

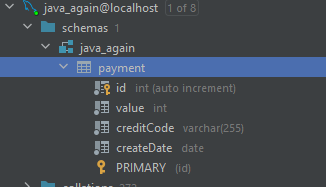


Рисунок 2 – схема базы данных приложения

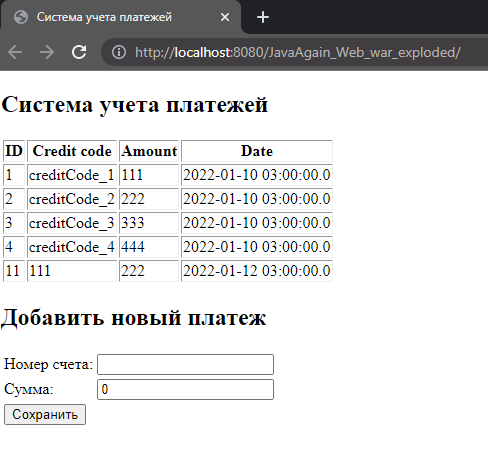


Рисунок 3 – домашняя страница приложения

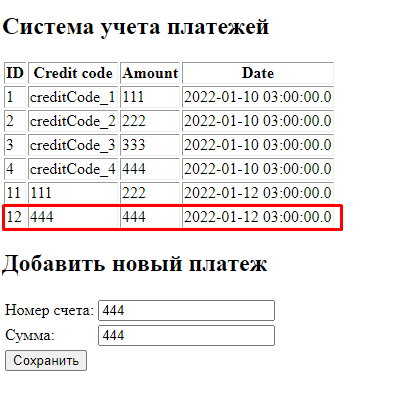


Рисунок 4 – результат добавления записи

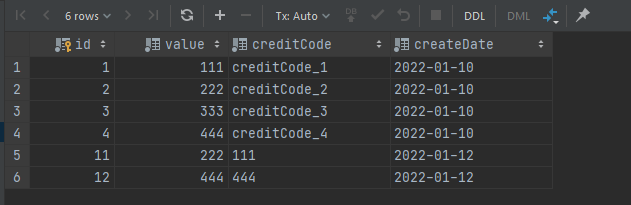


Рисунок 5 – выгрузка из базы данных

# Листинг программного кода

**PaymentBean.java**

**package** beans;

**import** **repositories.PaymentRepository**;

**import** **entity.Payment**;

**import** **utils.DateUtils**;

**import** **java.util.List**;

**public** **class** **PaymentBean** {

/\* NOTE: Uncomment if server is TomEE

@EJB

\*/

**private** **final** PaymentRepository paymentRepository;

**private** Payment workingModel;

**public** **PaymentBean**() {

paymentRepository = **new** PaymentRepository();

workingModel = **new** Payment();

}

**public** List<Payment> **getPayments**() {

**return** paymentRepository.getAllPayments();

}

// NOTE: working model was already set from the front

**public** String **savePayment**() {

**if** (!validateRequest()) {

**return** "index";

}

workingModel.setCreateDate(DateUtils.getUTCdatetimeAsDate());

paymentRepository.addPayment(workingModel);

**return** "index";

}

**private** **boolean** **validateRequest**() {

**final** String code = workingModel.getCreditCode();

**return** code != **null** && code.length() > **0**;

}

**public** Payment **getWorkingModel**() {

**return** workingModel;

}

**public** **void** **setWorkingModel**(Payment workingModel) {

**this**.workingModel = workingModel;

}

}

**PaymentRepository.java**

**package** repositories;

**import** **entity.Payment**;

**import** **javax.ejb.Stateless**;

**import** **javax.persistence.EntityManager**;

**import** **javax.persistence.EntityManagerFactory**;

**import** **javax.persistence.Persistence**;

**import** **javax.persistence.criteria.CriteriaQuery**;

**import** **java.util.ArrayList**;

/\*\*

\*

\* @author ILans

\*/

**@Stateless**

**public** **class** **PaymentRepository** {

**public** EntityManager **getEntityManager**() {

EntityManagerFactory emf = Persistence.createEntityManagerFactory("unit");

**return** emf.createEntityManager();

}

**public** **void** **addPayment**(Payment payment) {

EntityManager em = getEntityManager();

em.getTransaction().begin();

**if** (!em.contains(payment)) {

em.persist(payment);

em.flush();

}

em.getTransaction().commit();

}

**public** ArrayList<Payment> **getAllPayments**() {

EntityManager entityManager = getEntityManager();

CriteriaQuery criteriaQuery = entityManager.getCriteriaBuilder().createQuery();

criteriaQuery.select(criteriaQuery.from(Payment.class));

**return** **new** ArrayList<Payment>(entityManager.createQuery(criteriaQuery).getResultList());

}

}

**Index.xhtml**

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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"

xmlns:f="http://java.sun.com/jsf/core"

xmlns:h="http://java.sun.com/jsf/html">

<h:head>

<title>Система учета платежей</title>

</h:head>

<h:body>

<h:form id="id-form" >

<h2><h:outputText value ="Система учета платежей"></h:outputText></h2>

<h:dataTable style="border: 4px;" value="#{paymentBean.payments}" var = "item" align="center" border="1" >

<h:column>

<f:facet name="header"> ID </f:facet>

<h:outputText value="#{item.id}" />

</h:column>

<h:column>

<f:facet name="header"> Credit code </f:facet>

<h:outputText value="#{item.creditCode}" />

</h:column>

<h:column>

<f:facet name="header" > Amount </f:facet>

<h:outputText value="#{item.value}" />

</h:column>

<h:column>

<f:facet name="header"> Date </f:facet>

<h:outputText value="#{item.createDate}" />

</h:column>

</h:dataTable>

</h:form>

<h:form>

<h2><h:outputText value ="Добавить новый платеж"></h:outputText></h2>

<h:panelGrid columns="3">

<h:outputLabel for="#{paymentBean.workingModel.creditCode}">Номер счета:</h:outputLabel>

<h:inputText value="#{paymentBean.workingModel.creditCode}" id="cname"></h:inputText>

<br/>

<h:outputLabel for="#{paymentBean.workingModel.value}">Сумма:</h:outputLabel>

<h:inputText value="#{paymentBean.workingModel.value}"></h:inputText>

<br/>

<h:commandButton action="#{paymentBean.savePayment}" value="Сохранить"></h:commandButton>

</h:panelGrid>

</h:form>

</h:body>

</html>

**web.xml**

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

<web-app 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\_4\_0.xsd"

version="4.0">

<welcome-file-list>

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

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

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

<welcome-file>default.html</welcome-file>

<welcome-file>default.htm</welcome-file>

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

</welcome-file-list>

<servlet>

<servlet-name>faces</servlet-name>

<servlet-class>javax.faces.webapp.FacesServlet</servlet-class>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>faces</servlet-name>

<url-pattern>\*.xhtml</url-pattern>

</servlet-mapping>

</web-app>

**Db\_create.sql**

**CREATE** **TABLE** IF **NOT** **EXISTS** `payment` (

id INT AUTO\_INCREMENT **PRIMARY** **KEY**,

value INT **NOT** **NULL**,

creditCode VARCHAR(**255**) **NOT** **NULL**,

createDate DATE **NOT** **NULL**

);

-- test data

**insert** **into** payment(value, creditCode, createDate)

**values**

(**111**, "creditCode\_1", now()),

(**222**, "creditCode\_2", now()),

(**333**, "creditCode\_3", now()),

(**444**, "creditCode\_4", now())