**package** com.online.garments.deal.bean;

**import** java.sql.Timestamp;

**public** **abstract** **class** BaseBean **implements** DropdownListBean, Comparable<BaseBean> {

**protected** **long** id;

**protected** String createdBy;

**protected** String modifiedBy;

**protected** Timestamp createdDatetime;

**protected** Timestamp modifiedDatetime;

**public** **long** getId() {

**return** id;

}

**public** **void** setId(**long** id) {

**this**.id = id;

}

**public** String getCreatedBy() {

**return** createdBy;

}

**public** **void** setCreatedBy(String createdBy) {

**this**.createdBy = createdBy;

}

**public** String getModifiedBy() {

**return** modifiedBy;

}

**public** **void** setModifiedBy(String modifiedBy) {

**this**.modifiedBy = modifiedBy;

}

**public** Timestamp getCreatedDatetime() {

**return** createdDatetime;

}

**public** **void** setCreatedDatetime(Timestamp createdDatetime) {

**this**.createdDatetime = createdDatetime;

}

**public** Timestamp getModifiedDatetime() {

**return** modifiedDatetime;

}

**public** **void** setModifiedDatetime(Timestamp modifiedDatetime) {

**this**.modifiedDatetime = modifiedDatetime;

}

**public** **int** compareTo(BaseBean next) {

// **TODO** Auto-generated method stub

**return** getValue().compareTo(next.getValue());

}

}

**package** com.online.garments.deal.bean;

**public** **class** CustomerBean **extends** BaseBean {

**private** String Name;

**private** String age;

**private** String address;

**private** String contectNo;

**private** String productChoice;

**private** String itemCode;

**private** String MultipleItems;

**private** String login;

**private** String password;

**private** String confirmPassword;

**public** String getPassword() {

**return** password;

}

**public** **void** setPassword(String password) {

**this**.password = password;

}

**public** String getConfirmPassword() {

**return** confirmPassword;

}

**public** **void** setConfirmPassword(String confirmPassword) {

**this**.confirmPassword = confirmPassword;

}

**public** String getLogin() {

**return** login;

}

**public** **void** setLogin(String login) {

**this**.login = login;

}

**public** String getName() {

**return** Name;

}

**public** **void** setName(String name) {

Name = name;

}

**public** String getAge() {

**return** age;

}

**public** **void** setAge(String age) {

**this**.age = age;

}

**public** String getAddress() {

**return** address;

}

**public** **void** setAddress(String address) {

**this**.address = address;

}

**public** String getContectNo() {

**return** contectNo;

}

**public** **void** setContectNo(String contectNo) {

**this**.contectNo = contectNo;

}

**public** String getProductChoice() {

**return** productChoice;

}

**public** **void** setProductChoice(String productChoice) {

**this**.productChoice = productChoice;

}

**public** String getItemCode() {

**return** itemCode;

}

**public** **void** setItemCode(String itemCode) {

**this**.itemCode = itemCode;

}

**public** String getMultipleItems() {

**return** MultipleItems;

}

**public** **void** setMultipleItems(String multipleItems) {

MultipleItems = multipleItems;

}

@Override

**public** String getKey() {

// **TODO** Auto-generated method stub

**return** id+"";

}

@Override

**public** String getValue() {

// **TODO** Auto-generated method stub

**return** Name;

}

}

**package** com.online.garments.deal.bean;

**public** **interface** DropdownListBean

{

**public** String getKey();

**public** String getValue();

}

**package** com.online.garments.deal.bean;

**public** **class** LoginBean **extends** BaseBean {

**private** String login;

**private** String password;

**private** **long** role;

**private** String confirmPassword;

**public** String getConfirmPassword() {

**return** confirmPassword;

}

**public** **void** setConfirmPassword(String confirmPassword) {

**this**.confirmPassword = confirmPassword;

}

**public** **long** getRole() {

**return** role;

}

**public** **void** setRole(**long** role) {

**this**.role = role;

}

**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;

}

@Override

**public** String getKey() {

// **TODO** Auto-generated method stub

**return** id+"";

}

@Override

**public** String getValue() {

// **TODO** Auto-generated method stub

**return** login;

}

}

**package** com.online.garments.deal.bean;

**public** **class** ManagerBean **extends** BaseBean {

**private** String Name;

**private** String emailId;

**private** String contactNo;

**private** String address;

**private** String login;

**private** String password;

**private** **long** role;

**private** String confirmPassword;

**public** String getConfirmPassword() {

**return** confirmPassword;

}

**public** **void** setConfirmPassword(String confirmPassword) {

**this**.confirmPassword = confirmPassword;

}

**public** String getName() {

**return** Name;

}

**public** **void** setName(String name) {

Name = name;

}

**public** String getEmailId() {

**return** emailId;

}

**public** **void** setEmailId(String emailId) {

**this**.emailId = emailId;

}

**public** String getContactNo() {

**return** contactNo;

}

**public** **void** setContactNo(String contactNo) {

**this**.contactNo = contactNo;

}

**public** String getAddress() {

**return** address;

}

**public** **void** setAddress(String address) {

**this**.address = address;

}

**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** **long** getRole() {

**return** role;

}

**public** **void** setRole(**long** role) {

**this**.role = role;

}

@Override

**public** String getKey() {

// **TODO** Auto-generated method stub

**return** id+"";

}

@Override

**public** String getValue() {

// **TODO** Auto-generated method stub

**return** Name;

}

}

**package** com.online.garments.deal.bean;

**import** java.util.Date;

**public** **class** PaymentBean **extends** BaseBean {

**private** **long** paymentId;

**private** **long** customerId;

**private** **long** productId;

**private** Date paymentDate;

**private** **double** amount;

**private** String login;

**public** String getLogin() {

**return** login;

}

**public** **void** setLogin(String login) {

**this**.login = login;

}

**public** **long** getPaymentId() {

**return** paymentId;

}

**public** **void** setPaymentId(**long** paymentId) {

**this**.paymentId = paymentId;

}

**public** **long** getCustomerId() {

**return** customerId;

}

**public** **void** setCustomerId(**long** customerId) {

**this**.customerId = customerId;

}

**public** **long** getProductId() {

**return** productId;

}

**public** **void** setProductId(**long** productId) {

**this**.productId = productId;

}

**public** Date getPaymentDate() {

**return** paymentDate;

}

**public** **void** setPaymentDate(Date paymentDate) {

**this**.paymentDate = paymentDate;

}

**public** **double** getAmount() {

**return** amount;

}

**public** **void** setAmount(**double** amount) {

**this**.amount = amount;

}

@Override

**public** String getKey() {

// **TODO** Auto-generated method stub

**return** **null**;

}

@Override

**public** String getValue() {

// **TODO** Auto-generated method stub

**return** **null**;

}

}

**package** com.online.garments.deal.bean;

**public** **class** ProductBean **extends** BaseBean{

**private** String productName;

**private** String productQuantity;

**private** String productChoice;

**private** String itemCode;

**private** String image;

**private** **double** price;

**public** **double** getPrice() {

**return** price;

}

**public** **void** setPrice(**double** price) {

**this**.price = price;

}

**public** String getImage() {

**return** image;

}

**public** **void** setImage(String image) {

**this**.image = image;

}

**public** String getProductName() {

**return** productName;

}

**public** **void** setProductName(String productName) {

**this**.productName = productName;

}

**public** String getProductQuantity() {

**return** productQuantity;

}

**public** **void** setProductQuantity(String productQuantity) {

**this**.productQuantity = productQuantity;

}

**public** String getProductChoice() {

**return** productChoice;

}

**public** **void** setProductChoice(String productChoice) {

**this**.productChoice = productChoice;

}

**public** String getItemCode() {

**return** itemCode;

}

**public** **void** setItemCode(String itemCode) {

**this**.itemCode = itemCode;

}

@Override

**public** String getKey() {

// **TODO** Auto-generated method stub

**return** id+"";

}

@Override

**public** String getValue() {

// **TODO** Auto-generated method stub

**return** productName;

}

}

package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.DataValidator;

import com.online.garments.deal.util.ServletUtility;

@WebServlet("/BaseCtl")

public abstract class BaseCtl extends HttpServlet

{

private static final Logger log=Logger.getLogger(BaseCtl.class);

public static final String OP\_SAVE = "Save";

public static final String OP\_CANCEL = "Cancel";

public static final String OP\_DELETE = "Delete";

public static final String OP\_LIST = "List";

public static final String OP\_SEARCH = "Search";

public static final String OP\_VIEW = "View";

public static final String OP\_NEXT = "Next";

public static final String OP\_PREVIOUS = "Previous";

public static final String OP\_NEW = "New";

public static final String OP\_GO = "Go";

public static final String OP\_BACK = "Back";

public static final String OP\_LOG\_OUT = "Logout";

public static final String OP\_RESET = "Reset";

public static final String OP\_PAYMENT = "Payment";

public static final String MSG\_SUCCESS = "success";

public static final String MSG\_ERROR = "error";

public BaseCtl() {

// TODO Auto-generated constructor stub

}

protected boolean validate(HttpServletRequest request) {

return true;

}

protected void preload(HttpServletRequest request) {

}

protected BaseBean populateBean(HttpServletRequest request) {

return null;

}

protected BaseBean populateDTO(BaseBean dto, HttpServletRequest request) {

log.debug("BaseCtl populate DTO method start");

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

String modifiedBy = null;

LoginBean userbean = (LoginBean) request.getSession().getAttribute("user");

if (userbean == null) {

// If record is created without login

createdBy = "root";

modifiedBy = "root";

} else {

modifiedBy = userbean.getLogin();

// If record is created first time

if ("null".equalsIgnoreCase(createdBy)|| DataValidator.isNull(createdBy)) {

createdBy = modifiedBy;

}

}

dto.setCreatedBy(createdBy);

dto.setModifiedBy(modifiedBy);

long cdt = DataUtility.getLong(request.getParameter("createdDatetime"));

if (cdt > 0) {

dto.setCreatedDatetime(DataUtility.getTimestamp(cdt));

} else {

dto.setCreatedDatetime(DataUtility.getCurrentTimestamp());

}

dto.setModifiedDatetime(DataUtility.getCurrentTimestamp());

log.debug("BaseCtl populate DTO method end");

return dto;

}

@Override

protected void service(HttpServletRequest request,HttpServletResponse response) throws ServletException, IOException {

log.debug("BaseCtl service method start");

// Load the preloaded data required to display at HTML form

preload(request);

String op = DataUtility.getString(request.getParameter("operation"));

// Check if operation is not DELETE, VIEW, CANCEL, and NULL then

// perform input data validation

System.out.println("operation ="+op);

if (DataValidator.isNotNull(op) && !OP\_CANCEL.equalsIgnoreCase(op)&& !OP\_VIEW.equalsIgnoreCase(op)&& !OP\_DELETE.equalsIgnoreCase(op)&&!OP\_RESET.equalsIgnoreCase(op)) {

// Check validation, If fail then send back to page with error

// messages

if (!validate(request)) {

BaseBean bean = (BaseBean) populateBean(request);

ServletUtility.setBean(bean, request);

ServletUtility.forward(getView(), request, response);

return;

}

}

log.debug("BaseCtl service method end");

super.service(request, response);

}

protected abstract String getView();

}package com.online.garments.deal.controller;

import java.io.IOException;

import java.util.List;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.bean.PaymentBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.model.PaymentModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "BookListCtl", urlPatterns = { "/ctl/BookListCtl" })

public class BookListCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

private static Logger log = Logger.getLogger(PaymentListCtl.class);

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("PaymentListCtl populateBean method start");

PaymentBean bean = new PaymentBean();

bean.setProductId(DataUtility.getLong(request.getParameter("Pro")));

log.debug("PaymentListCtl populateBean method end");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("PaymentListCtl doGet method start");

List list = null;

int pageNo = 1;

int pageSize = DataUtility.getInt(PropertyReader.getValue("page.size"));

long idd=DataUtility.getLong(request.getParameter("idd"));

PaymentModel model = new PaymentModel();

PaymentBean bean = (PaymentBean) populateBean(request);

try {

if(idd>0) {

PaymentBean pBean=new PaymentBean();

pBean.setId(idd);

model.delete(pBean);

ServletUtility.setSuccessMessage("Data Deleted Succsessfully", request);

}

HttpSession session=request.getSession();

LoginBean lBean=(LoginBean)session.getAttribute("user");

if(lBean.getRole()==3) {

bean.setLogin(lBean.getLogin());

}

list = model.search(bean, pageNo, pageSize);

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("No Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("PaymentListCtl doGet method end");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("PaymentListCtl doPost method start");

List list = null;

int pageNo = DataUtility.getInt(request.getParameter("pageNo"));

int pageSize = DataUtility.getInt(request.getParameter("pageSize"));

pageNo = (pageNo == 0) ? 1 : pageNo;

pageSize = (pageSize == 0) ? DataUtility.getInt(PropertyReader.getValue("page.size")) : pageSize;

PaymentBean bean = (PaymentBean) populateBean(request);

PaymentModel model = new PaymentModel();

String[] ids = request.getParameterValues("ids");

String op = DataUtility.getString(request.getParameter("operation"));

if (OP\_SEARCH.equalsIgnoreCase(op) || OP\_NEXT.equalsIgnoreCase(op) || OP\_PREVIOUS.equalsIgnoreCase(op)) {

if (OP\_SEARCH.equalsIgnoreCase(op)) {

pageNo = 1;

} else if (OP\_NEXT.equalsIgnoreCase(op)) {

pageNo++;

} else if (OP\_PREVIOUS.equalsIgnoreCase(op) && pageNo > 1) {

pageNo--;

}

} else if (OP\_NEW.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

return;

} else if (OP\_DELETE.equalsIgnoreCase(op)) {

pageNo = 1;

if (ids != null && ids.length > 0) {

PaymentBean deletebean = new PaymentBean();

for (String id : ids) {

deletebean.setId(DataUtility.getInt(id));

try {

model.delete(deletebean);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

}

ServletUtility.setSuccessMessage("Data Deleted Successfully", request);

} else {

ServletUtility.setErrorMessage("Select at least one record", request);

}

} else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.BOOK\_LIST\_CTL, request, response);

return;

}

try {

HttpSession session=request.getSession();

LoginBean lBean=(LoginBean)session.getAttribute("user");

if(lBean.getRole()==3) {

bean.setLogin(lBean.getLogin());

}

list = model.search(bean, pageNo, pageSize);

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("NO Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("PaymentListCtl doPost method end");

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.BOOK\_LIST\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.CustomerBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.bean.ManagerBean;

import com.online.garments.deal.bean.PaymentBean;

import com.online.garments.deal.bean.ProductBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.model.CustomerModel;

import com.online.garments.deal.model.LoginModel;

import com.online.garments.deal.model.ManagerModel;

import com.online.garments.deal.model.PaymentModel;

import com.online.garments.deal.model.ProductModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.DataValidator;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "CustomerCtl", urlPatterns = { "/CustomerCtl" })

public class CustomerCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

private static Logger log = Logger.getLogger(CustomerCtl.class);

@Override

protected boolean validate(HttpServletRequest request) {

log.debug("CustomerCtl validate method start");

boolean pass = true;

if (DataValidator.isNull(request.getParameter("name"))) {

request.setAttribute("name",PropertyReader.getValue("error.require", "Name"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("contact")))

{

request.setAttribute("contact",PropertyReader.getValue("error.require", "Contact No"));

pass = false;

}else if(!DataValidator.isPhoneNo(request.getParameter("contact")))

{

request.setAttribute("contact", PropertyReader.getValue("error.invalid","Contact No"));

pass=false;

}

if (DataValidator.isNull(request.getParameter("age")))

{

request.setAttribute("age",PropertyReader.getValue("error.require", "Age"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("address")))

{

request.setAttribute("address",PropertyReader.getValue("error.require", "Address"));

pass = false;

}

String op=DataUtility.getString(request.getParameter("operation"));

HttpSession session=request.getSession();

LoginBean lBean=(LoginBean)session.getAttribute("user");

if(lBean==null) {

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

if (DataValidator.isNull(login)) {

request.setAttribute("login", PropertyReader.getValue("error.require", "Login Id"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("password"))) {

request.setAttribute("password", PropertyReader.getValue("error.require", "Password"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("confirmPassword"))) {

request.setAttribute("confirmPassword", PropertyReader.getValue("error.require", "Confirm Password"));

pass = false;

}

if(!OP\_PAYMENT.equalsIgnoreCase(op)) {

if (!request.getParameter("password").equals(

request.getParameter("confirmPassword"))

&& !"".equals(request.getParameter("confirmPassword"))) {

/\*ServletUtility.setErrorMessage("Confirm Password did not match",

request);\*/

request.setAttribute("confirmPassword", PropertyReader.getValue("error.confirmPassword","Confirm Password"));

pass = false;

}

}

}

if(OP\_PAYMENT.equalsIgnoreCase(op)) {

pass=true;

}

log.debug("CustomerCtl validate method end");

return pass;

}

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("CustomerCtl Method populatebean Started");

CustomerBean bean = new CustomerBean();

bean.setId(DataUtility.getLong(request.getParameter("id")));

bean.setName(DataUtility.getString(request.getParameter("name")));

bean.setAge(DataUtility.getString(request.getParameter("age")));

bean.setContectNo(DataUtility.getString(request.getParameter("contact")));

bean.setAddress(DataUtility.getString(request.getParameter("address")));

HttpSession session=request.getSession();

LoginBean lBean=(LoginBean)session.getAttribute("user");

if(lBean==null) {

bean.setLogin(DataUtility.getString(request.getParameter("login")));

bean.setPassword(DataUtility.getString(request.getParameter("password")));

bean.setConfirmPassword(DataUtility.getString(request

.getParameter("confirmPassword")));

}else {

bean.setLogin(lBean.getLogin());

bean.setPassword(lBean.getPassword());

}

populateDTO(bean, request);

log.debug("CustomerCtl Method populatebean Ended");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("CustomerCtl Method doGet Started");

System.out.println("in do get");

String op = DataUtility.getString(request.getParameter("operation"));

long id = DataUtility.getLong(request.getParameter("id"));

HttpSession session=request.getSession(true);

long pid=DataUtility.getLong(request.getParameter("pid"));

if(pid==0) {

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

}else {

session.setAttribute("pid", pid);

ServletUtility.setOpration("Add", request);

CustomerModel model = new CustomerModel();

if (id > 0 || op != null) {

CustomerBean bean;

try {

bean = model.findByPk(id);

ServletUtility.setOpration("Edit", request);

ServletUtility.setBean(bean, request);

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

}

ServletUtility.forward(getView(), request, response);

}

log.debug("CustomerCtl Method doGett Ended");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("CustomerCtl Method doPost Started");

System.out.println("in do post");

String op = DataUtility.getString(request.getParameter("operation"));

// get model

HttpSession session=request.getSession();

CustomerModel model = new CustomerModel();

long id = DataUtility.getLong(request.getParameter("id"));

ServletUtility.setOpration("Add", request);

if (OP\_SAVE.equalsIgnoreCase(op)) {

CustomerBean bean = (CustomerBean) populateBean(request);

try {

if (id > 0) {

model.update(bean);

ServletUtility.setSuccessMessage("Data is successfully Updated",request);

ServletUtility.setOpration("Edit", request);

ServletUtility.setBean(bean, request);

} else {

session.setAttribute("cusBean", bean);

ServletUtility.forward(OGDView.PAYMENT\_VIEW, request, response);

return;

}

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

} catch (DuplicateRecordException e) {

ServletUtility.setBean(bean, request);

ServletUtility.setErrorMessage(e.getMessage(), request);

ServletUtility.forward(getView(), request, response);

return;

}

}

else if (OP\_DELETE.equalsIgnoreCase(op)) {

CustomerBean bean = (CustomerBean) populateBean(request);

try {

model.delete(bean);

ServletUtility.redirect(OGDView.CUSTOMER\_LIST\_CTL, request,response);

return;

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

} else if (OP\_CANCEL.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.CUSTOMER\_LIST\_CTL, request, response);

return;

}else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.CUSTOMER\_CTL, request, response);

return;

}else if(OP\_PAYMENT.equalsIgnoreCase(op)) {

CustomerBean cBean=(CustomerBean)session.getAttribute("cusBean");

ProductModel pModel=new ProductModel();

long pId=(long)session.getAttribute("pid");

try {

LoginBean lBean=(LoginBean)session.getAttribute("user");

if(lBean==null) {

LoginModel lModel=new LoginModel();

LoginBean lbBean=new LoginBean();

lbBean.setLogin(cBean.getLogin());

lbBean.setPassword(cBean.getPassword());

lbBean.setRole(3L);

lModel.add(lbBean);

}

ProductBean pBean=pModel.findByPk(pId);

cBean.setItemCode(pBean.getItemCode());

if(lBean!=null) {

cBean.setLogin(lBean.getLogin());

}

long pk=model.add(cBean);

PaymentModel pyModel=new PaymentModel();

PaymentBean pyBean=new PaymentBean();

pyBean.setCustomerId(pk);

pyBean.setProductId(pId);

pyBean.setAmount(pBean.getPrice());

pyBean.setLogin(cBean.getLogin());

if(lBean!=null) {

pyBean.setLogin(lBean.getLogin());

}

pyModel.add(pyBean);

ServletUtility.setSuccessMessage("Payment Done Successfully", request);

ServletUtility.forward(OGDView.SUCCESS\_VIEW, request, response);

return;

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

} catch (DuplicateRecordException e) {

ServletUtility.setErrorMessage(e.getMessage(), request);

}

}

log.debug("CustomerCtl Method doPost Ended");

ServletUtility.forward(getView(), request, response);

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.CUSTOMER\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import java.util.List;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.CustomerBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.model.CustomerModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "CustomerListCtl", urlPatterns = { "/ctl/CustomerListCtl" })

public class CustomerListCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

private static Logger log = Logger.getLogger(CustomerListCtl.class);

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("CustomerListCtl populateBean method start");

CustomerBean bean = new CustomerBean();

bean.setName(DataUtility.getString(request.getParameter("name")));

bean.setItemCode(DataUtility.getString(request.getParameter("code")));

log.debug("CustomerListCtl populateBean method end");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("CustomerListCtl doGet method start");

List list = null;

int pageNo = 1;

int pageSize = DataUtility.getInt(PropertyReader.getValue("page.size"));

long idd=DataUtility.getLong(request.getParameter("idd"));

CustomerModel model = new CustomerModel();

CustomerBean bean = (CustomerBean) populateBean(request);

try {

if(idd>0) {

CustomerBean pBean=new CustomerBean();

pBean.setId(idd);

model.delete(pBean);

ServletUtility.setSuccessMessage("Data Deleted Succsessfully", request);

}

list = model.search(bean, pageNo, pageSize);

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("No Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("CustomerListCtl doGet method end");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("CustomerListCtl doPost method start");

List list = null;

int pageNo = DataUtility.getInt(request.getParameter("pageNo"));

int pageSize = DataUtility.getInt(request.getParameter("pageSize"));

pageNo = (pageNo == 0) ? 1 : pageNo;

pageSize = (pageSize == 0) ? DataUtility.getInt(PropertyReader.getValue("page.size")) : pageSize;

CustomerBean bean = (CustomerBean) populateBean(request);

CustomerModel model = new CustomerModel();

String[] ids = request.getParameterValues("ids");

String op = DataUtility.getString(request.getParameter("operation"));

if (OP\_SEARCH.equalsIgnoreCase(op) || OP\_NEXT.equalsIgnoreCase(op) || OP\_PREVIOUS.equalsIgnoreCase(op)) {

if (OP\_SEARCH.equalsIgnoreCase(op)) {

pageNo = 1;

} else if (OP\_NEXT.equalsIgnoreCase(op)) {

pageNo++;

} else if (OP\_PREVIOUS.equalsIgnoreCase(op) && pageNo > 1) {

pageNo--;

}

} else if (OP\_NEW.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

return;

} else if (OP\_DELETE.equalsIgnoreCase(op)) {

pageNo = 1;

if (ids != null && ids.length > 0) {

CustomerBean deletebean = new CustomerBean();

for (String id : ids) {

deletebean.setId(DataUtility.getInt(id));

try {

model.delete(deletebean);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

}

ServletUtility.setSuccessMessage("Data Deleted Successfully", request);

} else {

ServletUtility.setErrorMessage("Select at least one record", request);

}

} else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.CUSTOMER\_LIST\_CTL, request, response);

return;

}

try {

list = model.search(bean, pageNo, pageSize);

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("NO Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("CustomerListCtl doPost method end");

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.CUSTOMER\_LIST\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.Filter;

import javax.servlet.FilterChain;

import javax.servlet.FilterConfig;

import javax.servlet.ServletException;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import javax.servlet.annotation.WebFilter;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.log4j.Logger;

import com.online.garments.deal.util.ServletUtility;

@WebFilter(filterName="FrontCtl",urlPatterns={"/ctl/\*","/doc/\*"})

public class FrontController implements Filter {

private static Logger log=Logger.getLogger(FrontController.class);

public void destroy() {

}

public void doFilter(ServletRequest req, ServletResponse resp,FilterChain chain) throws IOException, ServletException {

log.debug("FrontController doFilter method start");

HttpServletRequest request = (HttpServletRequest) req;

HttpServletResponse response = (HttpServletResponse) resp;

HttpSession session = request.getSession(true);

if (session.getAttribute("user") == null) {

ServletUtility.setErrorMessage("Your session has been expired. Please re-Login.", request);

String hitUri=request.getRequestURI();

req.setAttribute("uri", hitUri);

ServletUtility.forward("/LoginCtl", request, response);

} else {

chain.doFilter(req, resp);

System.out.println("respronse front coltoller");

}

log.debug("FrontController doFilter method end");

}

public void init(FilterConfig conf) throws ServletException {

// TODO Auto-generated method stub

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.model.LoginModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.DataValidator;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "LoginCtl", urlPatterns = { "/LoginCtl" })

public class LoginCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

public static final String OP\_REGISTER = "Register";

public static final String OP\_SIGN\_IN = "SignIn";

public static final String OP\_SIGN\_UP = "SignUp";

public static final String OP\_LOG\_OUT = "logout";

public static String HIT\_URI = null;

private static Logger log = Logger.getLogger(LoginCtl.class);

public LoginCtl() {

super();

// TODO Auto-generated constructor stub

}

@Override

protected boolean validate(HttpServletRequest request) {

log.debug("LoginCtl Method validate Started");

boolean pass = true;

String op = request.getParameter("operation");

if (OP\_SIGN\_UP.equals(op) || OP\_LOG\_OUT.equals(op)) {

return pass;

}

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

if (DataValidator.isNull(login)) {

request.setAttribute("login", PropertyReader.getValue("error.require", "Login Id"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("password"))) {

request.setAttribute("password", PropertyReader.getValue("error.require", "Password"));

pass = false;

}

log.debug("LoginCtl Method validate Ended");

return pass;

}

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("LoginCtl Method populateBean Started");

LoginBean bean = new LoginBean();

bean.setId(DataUtility.getLong(request.getParameter("id")));

bean.setLogin(DataUtility.getString(request.getParameter("login")));

bean.setPassword(DataUtility.getString(request.getParameter("password")));

log.debug("LOginCtl Method PopulatedBean End");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

log.debug("Method doGet Started");

HttpSession session = request.getSession(true);

String op = DataUtility.getString(request.getParameter("operation"));

LoginModel model = new LoginModel();

long id = DataUtility.getLong(request.getParameter("id"));

if (id > 0) {

LoginBean loginBean;

try {

loginBean = model.findByPK(id);

ServletUtility.setBean(loginBean, request);

} catch (Exception e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

} else if (OP\_LOG\_OUT.equals(op)) {

session = request.getSession(false);

session.invalidate();

ServletUtility.setSuccessMessage("You have been logged out successfully", request);

ServletUtility.forward(OGDView.LOGIN\_VIEW, request, response);

return;

}

if (session.getAttribute("user") != null) {

ServletUtility.redirect(OGDView.WELCOME\_CTL, request, response);

return;

}

ServletUtility.forward(getView(), request, response);

log.debug("Method doGet end");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

log.debug(" LoginCtl Method doPost Started");

HttpSession session = request.getSession(true);

String op = DataUtility.getString(request.getParameter("operation"));

// get Model

LoginModel model = new LoginModel();

long id = DataUtility.getLong(request.getParameter("id"));

if (OP\_SIGN\_IN.equalsIgnoreCase(op)) {

LoginBean bean = (LoginBean) populateBean(request);

try {

bean = model.authenticate(bean.getLogin(), bean.getPassword());

if (bean != null) {

session.setAttribute("user", bean);

// save state of session remember URL

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

if (uri == null || "null".equalsIgnoreCase(uri)) {

ServletUtility.redirect(OGDView.WELCOME\_CTL, request, response);

return;

} else {

ServletUtility.redirect(uri, request, response);

}

return;

} else {

bean = (LoginBean) populateBean(request);

ServletUtility.setBean(bean, request);

ServletUtility.setErrorMessage("Invalid LoginId And Password", request);

}

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

} else if (OP\_SIGN\_UP.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.USER\_REGISTRATION\_CTL, request, response);

return;

}

ServletUtility.forward(getView(), request, response);

log.debug("UserCtl Method doPost Ended");

}

@Override

protected String getView() {

return OGDView.LOGIN\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.bean.ManagerBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.model.LoginModel;

import com.online.garments.deal.model.ManagerModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.DataValidator;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "ManagerCtl", urlPatterns = { "/ctl/ManagerCtl" })

public class ManagerCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

private static Logger log = Logger.getLogger(ManagerCtl.class);

@Override

protected boolean validate(HttpServletRequest request) {

log.debug("ManagerCtl validate method start");

boolean pass = true;

if (DataValidator.isNull(request.getParameter("name"))) {

request.setAttribute("name",PropertyReader.getValue("error.require", "Name"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("contact")))

{

request.setAttribute("contact",PropertyReader.getValue("error.require", "Contact No"));

pass = false;

}else if(!DataValidator.isPhoneNo(request.getParameter("contact")))

{

request.setAttribute("contact", PropertyReader.getValue("error.invalid","Contact No"));

pass=false;

}

if (DataValidator.isNull(request.getParameter("email")))

{

request.setAttribute("email",PropertyReader.getValue("error.require", "Email "));

pass = false;

} else if (!DataValidator.isEmail(request.getParameter("email"))) {

request.setAttribute("email",PropertyReader.getValue("error.email", "Email "));

pass = false;

}

if (DataValidator.isNull(request.getParameter("address")))

{

request.setAttribute("address",PropertyReader.getValue("error.require", "Address"));

pass = false;

}

if("-----Select-----".equalsIgnoreCase(request.getParameter("role"))){

request.setAttribute("role",PropertyReader.getValue("error.require", "Role"));

pass=false;

}

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

if (DataValidator.isNull(login)) {

request.setAttribute("login",

PropertyReader.getValue("error.require", "Login Id"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("password"))) {

request.setAttribute("password",

PropertyReader.getValue("error.require", "Password"));

pass = false;

} else if (!DataValidator.isPassword(request.getParameter("password"))) {

request.setAttribute("password",

PropertyReader.getValue("error.password", "Password"));

return false;

}else if (!DataValidator.isPassword(request.getParameter("password"))) {

request.setAttribute("password",

PropertyReader.getValue("error.password", "Password"));

return false;

}

if (!request.getParameter("password").equals(

request.getParameter("confirmPassword"))

&& !"".equals(request.getParameter("confirmPassword"))) {

/\*ServletUtility.setErrorMessage("Confirm Password did not match",

request);\*/

request.setAttribute("confirmPassword", PropertyReader.getValue("error.confirmPassword","Confirm Password"));

pass = false;

}

log.debug("ManagerCtl validate method end");

return pass;

}

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("ManagerCtl Method populatebean Started");

ManagerBean bean = new ManagerBean();

bean.setId(DataUtility.getLong(request.getParameter("id")));

bean.setName(DataUtility.getString(request.getParameter("name")));

bean.setRole(DataUtility.getLong(request.getParameter("role")));

bean.setContactNo(DataUtility.getString(request.getParameter("contact")));

bean.setEmailId(DataUtility.getString(request.getParameter("email")));

bean.setAddress(DataUtility.getString(request.getParameter("address")));

bean.setLogin(DataUtility.getString(request.getParameter("login")));

bean.setPassword(DataUtility.getString(request.getParameter("password")));

bean.setConfirmPassword(DataUtility.getString(request

.getParameter("confirmPassword")));

populateDTO(bean, request);

log.debug("ManagerCtl Method populatebean Ended");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("ManagerCtl Method doGet Started");

System.out.println("in do get");

String op = DataUtility.getString(request.getParameter("operation"));

long id = DataUtility.getLong(request.getParameter("id"));

// get model

ServletUtility.setOpration("Add", request);

ManagerModel model = new ManagerModel();

if (id > 0 || op != null) {

ManagerBean bean;

try {

bean = model.findByPk(id);

ServletUtility.setOpration("Edit", request);

ServletUtility.setBean(bean, request);

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

}

ServletUtility.forward(getView(), request, response);

log.debug("ManagerCtl Method doGett Ended");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("ManagerCtl Method doPost Started");

System.out.println("in do post");

String op = DataUtility.getString(request.getParameter("operation"));

// get model

ManagerModel model = new ManagerModel();

long id = DataUtility.getLong(request.getParameter("id"));

ServletUtility.setOpration("Add", request);

if (OP\_SAVE.equalsIgnoreCase(op)) {

ManagerBean bean = (ManagerBean) populateBean(request);

LoginModel lModel=new LoginModel();

LoginBean lBean=new LoginBean();

try {

if (id > 0) {

LoginBean lbBean= lModel.findByLogin(bean.getLogin());

lbBean.setLogin(bean.getLogin());

lbBean.setPassword(bean.getPassword());

lbBean.setRole(bean.getRole());

lModel.update(lbBean);

model.update(bean);

ServletUtility.setSuccessMessage("Data is successfully Updated",request);

ServletUtility.setOpration("Edit", request);

ServletUtility.setBean(bean, request);

} else {

lBean.setLogin(bean.getLogin());

lBean.setPassword(bean.getPassword());

lBean.setRole(bean.getRole());

lModel.add(lBean);

long pk = model.add(bean);

ServletUtility.setSuccessMessage("Data is successfully saved",request);

}

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

} catch (DuplicateRecordException e) {

ServletUtility.setBean(bean, request);

ServletUtility.setErrorMessage(e.getMessage(), request);

}

ServletUtility.forward(getView(), request, response);

}

else if (OP\_DELETE.equalsIgnoreCase(op)) {

ManagerBean bean = (ManagerBean) populateBean(request);

try {

model.delete(bean);

ServletUtility.redirect(OGDView.MANAGER\_CTL, request,response);

return;

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

} else if (OP\_CANCEL.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.MANAGER\_CTL, request, response);

}else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.MANAGER\_CTL, request, response);

return;

}

log.debug("ManagerCtl Method doPost Ended");

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.MANAGER\_VIEW;

}

}**package** com.online.garments.deal.controller;

**public** **interface** OGDView {

**public** String ***APP\_CONTEXT*** = "/ONLINE-GARMENTS-DEAL";

**public** String ***LAYOUT\_VIEW*** = "/BaseLayout.jsp";

**public** String ***PAGE\_FOLDER*** = "/jsp";

**public** String ***JAVA\_DOC\_VIEW*** = ***APP\_CONTEXT*** + "/doc/index.html";

**public** String ***ERROR\_VIEW*** = ***PAGE\_FOLDER*** + "/Error.jsp";

**public** String ***USER\_VIEW*** = ***PAGE\_FOLDER*** + "/UserView.jsp";

**public** String ***USER\_LIST\_VIEW*** = ***PAGE\_FOLDER*** + "/UserListView.jsp";

**public** String ***USER\_REGISTRATION\_VIEW*** = ***PAGE\_FOLDER*** + "/UserRegistrationView.jsp";

**public** String ***PRODUCT\_VIEW*** = ***PAGE\_FOLDER*** + "/ProductView.jsp";

**public** String ***PRODUCT\_LIST\_VIEW*** = ***PAGE\_FOLDER*** + "/ProductListView.jsp";

**public** String ***PAYMENT\_LIST\_VIEW*** = ***PAGE\_FOLDER*** + "/PaymentListView.jsp";

**public** String ***CUSTOMER\_VIEW*** = ***PAGE\_FOLDER*** + "/CustomerView.jsp";

**public** String ***CUSTOMER\_LIST\_VIEW*** = ***PAGE\_FOLDER*** + "/CustomerListView.jsp";

**public** String ***BOOK\_LIST\_VIEW*** = ***PAGE\_FOLDER*** + "/BookListView.jsp";

**public** String ***LOGIN\_VIEW*** = ***PAGE\_FOLDER*** + "/LoginView.jsp";

**public** String ***WELCOME\_VIEW*** = ***PAGE\_FOLDER*** + "/Welcome.jsp";

**public** String ***CHANGE\_PASSWORD\_VIEW*** = ***PAGE\_FOLDER*** + "/ChangePasswordView.jsp";

**public** String ***MY\_PROFILE\_VIEW*** = ***PAGE\_FOLDER*** + "/MyProfileView.jsp";

**public** String ***FORGET\_PASSWORD\_VIEW*** = ***PAGE\_FOLDER*** + "/ForgetPasswordView.jsp";

**public** String ***MANAGER\_VIEW*** = ***PAGE\_FOLDER*** + "/ManagerView.jsp";

**public** String ***PAYMENT\_VIEW*** = ***PAGE\_FOLDER*** + "/PaymentView.jsp";

**public** String ***SUCCESS\_VIEW*** = ***PAGE\_FOLDER*** + "/SuccessView.jsp";

**public** String ***ERROR\_CTL*** = "/ctl/ErrorCtl";

**public** String ***USER\_CTL*** = ***APP\_CONTEXT*** + "/ctl/UserCtl";

**public** String ***USER\_LIST\_CTL*** = ***APP\_CONTEXT*** + "/ctl/UserListCtl";

**public** String ***MANAGER\_CTL*** = ***APP\_CONTEXT*** + "/ctl/ManagerCtl";

**public** String ***PRODUCT\_CTL*** = ***APP\_CONTEXT*** + "/ctl/ProductCtl";

**public** String ***PRODUCT\_LIST\_CTL*** = ***APP\_CONTEXT*** + "/ProductListCtl";

**public** String ***PAYMENT\_LIST\_CTL*** = ***APP\_CONTEXT*** + "/ctl/PaymentListCtl";

**public** String ***CUSTOMER\_CTL*** = ***APP\_CONTEXT*** + "/CustomerCtl";

**public** String ***CUSTOMER\_LIST\_CTL*** = ***APP\_CONTEXT*** + "/ctl/CustomerListCtl";

**public** String ***BOOK\_LIST\_CTL*** = ***APP\_CONTEXT*** + "/ctl/BookListCtl";

**public** String ***USER\_REGISTRATION\_CTL*** = ***APP\_CONTEXT*** + "/UserRegistrationCtl";

**public** String ***LOGIN\_CTL*** = ***APP\_CONTEXT*** + "/LoginCtl";

**public** String ***WELCOME\_CTL*** = ***APP\_CONTEXT*** + "/WelcomeCtl";

**public** String ***LOGOUT\_CTL*** = ***APP\_CONTEXT*** + "/LoginCtl";

**public** String ***GET\_MARKSHEET\_CTL*** = ***APP\_CONTEXT*** + "/ctl/GetMarksheetCtl";

**public** String ***CHANGE\_PASSWORD\_CTL*** = ***APP\_CONTEXT*** + "/ctl/ChangePasswordCtl";

**public** String ***MY\_PROFILE\_CTL*** = ***APP\_CONTEXT*** + "/ctl/MyProfileCtl";

**public** String ***FORGET\_PASSWORD\_CTL*** = ***APP\_CONTEXT*** + "/ForgetPasswordCtl";

**public** String ***MARKSHEET\_MERIT\_LIST\_CTL*** = ***APP\_CONTEXT*** + "/ctl/MarksheetMeritListCtl";

}

package com.online.garments.deal.controller;

import java.io.IOException;

import java.util.List;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.PaymentBean;

import com.online.garments.deal.bean.ProductBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.model.PaymentModel;

import com.online.garments.deal.model.ProductModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "PaymentListCtl", urlPatterns = { "/ctl/PaymentListCtl" })

public class PaymentListCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

private static Logger log = Logger.getLogger(PaymentListCtl.class);

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("PaymentListCtl populateBean method start");

PaymentBean bean = new PaymentBean();

bean.setProductId(DataUtility.getLong(request.getParameter("Pro")));

log.debug("PaymentListCtl populateBean method end");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("PaymentListCtl doGet method start");

List list = null;

int pageNo = 1;

int pageSize = DataUtility.getInt(PropertyReader.getValue("page.size"));

long idd=DataUtility.getLong(request.getParameter("idd"));

PaymentModel model = new PaymentModel();

PaymentBean bean = (PaymentBean) populateBean(request);

try {

if(idd>0) {

PaymentBean pBean=new PaymentBean();

pBean.setId(idd);

model.delete(pBean);

ServletUtility.setSuccessMessage("Data Deleted Succsessfully", request);

}

list = model.search(bean, pageNo, pageSize);

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("No Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("PaymentListCtl doGet method end");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("PaymentListCtl doPost method start");

List list = null;

int pageNo = DataUtility.getInt(request.getParameter("pageNo"));

int pageSize = DataUtility.getInt(request.getParameter("pageSize"));

pageNo = (pageNo == 0) ? 1 : pageNo;

pageSize = (pageSize == 0) ? DataUtility.getInt(PropertyReader.getValue("page.size")) : pageSize;

PaymentBean bean = (PaymentBean) populateBean(request);

PaymentModel model = new PaymentModel();

String[] ids = request.getParameterValues("ids");

String op = DataUtility.getString(request.getParameter("operation"));

if (OP\_SEARCH.equalsIgnoreCase(op) || OP\_NEXT.equalsIgnoreCase(op) || OP\_PREVIOUS.equalsIgnoreCase(op)) {

if (OP\_SEARCH.equalsIgnoreCase(op)) {

pageNo = 1;

} else if (OP\_NEXT.equalsIgnoreCase(op)) {

pageNo++;

} else if (OP\_PREVIOUS.equalsIgnoreCase(op) && pageNo > 1) {

pageNo--;

}

} else if (OP\_NEW.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

return;

} else if (OP\_DELETE.equalsIgnoreCase(op)) {

pageNo = 1;

if (ids != null && ids.length > 0) {

PaymentBean deletebean = new PaymentBean();

for (String id : ids) {

deletebean.setId(DataUtility.getInt(id));

try {

model.delete(deletebean);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

}

ServletUtility.setSuccessMessage("Data Deleted Successfully", request);

} else {

ServletUtility.setErrorMessage("Select at least one record", request);

}

} else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

return;

}

try {

list = model.search(bean, pageNo, pageSize);

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("NO Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("PaymentListCtl doPost method end");

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.PAYMENT\_LIST\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.File;

import java.io.IOException;

import java.nio.file.Paths;

import javax.servlet.ServletException;

import javax.servlet.annotation.MultipartConfig;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.Part;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.bean.ManagerBean;

import com.online.garments.deal.bean.ProductBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.model.LoginModel;

import com.online.garments.deal.model.ManagerModel;

import com.online.garments.deal.model.ProductModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.DataValidator;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "ProductCtl", urlPatterns = { "/ctl/ProductCtl" })

@MultipartConfig(maxFileSize = 16177215)

public class ProductCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

private static Logger log = Logger.getLogger(ProductCtl.class);

@Override

protected boolean validate(HttpServletRequest request) {

log.debug("ProductCtl validate method start");

boolean pass = true;

if (DataValidator.isNull(request.getParameter("name"))) {

request.setAttribute("name", PropertyReader.getValue("error.require", "Product Name"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("code"))) {

request.setAttribute("code", PropertyReader.getValue("error.require", "Item Code"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("quantity"))) {

request.setAttribute("quantity", PropertyReader.getValue("error.require", "Quantity "));

pass = false;

}

if (DataValidator.isNull(request.getParameter("choice"))) {

request.setAttribute("choice", PropertyReader.getValue("error.require", "Choice"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("price"))) {

request.setAttribute("price", PropertyReader.getValue("error.require", "Price"));

pass = false;

}else if (DataValidator.isNotNull(request.getParameter("price"))

&& !DataValidator.isDouble(request.getParameter("price"))) {

request.setAttribute("price", PropertyReader.getValue("error.double", "Price"));

pass = false;

}

Part part = null;

try {

part = request.getPart("photo");

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (ServletException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

String imgName = Paths.get(part.getSubmittedFileName()).getFileName().toString();

if (DataValidator.isNull(imgName)) {

request.setAttribute("photo", PropertyReader.getValue("error.require", "Image"));

pass = false;

}

log.debug("ProductCtl validate method end");

return pass;

}

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("ProductCtl Method populatebean Started");

ProductBean bean = new ProductBean();

bean.setId(DataUtility.getLong(request.getParameter("id")));

bean.setProductName(DataUtility.getString(request.getParameter("name")));

bean.setProductChoice(DataUtility.getString(request.getParameter("choice")));

bean.setProductQuantity(DataUtility.getString(request.getParameter("quantity")));

bean.setItemCode(DataUtility.getString(request.getParameter("code")));

bean.setPrice(DataUtility.getDouble(request.getParameter("price")));

populateDTO(bean, request);

log.debug("ProductCtl Method populatebean Ended");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

log.debug("ProductCtl Method doGet Started");

System.out.println("in do get");

String op = DataUtility.getString(request.getParameter("operation"));

long id = DataUtility.getLong(request.getParameter("id"));

// get model

ServletUtility.setOpration("Add", request);

ProductModel model = new ProductModel();

if (id > 0 || op != null) {

ProductBean bean;

try {

bean = model.findByPk(id);

ServletUtility.setOpration("Edit", request);

ServletUtility.setBean(bean, request);

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

}

ServletUtility.forward(getView(), request, response);

log.debug("ProductCtl Method doGett Ended");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

log.debug("ProductCtl Method doPost Started");

System.out.println("in do post");

String op = DataUtility.getString(request.getParameter("operation"));

// get model

ProductModel model = new ProductModel();

long id = DataUtility.getLong(request.getParameter("id"));

ServletUtility.setOpration("Add", request);

if (OP\_SAVE.equalsIgnoreCase(op)) {

ProductBean bean = (ProductBean) populateBean(request);

bean.setImage(SubImage(request, response));

try {

if (id > 0) {

model.update(bean);

ServletUtility.setSuccessMessage("Data is successfully Updated", request);

ServletUtility.setOpration("Edit", request);

ServletUtility.setBean(bean, request);

} else {

long pk = model.add(bean);

ServletUtility.setSuccessMessage("Data is successfully saved", request);

}

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

} catch (DuplicateRecordException e) {

ServletUtility.setBean(bean, request);

ServletUtility.setErrorMessage(e.getMessage(), request);

}

ServletUtility.forward(getView(), request, response);

}

else if (OP\_DELETE.equalsIgnoreCase(op)) {

ProductBean bean = (ProductBean) populateBean(request);

try {

model.delete(bean);

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

return;

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

} else if (OP\_CANCEL.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

} else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_CTL, request, response);

return;

}

log.debug("ProductCtl Method doPost Ended");

}

protected String SubImage(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("");

String savePath = DataUtility.getString(PropertyReader.getValue("imagePath"));

File fileSaveDir = new File(savePath);

if (!fileSaveDir.exists()) {

fileSaveDir.mkdir();

}

Part part = request.getPart("photo");

String fileName = extractFileName(part);

part.write(savePath + File.separator + fileName);

String filePath = fileName;

System.out.println("Path----" + savePath + File.separator + fileName);

return fileName;

}

private String extractFileName(Part part) {

String contentDisp = part.getHeader("content-disposition");

String[] items = contentDisp.split(";");

for (String s : items) {

if (s.trim().startsWith("filename")) {

return s.substring(s.indexOf("=") + 2, s.length() - 1);

}

}

return "";

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.PRODUCT\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import java.util.List;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.ProductBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.model.ProductModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "ProductListCtl", urlPatterns = { "/ProductListCtl" })

public class ProductListCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

private static Logger log = Logger.getLogger(ProductListCtl.class);

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("ProductListCtl populateBean method start");

ProductBean bean = new ProductBean();

bean.setProductName(DataUtility.getString(request.getParameter("name")));

bean.setItemCode(DataUtility.getString(request.getParameter("code")));

log.debug("ProductListCtl populateBean method end");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("ProductListCtl doGet method start");

List list = null;

int pageNo = 1;

int pageSize = DataUtility.getInt(PropertyReader.getValue("page.size"));

long idd=DataUtility.getLong(request.getParameter("idd"));

ProductModel model = new ProductModel();

ProductBean bean = (ProductBean) populateBean(request);

try {

if(idd>0) {

ProductBean pBean=new ProductBean();

pBean.setId(idd);

model.delete(pBean);

ServletUtility.setSuccessMessage("Data Deleted Succsessfully", request);

}

list = model.list();

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("No Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("ProductListCtl doGet method end");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("ProductListCtl doPost method start");

List list = null;

int pageNo = DataUtility.getInt(request.getParameter("pageNo"));

int pageSize = DataUtility.getInt(request.getParameter("pageSize"));

pageNo = (pageNo == 0) ? 1 : pageNo;

pageSize = (pageSize == 0) ? DataUtility.getInt(PropertyReader.getValue("page.size")) : pageSize;

ProductBean bean = (ProductBean) populateBean(request);

ProductModel model = new ProductModel();

String[] ids = request.getParameterValues("ids");

String op = DataUtility.getString(request.getParameter("operation"));

if (OP\_SEARCH.equalsIgnoreCase(op) || OP\_NEXT.equalsIgnoreCase(op) || OP\_PREVIOUS.equalsIgnoreCase(op)) {

if (OP\_SEARCH.equalsIgnoreCase(op)) {

pageNo = 1;

} else if (OP\_NEXT.equalsIgnoreCase(op)) {

pageNo++;

} else if (OP\_PREVIOUS.equalsIgnoreCase(op) && pageNo > 1) {

pageNo--;

}

} else if (OP\_NEW.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_CTL, request, response);

return;

} else if (OP\_DELETE.equalsIgnoreCase(op)) {

pageNo = 1;

if (ids != null && ids.length > 0) {

ProductBean deletebean = new ProductBean();

for (String id : ids) {

deletebean.setId(DataUtility.getInt(id));

try {

model.delete(deletebean);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

}

ServletUtility.setSuccessMessage("Data Deleted Successfully", request);

} else {

ServletUtility.setErrorMessage("Select at least one record", request);

}

} else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

return;

}

try {

list = model.list();

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("NO Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("ProductListCtl doPost method end");

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.PRODUCT\_LIST\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.model.LoginModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.DataValidator;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "UserRegistrationCtl", urlPatterns = { "/UserRegistrationCtl" })

public class UserRegistrationCtl extends BaseCtl {

public static final String OP\_SIGN\_UP = "SignUp";

private static Logger log = Logger.getLogger(UserRegistrationCtl.class);

@Override

protected boolean validate(HttpServletRequest request) {

log.debug("UserRegistrationCtl Method validate Started");

boolean pass = true;

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

if (DataValidator.isNull(login)) {

request.setAttribute("login", PropertyReader.getValue("error.require", "Login Id"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("password"))) {

request.setAttribute("password", PropertyReader.getValue("error.require", "Password"));

pass = false;

}

if (!request.getParameter("password").equals(

request.getParameter("confirmPassword"))

&& !"".equals(request.getParameter("confirmPassword"))) {

/\*ServletUtility.setErrorMessage("Confirm Password did not match",

request);\*/

request.setAttribute("confirmPassword", PropertyReader.getValue("error.confirmPassword","Confirm Password"));

pass = false;

}

log.debug("UserRegistrationCtl Method validate Ended");

return pass;

}

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("UserRegistrationCtl Method populatebean Started");

LoginBean bean = new LoginBean();

bean.setId(DataUtility.getLong(request.getParameter("id")));

bean.setRole(3L);

bean.setLogin(DataUtility.getString(request.getParameter("login")));

bean.setPassword(DataUtility.getString(request.getParameter("password")));

bean.setConfirmPassword(DataUtility.getString(request

.getParameter("confirmPassword")));

populateDTO(bean, request);

log.debug("UserRegistrationCtl Method populatebean Ended");

return bean;

}

private static final long serialVersionUID = 1L;

public UserRegistrationCtl() {

super();

// TODO Auto-generated constructor stub

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("UserRegistrationCtl Method doGet Started");

ServletUtility.forward(getView(), request, response);

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

System.out.println("in post method");

log.debug("UserRegistrationCtl Method doPost Started");

String op = DataUtility.getString(request.getParameter("operation"));

// get model

LoginModel model = new LoginModel();

long id = DataUtility.getLong(request.getParameter("id"));

if (OP\_SIGN\_UP.equalsIgnoreCase(op)) {

LoginBean bean = (LoginBean) populateBean(request);

try {

// System.out.println("in try sign up");

long pk = model.add(bean);

//System.out.println("register");

bean.setId(pk);

request.getSession().setAttribute("UserBean", bean);

ServletUtility.setSuccessMessage("User Successfully Registered", request);

ServletUtility.forward(OGDView.USER\_REGISTRATION\_VIEW, request, response);

return;

} catch (DuplicateRecordException e) {

log.error(e);

ServletUtility.setBean(bean, request);

ServletUtility.setErrorMessage(e.getMessage(),

request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

}else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.USER\_REGISTRATION\_CTL, request, response);

return;

}

log.debug("UserRegistrationCtl Method doPost Ended");

}

@Override

protected String getView() {

return OGDView.USER\_REGISTRATION\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "WelcomeCtl", urlPatterns = { "/WelcomeCtl" })

public class WelcomeCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

/\*public WelcomeCtl() {

super();

// TODO Auto-generated constructor stub

}\*/

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

ServletUtility.forward(OGDView.WELCOME\_VIEW, request, response);

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.WELCOME\_VIEW;

}

}**package** com.online.garments.deal.exception;

**public** **class** ApplicationException **extends** Exception

{

**public** ApplicationException(String msg) {

**super**(msg);

}

}

**package** com.online.garments.deal.exception;

**public** **class** DatabaseException **extends** Exception

{

**public** DatabaseException(String msg) {

**super**(msg);

}

}

**package** com.online.garments.deal.exception;

**public** **class** DuplicateRecordException **extends** Exception

{

**public** DuplicateRecordException(String msg) {

**super**(msg);

}

}

**package** com.online.garments.deal.exception;

**public** **class** RecordNotFoundException **extends** Exception

{

**public** RecordNotFoundException(String msg) {

**super**(msg);

}

}

package com.online.garments.deal.model;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.util.ArrayList;

import java.util.List;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.CustomerBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DatabaseException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.util.JDBCDataSource;

public class CustomerModel {

private static Logger log = Logger.getLogger(CustomerModel.class);

public Integer nextPK() throws DatabaseException {

log.debug("Model nextPK Started");

Connection conn = null;

int pk = 0;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement("SELECT MAX(ID) FROM Customer");

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

pk = rs.getInt(1);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new DatabaseException("Exception : Exception in getting PK");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model nextPK End");

return pk + 1;

}

public CustomerBean findByName(String name) throws ApplicationException {

log.debug("Model findBy Name Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Customer WHERE name=?");

CustomerBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, name);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new CustomerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setAge(rs.getString(3));

bean.setAddress(rs.getString(4));

bean.setContectNo(rs.getString(5));

bean.setProductChoice(rs.getString(6));

bean.setItemCode(rs.getString(7));

bean.setMultipleItems(rs.getString(8));

bean.setCreatedBy(rs.getString(9));

bean.setModifiedBy(rs.getString(10));

bean.setCreatedDatetime(rs.getTimestamp(11));

bean.setModifiedDatetime(rs.getTimestamp(12));

bean.setLogin(rs.getString(13));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Customer by Name");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy Name End");

return bean;

}

public CustomerBean findByContactNo(String contact) throws ApplicationException {

log.debug("Model findBy contact Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Customer WHERE contactNo=?");

CustomerBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, contact);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new CustomerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setAge(rs.getString(3));

bean.setAddress(rs.getString(4));

bean.setContectNo(rs.getString(5));

bean.setProductChoice(rs.getString(6));

bean.setItemCode(rs.getString(7));

bean.setMultipleItems(rs.getString(8));

bean.setCreatedBy(rs.getString(9));

bean.setModifiedBy(rs.getString(10));

bean.setCreatedDatetime(rs.getTimestamp(11));

bean.setModifiedDatetime(rs.getTimestamp(12));

bean.setLogin(rs.getString(13));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Customer by contact");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy contact End");

return bean;

}

public CustomerBean findByPk(long id) throws ApplicationException {

log.debug("Model findBy PK Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Customer WHERE ID=?");

CustomerBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setLong(1, id);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new CustomerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setAge(rs.getString(3));

bean.setAddress(rs.getString(4));

bean.setContectNo(rs.getString(5));

bean.setProductChoice(rs.getString(6));

bean.setItemCode(rs.getString(7));

bean.setMultipleItems(rs.getString(8));

bean.setCreatedBy(rs.getString(9));

bean.setModifiedBy(rs.getString(10));

bean.setCreatedDatetime(rs.getTimestamp(11));

bean.setModifiedDatetime(rs.getTimestamp(12));

bean.setLogin(rs.getString(13));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Customer by Pk");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy Pk End");

return bean;

}

public long add(CustomerBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model add Started");

Connection conn = null;

int pk = 0;

CustomerBean duplicataLogin = findByContactNo(bean.getContectNo());

if (duplicataLogin != null) {

throw new DuplicateRecordException("Contact No Id already exists");

}

try {

conn = JDBCDataSource.getConnection();

pk = nextPK();

// Get auto-generated next primary key

System.out.println(pk + " in ModelJDBC");

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("INSERT INTO Customer VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?)");

pstmt.setInt(1, pk);

pstmt.setString(2, bean.getName());

pstmt.setString(3, bean.getAge());

pstmt.setString(4, bean.getAddress());

pstmt.setString(5, bean.getContectNo());

pstmt.setString(6, bean.getProductChoice());

pstmt.setString(7, bean.getItemCode());

pstmt.setString(8, bean.getMultipleItems());

pstmt.setString(9, bean.getCreatedBy());

pstmt.setString(10, bean.getModifiedBy());

pstmt.setTimestamp(11, bean.getCreatedDatetime());

pstmt.setTimestamp(12, bean.getModifiedDatetime());

pstmt.setString(13,bean.getLogin());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

e.printStackTrace();

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : add rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in add Customer");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model add End");

return pk;

}

public void delete(CustomerBean bean) throws ApplicationException {

log.debug("Model delete Started");

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("DELETE FROM Customer WHERE ID=?");

pstmt.setLong(1, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in delete Customer");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model delete Started");

}

public void update(CustomerBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model update Started");

Connection conn = null;

CustomerBean duplicataCustomer = findByContactNo(bean.getContectNo());

// Check if updated Role already exist

if (duplicataCustomer != null && duplicataCustomer.getId() != bean.getId()) {

throw new DuplicateRecordException("Contact No already exists");

}

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement(

"UPDATE Customer SET Name=?,age=?,Address=?,ContectNo=?,ProductChoice=?,ItemCode=?,MultipleItems=?,CREATEDBY=?,MODIFIEDBY=?,CREATEDDATETIME=?,MODIFIEDDATETIME=?,Login=? WHERE ID=?");

pstmt.setString(1, bean.getName());

pstmt.setString(2, bean.getAge());

pstmt.setString(3, bean.getAddress());

pstmt.setString(4, bean.getContectNo());

pstmt.setString(5, bean.getProductChoice());

pstmt.setString(6, bean.getItemCode());

pstmt.setString(7, bean.getMultipleItems());

pstmt.setString(8, bean.getCreatedBy());

pstmt.setString(9, bean.getModifiedBy());

pstmt.setTimestamp(10, bean.getCreatedDatetime());

pstmt.setTimestamp(11, bean.getModifiedDatetime());

pstmt.setString(12,bean.getLogin());

pstmt.setLong(13, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception in updating Customer ");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model update End");

}

public List search(CustomerBean bean) throws ApplicationException {

return search(bean, 0, 0);

}

public List search(CustomerBean bean, int pageNo, int pageSize)

throws ApplicationException {

log.debug("Model search Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Customer WHERE 1=1");

if (bean != null) {

if (bean.getId() > 0) {

sql.append(" AND id = " + bean.getId());

}

if (bean.getContectNo() != null && bean.getContectNo().length() > 0) {

sql.append(" AND ContectNo LIKE '" + bean.getContectNo() + "%'");

}

if (bean.getItemCode() != null && bean.getItemCode().length() > 0) {

sql.append(" AND ItemCode LIKE '" + bean.getItemCode() + "%'");

}

if (bean.getLogin() != null && bean.getLogin().length() > 0) {

sql.append(" AND Login LIKE '" + bean.getLogin() + "%'");

}

}

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" Limit " + pageNo + ", " + pageSize);

// sql.append(" limit " + pageNo + "," + pageSize);

}

ArrayList list = new ArrayList();

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new CustomerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setAge(rs.getString(3));

bean.setAddress(rs.getString(4));

bean.setContectNo(rs.getString(5));

bean.setProductChoice(rs.getString(6));

bean.setItemCode(rs.getString(7));

bean.setMultipleItems(rs.getString(8));

bean.setCreatedBy(rs.getString(9));

bean.setModifiedBy(rs.getString(10));

bean.setCreatedDatetime(rs.getTimestamp(11));

bean.setModifiedDatetime(rs.getTimestamp(12));

bean.setLogin(rs.getString(13));

list.add(bean);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in search Customer");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model search End");

return list;

}

public List list() throws ApplicationException {

return list(0, 0);

}

public List list(int pageNo, int pageSize) throws ApplicationException {

log.debug("Model list Started");

ArrayList list = new ArrayList();

StringBuffer sql = new StringBuffer("select \* from Customer");

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" limit " + pageNo + "," + pageSize);

}

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

CustomerBean bean = new CustomerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setAge(rs.getString(3));

bean.setAddress(rs.getString(4));

bean.setContectNo(rs.getString(5));

bean.setProductChoice(rs.getString(6));

bean.setItemCode(rs.getString(7));

bean.setMultipleItems(rs.getString(8));

bean.setCreatedBy(rs.getString(9));

bean.setModifiedBy(rs.getString(10));

bean.setCreatedDatetime(rs.getTimestamp(11));

bean.setModifiedDatetime(rs.getTimestamp(12));

bean.setLogin(rs.getString(13));

list.add(bean);

}

rs.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in getting list of Customer");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model list End");

return list;

}

}package com.online.garments.deal.model;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.util.ArrayList;

import java.util.List;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DatabaseException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.JDBCDataSource;

public class LoginModel {

private static Logger log = Logger.getLogger(LoginModel.class);

public Integer nextPK() throws DatabaseException {

log.debug("Model nextPK Started");

Connection conn = null;

int pk = 0;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement("SELECT MAX(ID) FROM Login");

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

pk = rs.getInt(1);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new DatabaseException("Exception : Exception in getting PK");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model nextPK End");

return pk + 1;

}

public LoginBean findByLogin(String login) throws ApplicationException {

log.debug("Model findBy LoginId Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Login WHERE Login=?");

LoginBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, login);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new LoginBean();

bean.setId(rs.getLong(1));

bean.setLogin(rs.getString(2));

bean.setPassword(rs.getString(3));

bean.setCreatedBy(rs.getString(4));

bean.setModifiedBy(rs.getString(5));

bean.setCreatedDatetime(rs.getTimestamp(6));

bean.setModifiedDatetime(rs.getTimestamp(7));

bean.setRole(rs.getLong(8));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting User by LoginId");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy EmailId End");

return bean;

}

public LoginBean findByPK(long pk) throws ApplicationException {

log.debug("Model findByPK Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Login WHERE ID=?");

LoginBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setLong(1, pk);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new LoginBean();

bean.setId(rs.getLong(1));

bean.setLogin(rs.getString(2));

bean.setPassword(rs.getString(3));

bean.setCreatedBy(rs.getString(4));

bean.setModifiedBy(rs.getString(5));

bean.setCreatedDatetime(rs.getTimestamp(6));

bean.setModifiedDatetime(rs.getTimestamp(7));

bean.setRole(rs.getLong(8));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Login by pk");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findByPK End");

return bean;

}

public long add(LoginBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model add Started");

Connection conn = null;

int pk = 0;

LoginBean duplicataLogin = findByLogin(bean.getLogin());

if (duplicataLogin != null) {

throw new DuplicateRecordException("Login Id already exists");

}

try {

conn = JDBCDataSource.getConnection();

pk = nextPK();

// Get auto-generated next primary key

System.out.println(pk + " in ModelJDBC");

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("INSERT INTO Login VALUES(?,?,?,?,?,?,?,?)");

pstmt.setInt(1, pk);

pstmt.setString(2, bean.getLogin());

pstmt.setString(3, bean.getPassword());

pstmt.setString(4, bean.getCreatedBy());

pstmt.setString(5, bean.getModifiedBy());

pstmt.setTimestamp(6, DataUtility.getCurrentTimestamp());

pstmt.setTimestamp(7, DataUtility.getCurrentTimestamp());

pstmt.setLong(8,bean.getRole());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

e.printStackTrace();

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : add rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in add Login");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model add End");

return pk;

}

public void delete(LoginBean bean) throws ApplicationException {

log.debug("Model delete Started");

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("DELETE FROM Login WHERE ID=?");

pstmt.setLong(1, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in delete Role");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model delete Started");

}

public void update(LoginBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model update Started");

Connection conn = null;

LoginBean duplicataRole = findByLogin(bean.getLogin());

// Check if updated Role already exist

if (duplicataRole != null && duplicataRole.getId() != bean.getId()) {

throw new DuplicateRecordException("Login Id already exists");

}

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement(

"UPDATE login SET Login=?,Password=?,CREATEDBY=?,MODIFIEDBY=?,CREATEDDATETIME=?,MODIFIEDDATETIME=?,Role=? WHERE ID=?");

pstmt.setString(1, bean.getLogin());

pstmt.setString(2, bean.getPassword());

pstmt.setString(3, bean.getCreatedBy());

pstmt.setString(4, bean.getModifiedBy());

pstmt.setTimestamp(5, bean.getCreatedDatetime());

pstmt.setTimestamp(6, bean.getModifiedDatetime());

pstmt.setLong(7,bean.getRole());

pstmt.setLong(8, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception in updating Login ");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model update End");

}

public List search(LoginBean bean) throws ApplicationException {

return search(bean, 0, 0);

}

public List search(LoginBean bean, int pageNo, int pageSize)

throws ApplicationException {

log.debug("Model search Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Login WHERE 1=1");

if (bean != null) {

if (bean.getId() > 0) {

sql.append(" AND id = " + bean.getId());

}

if (bean.getLogin() != null && bean.getLogin().length() > 0) {

sql.append(" AND LOGIN LIKE '" + bean.getLogin() + "%'");

}

}

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" Limit " + pageNo + ", " + pageSize);

// sql.append(" limit " + pageNo + "," + pageSize);

}

ArrayList list = new ArrayList();

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new LoginBean();

bean.setId(rs.getLong(1));

bean.setLogin(rs.getString(2));

bean.setPassword(rs.getString(3));

bean.setCreatedBy(rs.getString(4));

bean.setModifiedBy(rs.getString(5));

bean.setCreatedDatetime(rs.getTimestamp(6));

bean.setModifiedDatetime(rs.getTimestamp(7));

bean.setRole(rs.getLong(8));

list.add(bean);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in search Login");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model search End");

return list;

}

public List list() throws ApplicationException {

return list(0, 0);

}

public List list(int pageNo, int pageSize) throws ApplicationException {

log.debug("Model list Started");

ArrayList list = new ArrayList();

StringBuffer sql = new StringBuffer("select \* from Login");

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" limit " + pageNo + "," + pageSize);

}

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

LoginBean bean = new LoginBean();

bean.setId(rs.getLong(1));

bean.setLogin(rs.getString(2));

bean.setPassword(rs.getString(3));

bean.setCreatedBy(rs.getString(4));

bean.setModifiedBy(rs.getString(5));

bean.setCreatedDatetime(rs.getTimestamp(6));

bean.setModifiedDatetime(rs.getTimestamp(7));

bean.setRole(rs.getLong(8));

list.add(bean);

}

rs.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in getting list of Login");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model list End");

return list;

}

public LoginBean authenticate(String login, String password) throws ApplicationException {

log.debug("Model authenticate Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Login WHERE LOGIN = ? AND PASSWORD = ?");

LoginBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, login);

pstmt.setString(2, password);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new LoginBean();

bean.setId(rs.getLong(1));

bean.setLogin(rs.getString(2));

bean.setPassword(rs.getString(3));

bean.setCreatedBy(rs.getString(4));

bean.setModifiedBy(rs.getString(5));

bean.setCreatedDatetime(rs.getTimestamp(6));

bean.setModifiedDatetime(rs.getTimestamp(7));

bean.setRole(rs.getLong(8));

System.out.println("Usermodel here");

}

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in get Authenticate Login");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model authenticate End");

return bean;

}

}package com.online.garments.deal.model;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.util.ArrayList;

import java.util.List;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.CustomerBean;

import com.online.garments.deal.bean.ManagerBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DatabaseException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.util.JDBCDataSource;

public class ManagerModel {

private static Logger log = Logger.getLogger(ManagerModel.class);

public Integer nextPK() throws DatabaseException {

log.debug("Model nextPK Started");

Connection conn = null;

int pk = 0;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement("SELECT MAX(ID) FROM Manager");

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

pk = rs.getInt(1);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new DatabaseException("Exception : Exception in getting PK");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model nextPK End");

return pk + 1;

}

public ManagerBean findByLogin(String login) throws ApplicationException {

log.debug("Model findBy Name Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM manager WHERE Login=?");

ManagerBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, login);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ManagerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setEmailId(rs.getString(3));

bean.setContactNo(rs.getString(4));

bean.setAddress(rs.getString(5));

bean.setLogin(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Manager by Login");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy Name End");

return bean;

}

public ManagerBean findByPk(long id) throws ApplicationException {

log.debug("Model findBy PK Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Manager WHERE ID=?");

ManagerBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setLong(1, id);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ManagerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setEmailId(rs.getString(3));

bean.setContactNo(rs.getString(4));

bean.setAddress(rs.getString(5));

bean.setLogin(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Manager by Pk");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy Pk End");

return bean;

}

public long add(ManagerBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model add Started");

Connection conn = null;

int pk = 0;

ManagerBean duplicataLogin = findByLogin(bean.getLogin());

if (duplicataLogin != null) {

throw new DuplicateRecordException("Login Id already exists");

}

try {

conn = JDBCDataSource.getConnection();

pk = nextPK();

// Get auto-generated next primary key

System.out.println(pk + " in ModelJDBC");

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("INSERT INTO Manager VALUES(?,?,?,?,?,?,?,?,?,?)");

pstmt.setInt(1, pk);

pstmt.setString(2, bean.getName());

pstmt.setString(3, bean.getEmailId());

pstmt.setString(4, bean.getContactNo());

pstmt.setString(5, bean.getAddress());

pstmt.setString(6, bean.getLogin());

pstmt.setString(7, bean.getCreatedBy());

pstmt.setString(8, bean.getModifiedBy());

pstmt.setTimestamp(9, bean.getCreatedDatetime());

pstmt.setTimestamp(10, bean.getModifiedDatetime());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

e.printStackTrace();

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : add rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in add Manager");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model add End");

return pk;

}

public void delete(ManagerBean bean) throws ApplicationException {

log.debug("Model delete Started");

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("DELETE FROM Manager WHERE ID=?");

pstmt.setLong(1, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in delete Manager");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model delete Started");

}

public void update(ManagerBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model update Started");

Connection conn = null;

ManagerBean duplicataCustomer = findByLogin(bean.getLogin());

// Check if updated Role already exist

if (duplicataCustomer != null && duplicataCustomer.getId() != bean.getId()) {

throw new DuplicateRecordException("Login Id already exists");

}

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement(

"UPDATE Manager SET Name=?,EmailId=?,ContactNo=?,Address=?,Login=?,CREATEDBY=?,MODIFIEDBY=?,CREATEDDATETIME=?,MODIFIEDDATETIME=? WHERE ID=?");

pstmt.setString(1, bean.getName());

pstmt.setString(2, bean.getEmailId());

pstmt.setString(3, bean.getContactNo());

pstmt.setString(4, bean.getAddress());

pstmt.setString(5, bean.getLogin());

pstmt.setString(6, bean.getCreatedBy());

pstmt.setString(7, bean.getModifiedBy());

pstmt.setTimestamp(8, bean.getCreatedDatetime());

pstmt.setTimestamp(9, bean.getModifiedDatetime());

pstmt.setLong(10, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception in updating Manager ");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model update End");

}

public List search(ManagerBean bean) throws ApplicationException {

return search(bean, 0, 0);

}

public List search(ManagerBean bean, int pageNo, int pageSize)

throws ApplicationException {

log.debug("Model search Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Manager WHERE 1=1");

if (bean != null) {

if (bean.getId() > 0) {

sql.append(" AND id = " + bean.getId());

}

if (bean.getName() != null && bean.getName().length() > 0) {

sql.append(" AND Name LIKE '" + bean.getName() + "%'");

}

if (bean.getLogin() != null && bean.getLogin().length() > 0) {

sql.append(" AND Login LIKE '" + bean.getLogin() + "%'");

}

}

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" Limit " + pageNo + ", " + pageSize);

// sql.append(" limit " + pageNo + "," + pageSize);

}

ArrayList list = new ArrayList();

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ManagerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setEmailId(rs.getString(3));

bean.setContactNo(rs.getString(4));

bean.setAddress(rs.getString(5));

bean.setLogin(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

list.add(bean);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in search Manager");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model search End");

return list;

}

public List list() throws ApplicationException {

return list(0, 0);

}

public List list(int pageNo, int pageSize) throws ApplicationException {

log.debug("Model list Started");

ArrayList list = new ArrayList();

StringBuffer sql = new StringBuffer("select \* from Manager");

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" limit " + pageNo + "," + pageSize);

}

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

ManagerBean bean = new ManagerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setEmailId(rs.getString(3));

bean.setContactNo(rs.getString(4));

bean.setAddress(rs.getString(5));

bean.setLogin(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

list.add(bean);

}

rs.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in getting list of Manager");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model list End");

return list;

}

}package com.online.garments.deal.model;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.util.ArrayList;

import java.util.List;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.PaymentBean;

import com.online.garments.deal.bean.ProductBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DatabaseException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.JDBCDataSource;

public class PaymentModel {

private static Logger log = Logger.getLogger(PaymentModel.class);

public Integer nextPK() throws DatabaseException {

log.debug("Model nextPK Started");

Connection conn = null;

int pk = 0;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement("SELECT MAX(ID) FROM payment");

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

pk = rs.getInt(1);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new DatabaseException("Exception : Exception in getting PK");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model nextPK End");

return pk + 1;

}

public Integer nextPaymentId() throws DatabaseException {

log.debug("Model nextPK Started");

Connection conn = null;

int pk = 0;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement("SELECT MAX(paymentId) FROM payment");

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

pk = rs.getInt(1);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new DatabaseException("Exception : Exception in getting PK");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model nextPK End");

return pk + 1;

}

public PaymentBean findByPk(long Id) throws ApplicationException {

log.debug("Model findBy Id Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM payment WHERE Id=? ");

PaymentBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setLong(1, Id);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new PaymentBean();

bean.setId(rs.getLong(1));

bean.setPaymentId(rs.getLong(2));

bean.setCustomerId(rs.getLong(3));

bean.setProductId(rs.getLong(4));

bean.setPaymentDate(rs.getDate(5));

bean.setAmount(rs.getDouble(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setLogin(rs.getString(11));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Payment by Id");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy ID End");

return bean;

}

public PaymentBean findByProductAndCustomer(long productId,long customerId) throws ApplicationException {

log.debug("Model findBy Id Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM payment WHERE productId=? and customerId=? ");

PaymentBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setLong(1, productId);

pstmt.setLong(2, customerId);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new PaymentBean();

bean.setId(rs.getLong(1));

bean.setPaymentId(rs.getLong(2));

bean.setCustomerId(rs.getLong(3));

bean.setProductId(rs.getLong(4));

bean.setPaymentDate(rs.getDate(5));

bean.setAmount(rs.getDouble(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setLogin(rs.getString(11));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Customer And product by Id");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy ID End");

return bean;

}

public long add(PaymentBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model add Started");

Connection conn = null;

int pk = 0;

int paymentId = 0;

PaymentBean duplicataLogin = findByProductAndCustomer(bean.getPaymentId(), bean.getCustomerId());

if (duplicataLogin != null) {

throw new DuplicateRecordException("Payment Done Already ");

}

try {

conn = JDBCDataSource.getConnection();

pk = nextPK();

paymentId=nextPaymentId();

// Get auto-generated next primary key

System.out.println(pk + " in ModelJDBC");

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("INSERT INTO payment VALUES(?,?,?,?,?,?,?,?,?,?,?)");

pstmt.setInt(1, pk);

pstmt.setLong(2, paymentId);

pstmt.setLong(3, bean.getCustomerId());

pstmt.setLong(4, bean.getProductId());

pstmt.setDate(5, new java.sql.Date(new java.util.Date().getTime()));

pstmt.setDouble(6, bean.getAmount());

pstmt.setString(7, bean.getCreatedBy());

pstmt.setString(8, bean.getModifiedBy());

pstmt.setTimestamp(9, DataUtility.getCurrentTimestamp());

pstmt.setTimestamp(10, DataUtility.getCurrentTimestamp());

pstmt.setString(11,bean.getLogin());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

e.printStackTrace();

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : add rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in add Payment");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model add End");

return pk;

}

public void delete(PaymentBean bean) throws ApplicationException {

log.debug("Model delete Started");

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("DELETE FROM Payment WHERE ID=?");

pstmt.setLong(1, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in delete Payment");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model delete Started");

}

public void update(PaymentBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model update Started");

Connection conn = null;

PaymentBean duplicataCustomer = findByProductAndCustomer(bean.getPaymentId(), bean.getCustomerId());

// Check if updated Role already exist

if (duplicataCustomer != null && duplicataCustomer.getId() != bean.getId()) {

throw new DuplicateRecordException("Payment is already Done");

}

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement(

"UPDATE payment SET PaymentId=?,CustomerId=?,ProductId=?,paymentDate=?,Amount=?,CREATEDBY=?,MODIFIEDBY=?,CREATEDDATETIME=?,MODIFIEDDATETIME=?,login=? WHERE ID=?");

pstmt.setLong(1, bean.getPaymentId());

pstmt.setLong(2, bean.getCustomerId());

pstmt.setLong(3, bean.getProductId());

pstmt.setDate(4, new java.sql.Date( bean.getPaymentDate().getTime()));

pstmt.setDouble(5, bean.getAmount());

pstmt.setString(6, bean.getCreatedBy());

pstmt.setString(7, bean.getModifiedBy());

pstmt.setTimestamp(8, bean.getCreatedDatetime());

pstmt.setTimestamp(9, bean.getModifiedDatetime());

pstmt.setString(10, bean.getLogin());

pstmt.setLong(11, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception in updating Payment");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model update End");

}

public List search(PaymentBean bean) throws ApplicationException {

return search(bean, 0, 0);

}

public List search(PaymentBean bean, int pageNo, int pageSize)

throws ApplicationException {

log.debug("Model search Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Payment WHERE 1=1");

if (bean != null) {

if (bean.getId() > 0) {

sql.append(" AND id = " + bean.getId());

}

if (bean.getPaymentId() > 0) {

sql.append(" AND PaymentId = " + bean.getPaymentId());

}

if (bean.getCustomerId() > 0) {

sql.append(" AND CustomerId = " + bean.getCustomerId());

}

if (bean.getProductId() > 0) {

sql.append(" AND ProductId = " + bean.getProductId());

}

if (bean.getLogin() != null && bean.getLogin().length() > 0) {

sql.append(" AND Login LIKE '" + bean.getLogin() + "%'");

}

}

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" Limit " + pageNo + ", " + pageSize);

// sql.append(" limit " + pageNo + "," + pageSize);

}

ArrayList list = new ArrayList();

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new PaymentBean();

bean.setId(rs.getLong(1));

bean.setPaymentId(rs.getLong(2));

bean.setCustomerId(rs.getLong(3));

bean.setProductId(rs.getLong(4));

bean.setPaymentDate(rs.getDate(5));

bean.setAmount(rs.getDouble(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setLogin(rs.getString(11));

list.add(bean);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in search Payment");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model search End");

return list;

}

public List list() throws ApplicationException {

return list(0, 0);

}

public List list(int pageNo, int pageSize) throws ApplicationException {

log.debug("Model list Started");

ArrayList list = new ArrayList();

StringBuffer sql = new StringBuffer("select \* from Payment");

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" limit " + pageNo + "," + pageSize);

}

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

PaymentBean bean = new PaymentBean();

bean.setId(rs.getLong(1));

bean.setPaymentId(rs.getLong(2));

bean.setCustomerId(rs.getLong(3));

bean.setProductId(rs.getLong(4));

bean.setPaymentDate(rs.getDate(5));

bean.setAmount(rs.getDouble(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setLogin(rs.getString(11));

list.add(bean);

}

rs.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in getting list of Payment");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model list End");

return list;

}

}package com.online.garments.deal.model;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.util.ArrayList;

import java.util.List;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.CustomerBean;

import com.online.garments.deal.bean.ProductBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DatabaseException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.util.JDBCDataSource;

public class ProductModel {

private static Logger log = Logger.getLogger(ProductModel.class);

public Integer nextPK() throws DatabaseException {

log.debug("Model nextPK Started");

Connection conn = null;

int pk = 0;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement("SELECT MAX(ID) FROM Product");

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

pk = rs.getInt(1);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new DatabaseException("Exception : Exception in getting PK");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model nextPK End");

return pk + 1;

}

public ProductBean findByName(String name) throws ApplicationException {

log.debug("Model findByName Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Product WHERE name=?");

ProductBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, name);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ProductBean();

bean.setId(rs.getLong(1));

bean.setProductName(rs.getString(2));

bean.setProductQuantity(rs.getString(3));

bean.setProductChoice(rs.getString(4));

bean.setItemCode(rs.getString(5));

bean.setImage(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setPrice(rs.getDouble(11));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Product by Name");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findByName End");

return bean;

}

public ProductBean findByItemCode(String code) throws ApplicationException {

log.debug("Model findByName Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Product WHERE ItemCode=?");

ProductBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, code);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ProductBean();

bean.setId(rs.getLong(1));

bean.setProductName(rs.getString(2));

bean.setProductQuantity(rs.getString(3));

bean.setProductChoice(rs.getString(4));

bean.setItemCode(rs.getString(5));

bean.setImage(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setPrice(rs.getDouble(11));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Product by Code");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findByName End");

return bean;

}

public ProductBean findByNameAndCode(String name,String code) throws ApplicationException {

log.debug("Model findByNameCode Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Product WHERE Productname=? and itemCode=?");

ProductBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, name);

pstmt.setString(2, code);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ProductBean();

bean.setId(rs.getLong(1));

bean.setProductName(rs.getString(2));

bean.setProductQuantity(rs.getString(3));

bean.setProductChoice(rs.getString(4));

bean.setItemCode(rs.getString(5));

bean.setImage(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setPrice(rs.getDouble(11));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Product by NameAndCode");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy Name End");

return bean;

}

public ProductBean findByPk(long Id) throws ApplicationException {

log.debug("Model findBy Id Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Product WHERE Id=? ");

ProductBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setLong(1, Id);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ProductBean();

bean.setId(rs.getLong(1));

bean.setProductName(rs.getString(2));

bean.setProductQuantity(rs.getString(3));

bean.setProductChoice(rs.getString(4));

bean.setItemCode(rs.getString(5));

bean.setImage(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setPrice(rs.getDouble(11));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Product by Id");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy ID End");

return bean;

}

public long add(ProductBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model add Started");

Connection conn = null;

int pk = 0;

ProductBean duplicataLogin = findByNameAndCode(bean.getProductName(), bean.getItemCode());

if (duplicataLogin != null) {

throw new DuplicateRecordException(" Product is already exists");

}

try {

conn = JDBCDataSource.getConnection();

pk = nextPK();

// Get auto-generated next primary key

System.out.println(pk + " in ModelJDBC");

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("INSERT INTO product VALUES(?,?,?,?,?,?,?,?,?,?,?)");

pstmt.setInt(1, pk);

pstmt.setString(2, bean.getProductName());

pstmt.setString(3, bean.getProductQuantity());

pstmt.setString(4, bean.getProductChoice());

pstmt.setString(5, bean.getItemCode());

pstmt.setString(6, bean.getImage());

pstmt.setString(7, bean.getCreatedBy());

pstmt.setString(8, bean.getModifiedBy());

pstmt.setTimestamp(9, bean.getCreatedDatetime());

pstmt.setTimestamp(10, bean.getModifiedDatetime());

pstmt.setDouble(11, bean.getPrice());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

e.printStackTrace();

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : add rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in add Product");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model add End");

return pk;

}

public void delete(ProductBean bean) throws ApplicationException {

log.debug("Model delete Started");

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("DELETE FROM Product WHERE ID=?");

pstmt.setLong(1, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in delete Customer");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model delete Started");

}

public void update(ProductBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model update Started");

Connection conn = null;

ProductBean duplicataCustomer = findByNameAndCode(bean.getProductName(), bean.getItemCode());

// Check if updated Role already exist

if (duplicataCustomer != null && duplicataCustomer.getId() != bean.getId()) {

throw new DuplicateRecordException("Product is already exists");

}

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement(

"UPDATE Product SET ProductName=?,ProductQuantity=?,ProductChoice=?,ItemCode=?,Image=?,CREATEDBY=?,MODIFIEDBY=?,CREATEDDATETIME=?,MODIFIEDDATETIME=?,price=? WHERE ID=?");

pstmt.setString(1, bean.getProductName());

pstmt.setString(2, bean.getProductQuantity());

pstmt.setString(3, bean.getProductChoice());

pstmt.setString(4, bean.getItemCode());

pstmt.setString(5, bean.getImage());

pstmt.setString(6, bean.getCreatedBy());

pstmt.setString(7, bean.getModifiedBy());

pstmt.setTimestamp(8, bean.getCreatedDatetime());

pstmt.setTimestamp(9, bean.getModifiedDatetime());

pstmt.setDouble(10, bean.getPrice());

pstmt.setLong(11, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception in updating Product ");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model update End");

}

public List search(ProductBean bean) throws ApplicationException {

return search(bean, 0, 0);

}

public List search(ProductBean bean, int pageNo, int pageSize)

throws ApplicationException {

log.debug("Model search Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Product WHERE 1=1");

if (bean != null) {

if (bean.getId() > 0) {

sql.append(" AND id = " + bean.getId());

}

if (bean.getProductName() != null && bean.getProductName().length() > 0) {

sql.append(" AND ProductName LIKE '" + bean.getProductName() + "%'");

}

if (bean.getItemCode() != null && bean.getItemCode().length() > 0) {

sql.append(" AND ItemCode LIKE '" + bean.getItemCode() + "%'");

}

}

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" Limit " + pageNo + ", " + pageSize);

// sql.append(" limit " + pageNo + "," + pageSize);

}

ArrayList list = new ArrayList();

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ProductBean();

bean.setId(rs.getLong(1));

bean.setProductName(rs.getString(2));

bean.setProductQuantity(rs.getString(3));

bean.setProductChoice(rs.getString(4));

bean.setItemCode(rs.getString(5));

bean.setImage(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setPrice(rs.getDouble(11));

list.add(bean);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in search Product");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model search End");

return list;

}

public List list() throws ApplicationException {

return list(0, 0);

}

public List list(int pageNo, int pageSize) throws ApplicationException {

log.debug("Model list Started");

ArrayList list = new ArrayList();

StringBuffer sql = new StringBuffer("select \* from Product");

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" limit " + pageNo + "," + pageSize);

}

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

ProductBean bean = new ProductBean();

bean.setId(rs.getLong(1));

bean.setProductName(rs.getString(2));

bean.setProductQuantity(rs.getString(3));

bean.setProductChoice(rs.getString(4));

bean.setItemCode(rs.getString(5));

bean.setImage(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setPrice(rs.getDouble(11));

list.add(bean);

}

rs.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in getting list of Product");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model list End");

return list;

}

}package com.online.garments.deal.util;

import java.sql.Timestamp;

import java.text.SimpleDateFormat;

import java.util.Date;

public class DataUtility

{

public static final String APP\_DATE\_FORMAT = "MM/dd/yyyy";

// public static final String APP\_TIME\_FORMAT = "MM/dd/yyyy HH:mm:ss";

private static final SimpleDateFormat formatter = new SimpleDateFormat(APP\_DATE\_FORMAT);

//private static final SimpleDateFormat timeFormatter = new SimpleDateFormat(APP\_TIME\_FORMAT);

public static String getString(String val) {

if (DataValidator.isNotNull(val)) {

return val.trim();

} else {

return val;

}

}

public static String getStringData(Object val) {

if (val != null) {

return val.toString();

} else {

return "";

}

}

public static int getInt(String val) {

if (DataValidator.isInteger(val)) {

return Integer.parseInt(val);

} else {

return 0;

}

}

public static long getLong(String val) {

if (DataValidator.isLong(val)) {

return Long.parseLong(val);

} else {

return 0;

}

}

public static double getDouble(String val) {

if (DataValidator.isDouble(val)) {

return Double.parseDouble(val);

} else {

return 0.0;

}

}

public static Date getDate(String val) {

Date date = null;

try {

date = formatter.parse(val);

} catch (Exception e) {

}

return date;

}

public static Date getDate1(String val) {

Date date = null;

try {

date = formatter.parse(val);

}catch(Exception e){}

return date;

}

public static String getDateString(Date date) {

try {

if(date!=null) {

return formatter.format(date);

}

else{

return "";

}

} catch (Exception e) {

return "";

}

}

public static Date getDate(Date date, int day) {

return null;

}

public static Timestamp getTimestamp(long l) {

Timestamp timeStamp = null;

try {

timeStamp = new Timestamp(l);

} catch (Exception e) {

return null;

}

return timeStamp;

}

public static Timestamp getTimestamp(String cdt) {

Timestamp timeStamp = null;

try {

// timeStamp = new Timestamp((timeFormatter.parse(cdt)).getTime());

} catch (Exception e) {

return null;

}

return timeStamp;

}

public static long getTimestamp(Timestamp tm) {

try {

return tm.getTime();

} catch (Exception e) {

return 0;

}

}

public static Timestamp getCurrentTimestamp() {

Timestamp timeStamp = null;

try {

timeStamp = new Timestamp(new Date().getTime());

} catch (Exception e) {

}

return timeStamp;

}

public static void main(String[] args)

{

DataUtility d=new DataUtility();

Date date=new Date();

//System.out.println(getDateString(date));

//System.out.println(d.getCurrentTimestamp());

/\*

\* String s1=""; System.out.println(getString("Ravi"));

\* System.out.println(s1.isEmpty());

\* System.out.println(getDate("21/01/1994")); System.out.println(new

\* Date());

\*/

//System.out.println(getDate("01/21/1998"));

System.out.println("formate date :"+getDate("12/09/1991"));

}

}

package com.online.garments.deal.util;

import java.text.ParseException;

import java.util.Date;

public class DataValidator {

public static boolean isName(String val) {

String name = "^[A-Za-z ]\*$";

/\*

\* if (isNotNull(val)) { return val.matches(name);

\*

\* } else { return false; }

\*/

if (val.matches(name)) {

return true;

} else {

return false;

}

}

public static boolean isRollNO(String val) {

String passregex = "^([0-9]{2}[A-Z]{2}[0-9]{1,})\\S$";

if (val.matches(passregex)) {

return true;

} else {

return false;

}

}

public static boolean isPassword(String val) {

String passregex = "^(?=.\*[A-Z])(?=.\*[a-z])(?=.\*[0-9])(?=.\*[\\S])[A-Za-z0-9\\S]{6,12}$";

if (val.matches(passregex)) {

return true;

} else {

return false;

}

}

public static boolean isPhoneNo(String val) {

String regex = "^[7-9][0-9]{9}$";

if (val.matches(regex)) {

return true;

} else {

return false;

}

}

public static boolean isNull(String val) {

if (val == null || val.trim().length() == 0) {

return true;

} else {

return false;

}

}

public static boolean isNotNull(String val) {

return !isNull(val);

}

public static boolean isInteger(String val) {

if (isNotNull(val)) {

try {

int i = Integer.parseInt(val);

return true;

} catch (NumberFormatException e) {

return false;

}

} else {

return false;

}

}

public static boolean isLong(String val) {

if (isNotNull(val)) {

try {

long i = Long.parseLong(val);

return true;

} catch (NumberFormatException e) {

return false;

}

} else {

return false;

}

}

public static boolean isDouble(String val) {

if (isNotNull(val)) {

try {

double i = Double.parseDouble(val);

return true;

} catch (NumberFormatException e) {

return false;

}

} else {

return false;

}

}

/\*public static boolean isIntegerName(String val) {

String match = "^[0-9]{3}$";

if (val.matches(match)) {

return true;

} else {

return false;

}

}\*/

public static boolean isEmail(String val) {

String emailreg = "^[\_A-Za-z0-9-]+(\\.[\_A-Za-z0-9-]+)\*@[A-Za-z0-9]+(\\.[A-Za-z0-9]+)\*(\\.[A-Za-z]{2,})$";

if (isNotNull(val)) {

try {

return val.matches(emailreg);

} catch (NumberFormatException e) {

return false;

}

} else {

return false;

}

}

public static boolean isDate(String val) {

Date d = null;

if (isNotNull(val)) {

d = DataUtility.getDate(val);

}

return d != null;

}

public static void main(String[] args) {

// System.out.println(isPhoneNo("1234567abc"));

/\*

\* System.out.println("Not Null 2" + isNotNull("ABC"));

\* System.out.println("Not Null 3" + isNotNull(null));

\* System.out.println("Not Null 4" + isNull("123"));

\*

\* System.out.println("Is Int " + isInteger(null)); System.out.println(

\* "Is Int " + isInteger("ABC1")); System.out.println("Is Int " +

\* isInteger("123")); System.out.println("Is Int " + isNotNull("123"));

\*/

// System.out.println(isPhoneNo("9926913693"));

// System.out.println(isName("rohan jain"));

System.out.println(isInteger("87"));

System.out.println("os password :"+isPassword("Manoj@123"));

System.out.println("IS ROLL NO :"+isRollNO("18CS01"));

}

}**package** com.online.garments.deal.util;

**import** java.util.HashMap;

**public** **class** EmailBuilder {

**public** **static** String getUserRegistrationMessage(HashMap<String, String> map) {

StringBuilder msg = **new** StringBuilder();

msg.append("<HTML><BODY>");

msg.append("Registration is successful for ORS Project SUNRAYS Technologies");

msg.append("<H1>Hi! Greetings from SUNRAYS Technologies!</H1>");

msg.append(

"<P>Congratulations for registering on ORS! You can now access your ORS account online - anywhere, anytime and enjoy the flexibility to check the Marksheet Details.</P>");

msg.append(

"<P>Log in today at <a href='http://ors.sunraystechnologies.com'>http://ors.sunraystechnologies.com</a> with your following credentials:</P>");

msg.append("<P><B>Login Id : " + map.get("login") + "<BR>" + " Password : " + map.get("password") + "</B></p>");

msg.append(

"<P> As a security measure, we recommended that you change your password after you first log in.</p>");

msg.append(

"<p>For any assistance, please feel free to call us at +91 98273 60504 or 0731-4249244 helpline numbers.</p>");

msg.append("<p>You may also write to us at hrd@sunrays.co.in.</p>");

msg.append(

"<p>We assure you the best service at all times and look forward to a warm and long-standing association with you.</p>");

msg.append("<P><a href='http://www.sunrays.co.in' >-SUNRAYS Technolgies</a></P>");

msg.append("</BODY></HTML>");

**return** msg.toString();

}

**public** **static** String getForgetPasswordMessage(HashMap<String, String> map) {

StringBuilder msg = **new** StringBuilder();

msg.append("<HTML><BODY>");

msg.append("<H1>Your password is reccovered !! " + map.get("firstName") + " " + map.get("lastName") + "</H1>");

msg.append("<P><B>To access account user Login Id : " + map.get("login") + "<BR>" + " Password : "

+ map.get("password") + "</B></p>");

msg.append("</BODY></HTML>");

**return** msg.toString();

}

**public** **static** String getChangePasswordMessage(HashMap<String, String> map) {

StringBuilder msg = **new** StringBuilder();

msg.append("<HTML><BODY>");

msg.append("<H1>Your Password has been changed Successfully !! " + map.get("firstName") + " "

+ map.get("lastName") + "</H1>");

msg.append("<P><B>To access account user Login Id : " + map.get("login") + "<BR>" + " Password : "

+ map.get("password") + "</B></p>");

msg.append("</BODY></HTML>");

**return** msg.toString();

}

**public** **static** **void** main(String[] args) {

HashMap<String, String> map = **new** HashMap<String, String>();

map.put("login", "Rohan123");

map.put("password", "268651");

System.***out***.println(EmailBuilder.*getUserRegistrationMessage*(map));

}

}

**package** com.online.garments.deal.util;

**public** **class** EmailMessage

{

**private** String to = **null**;

**private** String from = **null**;

**private** String cc = **null**;

**private** String bcc = **null**;

**private** String subject = **null**;

**private** String message = **null**;

**private** **int** messageType = ***TEXT\_MSG***;

**public** **static** **final** **int** ***HTML\_MSG*** = 1;

**public** **static** **final** **int** ***TEXT\_MSG*** = 2;

**public** String getTo() {

**return** to;

}

**public** **void** setTo(String to) {

**this**.to = to;

}

**public** String getFrom() {

**return** from;

}

**public** **void** setFrom(String from) {

**this**.from = from;

}

**public** String getCc() {

**return** cc;

}

**public** **void** setCc(String cc) {

**this**.cc = cc;

}

**public** String getBcc() {

**return** bcc;

}

**public** **void** setBcc(String bcc) {

**this**.bcc = bcc;

}

**public** String getSubject() {

**return** subject;

}

**public** **void** setSubject(String subject) {

**this**.subject = subject;

}

**public** String getMessage() {

**return** message;

}

**public** **void** setMessage(String message) {

**this**.message = message;

}

**public** **int** getMessageType() {

**return** messageType;

}

**public** **void** setMessageType(**int** messageType) {

**this**.messageType = messageType;

}

}

**package** com.online.garments.deal.util;

**import** java.util.Properties;

**import** java.util.ResourceBundle;

**import** javax.mail.Message;

**import** javax.mail.PasswordAuthentication;

**import** javax.mail.Session;

**import** javax.mail.Transport;

**import** javax.mail.internet.InternetAddress;

**import** javax.mail.internet.MimeMessage;

**import** com.online.garments.deal.exception.ApplicationException;

**public** **class** EmailUtility

{

**static** ResourceBundle *rb* = ResourceBundle.*getBundle*("com.online.garments.deal.bundle.system");

**private** **static** **final** String ***SMTP\_HOST\_NAME*** = *rb*.getString("smtp.server");

**private** **static** **final** String ***SMTP\_PORT*** = *rb*.getString("smtp.port");

**private** **static** **final** String ***SSL\_FACTORY*** = "javax.net.ssl.SSLSocketFactory";

**private** **static** **final** String ***emailFromAddress*** = *rb*.getString("email.login");

**private** **static** **final** String ***emailPassword*** = *rb*.getString("email.pwd");

**private** **static** Properties *props* = **new** Properties();

**static** {

*props*.put("mail.smtp.host", ***SMTP\_HOST\_NAME***);

*props*.put("mail.smtp.auth", "true");

*props*.put("mail.debug", "true");

*props*.put("mail.smtp.port", ***SMTP\_PORT***);

*props*.put("mail.smtp.socketFactory.port", ***SMTP\_PORT***);

*props*.put("mail.smtp.socketFactory.class", ***SSL\_FACTORY***);

*props*.put("mail.smtp.socketFactory.fallback", "false");

*props*.put("mail.smtp.starttls.enable", "true");

}

**public** **static** **void** sendMail(EmailMessage emailMessageDTO) **throws** ApplicationException{

**try** {

// Connection to Mail Server

Session session = Session.*getDefaultInstance*(*props*,

**new** javax.mail.Authenticator() {

**protected** PasswordAuthentication getPasswordAuthentication() {

**return** **new** PasswordAuthentication(***emailFromAddress***,***emailPassword***);

}

});

// Make debug mode true to display debug messages at console

session.setDebug(**true**);

// Create a message

Message msg = **new** MimeMessage(session);

InternetAddress addressFrom = **new** InternetAddress(***emailFromAddress***);

msg.setFrom(addressFrom);

// Set TO addresses

String[] emailIds = **new** String[0];

**if** (emailMessageDTO.getTo() != **null**) {

emailIds = emailMessageDTO.getTo().split(",");

}

// Set CC addresses

String[] emailIdsCc = **new** String[0];

**if** (emailMessageDTO.getCc() != **null**) {

emailIdsCc = emailMessageDTO.getCc().split(",");

}

// Set BCC addresses

String[] emailIdsBcc = **new** String[0];

**if** (emailMessageDTO.getBcc() != **null**) {

emailIdsBcc = emailMessageDTO.getBcc().split(",");

}

InternetAddress[] addressTo = **new** InternetAddress[emailIds.length];

**for** (**int** i = 0; i < emailIds.length; i++) {

addressTo[i] = **new** InternetAddress(emailIds[i]);

}

InternetAddress[] addressCc = **new** InternetAddress[emailIdsCc.length];

**for** (**int** i = 0; i < emailIdsCc.length; i++) {

addressCc[i] = **new** InternetAddress(emailIdsCc[i]);

}

InternetAddress[] addressBcc = **new** InternetAddress[emailIdsBcc.length];

**for** (**int** i = 0; i < emailIdsBcc.length; i++) {

addressBcc[i] = **new** InternetAddress(emailIdsBcc[i]);

}

**if** (addressTo.length > 0) {

msg.setRecipients(Message.RecipientType.***TO***, addressTo);

}

**if** (addressCc.length > 0) {

msg.setRecipients(Message.RecipientType.***CC***, addressCc);

}

**if** (addressBcc.length > 0) {

msg.setRecipients(Message.RecipientType.***BCC***, addressBcc);

}

// Setting the Subject and Content Type

msg.setSubject(emailMessageDTO.getSubject());

// Set message MIME type

**switch** (emailMessageDTO.getMessageType()) {

**case** EmailMessage.***HTML\_MSG***:

msg.setContent(emailMessageDTO.getMessage(), "text/html");

**break**;

**case** EmailMessage.***TEXT\_MSG***:

msg.setContent(emailMessageDTO.getMessage(), "text/plain");

**break**;

}

// Send the mail

Transport.*send*(msg);

} **catch** (Exception ex) {

**throw** **new** ApplicationException("Email " + ex.getMessage());

}

}

}

package com.online.garments.deal.util;

import java.util.Collections;

import java.util.HashMap;

import java.util.LinkedHashMap;

import java.util.List;

import java.util.Map;

import java.util.Set;

import java.util.TreeMap;

import com.online.garments.deal.bean.DropdownListBean;

public class HTMLUtility {

public static String getList(String name, String selectedVal,HashMap<String, String> map) {

/\*Map<String, String> treeMap = new LinkedHashMap<String, String>(map);\*/

StringBuffer sb = new StringBuffer( "<select class='form-control' name='" + name + "'>");

Set<String> keys = map.keySet();

String val = null;

String select = "-----Select-----";

sb.append("<option selected value='" + select + "'>" + select+ "</option>");

for (String key : keys) {

val = map.get(key);

if (key.trim().equals(selectedVal)) {

sb.append("<option selected value='" + key + "'>" + val

+ "</option>");

} else {

sb.append("<option value='" + key + "'>" + val + "</option>");

}

}

sb.append("</select>");

return sb.toString();

}

public static String getList(String name, String selectedVal, List list) {

Collections.sort(list);

List<DropdownListBean> dd = (List<DropdownListBean>) list;

StringBuffer sb = new StringBuffer("<select class='form-control' name='" + name + "' id='" + name+ "'>");

String key = null;

String val = null;

String select = "-----Select-----";

sb.append("<option selected value='" + select + "'>" + select+ "</option>");

for (DropdownListBean obj : dd) {

key = obj.getKey();

val = obj.getValue();

if (key.trim().equals(selectedVal)) {

sb.append("<option selected value='" + key + "'>" + val+ "</option>");

} else {

sb.append("<option value='" + key + "'>" + val + "</option>");

}

}

sb.append("</select>");

return sb.toString();

}

}package com.online.garments.deal.util;

import java.sql.Connection;

import java.sql.SQLException;

import java.util.ResourceBundle;

import com.mchange.v2.c3p0.ComboPooledDataSource;

import com.online.garments.deal.exception.ApplicationException;

public class JDBCDataSource

{

private static JDBCDataSource datasource;

private JDBCDataSource() {

}

private ComboPooledDataSource cpds = null;

public static JDBCDataSource getInstance() {

if (datasource == null) {

ResourceBundle rb = ResourceBundle.getBundle("com.online.garments.deal.bundle.system");

datasource = new JDBCDataSource();

datasource.cpds = new ComboPooledDataSource();

try {

datasource.cpds.setDriverClass(rb.getString("driver"));

} catch (Exception e) {

e.printStackTrace();

}

datasource.cpds.setJdbcUrl(rb.getString("url"));

datasource.cpds.setUser(rb.getString("username"));

datasource.cpds.setPassword(rb.getString("password"));

datasource.cpds.setInitialPoolSize(new Integer((String) rb .getString("initialPoolSize")));

datasource.cpds.setAcquireIncrement(new Integer((String) rb.getString("acquireIncrement")));

datasource.cpds.setMaxPoolSize(new Integer((String) rb.getString("maxPoolSize")));

datasource.cpds.setMaxIdleTime(DataUtility.getInt(rb.getString("timeout")));

datasource.cpds.setMinPoolSize(new Integer((String) rb.getString("minPoolSize")));

}

return datasource;

}

public static Connection getConnection() throws Exception {

return getInstance().cpds.getConnection();

}

public static void closeConnection(Connection connection) {

if (connection != null) {

try {

connection.close();

} catch (Exception e) {

}

}

}

public static void trnRollback(Connection connection)

throws ApplicationException {

if (connection != null) {

try {

connection.rollback();

} catch (SQLException ex) {

throw new ApplicationException(ex.toString());

}

}

}

}**package** com.online.garments.deal.util;

**import** java.util.ResourceBundle;

**public** **class** PropertyReader

{

**private** **static** ResourceBundle *rb* = ResourceBundle.*getBundle*("com.online.garments.deal.bundle.system");

**public** **static** String getValue(String key) {

String val = **null**;

**try** {

val = *rb*.getString(key);

} **catch** (Exception e) {

val = key;

}

**return** val;

}

**public** **static** String getValue(String key, String param) {

String msg = *getValue*(key);

msg = msg.replace("{0}", param);

**return** msg;

}

**public** **static** String getValue(String key, String[] params) {

String msg = *getValue*(key);

**for** (**int** i = 0; i < params.length; i++) {

msg = msg.replace("{" + i + "}", params[i]);

}

**return** msg;

}

**public** **static** **void** main(String[] args) {

String params = "email" ;

System.***out***.println(PropertyReader.*getValue*("requires", params));

}

}

package com.online.garments.deal.util;

import java.io.IOException;

import java.util.List;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.controller.BaseCtl;

import com.online.garments.deal.controller.OGDView;

public class ServletUtility {

public static void forward(String page, HttpServletRequest request, HttpServletResponse response)

throws IOException, ServletException {

RequestDispatcher rd = request.getRequestDispatcher(page);

System.out.println(page);

rd.forward(request, response);

}

public static void redirect(String page, HttpServletRequest request, HttpServletResponse response)

throws IOException, ServletException {

response.sendRedirect(page);

}

public static void handleException(Exception e, HttpServletRequest request, HttpServletResponse response)

throws IOException, ServletException {

request.setAttribute("exception", e);

ServletUtility.forward(OGDView.ERROR\_CTL, request, response);

e.printStackTrace();

}

public static String getErrorMessage(String property, HttpServletRequest request) {

String val = (String) request.getAttribute(property);

if (val == null) {

return "";

} else {

return val;

}

}

public static String getMessage(String property, HttpServletRequest request) {

String val = (String) request.getAttribute(property);

if (val == null) {

return "";

} else {

return val;

}

}

public static void setErrorMessage(String msg, HttpServletRequest request) {

request.setAttribute(BaseCtl.MSG\_ERROR, msg);

}

public static String getErrorMessage(HttpServletRequest request) {

String val = (String) request.getAttribute(BaseCtl.MSG\_ERROR);

if (val == null) {

return "";

} else {

return val;

}

}

public static void setSuccessMessage(String msg, HttpServletRequest request) {

request.setAttribute(BaseCtl.MSG\_SUCCESS, msg);

}

public static String getSuccessMessage(HttpServletRequest request) {

String val = (String) request.getAttribute(BaseCtl.MSG\_SUCCESS);

if (val == null) {

return "";

} else {

return val;

}

}

public static void setBean(BaseBean bean, HttpServletRequest request) {

request.setAttribute("bean", bean);

}

public static BaseBean getBean(HttpServletRequest request) {

return (BaseBean) request.getAttribute("bean");

}

public static String getParameter(String property, HttpServletRequest request) {

String val = (String) request.getParameter(property);

if (val == null) {

return "";

} else {

return val;

}

}

public static void setList(List list, HttpServletRequest request) {

request.setAttribute("list", list);

}

public static List getList(HttpServletRequest request) {

return (List) request.getAttribute("list");

}

public static void setPageNo(int pageNo, HttpServletRequest request) {

request.setAttribute("pageNo", pageNo);

}

public static int getPageNo(HttpServletRequest request) {

return (Integer) request.getAttribute("pageNo");

}

public static void setPageSize(int pageSize, HttpServletRequest request) {

request.setAttribute("pageSize", pageSize);

}

public static int getPageSize(HttpServletRequest request) {

return (Integer) request.getAttribute("pageSize");

}

public static void setOpration(String msg, HttpServletRequest request) {

request.setAttribute("Opration", msg);

}

public static String getOpration(HttpServletRequest request) {

String val = (String) request.getAttribute("Opration");

if (val == null) {

return "";

} else {

return val;

}

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.DataValidator;

import com.online.garments.deal.util.ServletUtility;

@WebServlet("/BaseCtl")

public abstract class BaseCtl extends HttpServlet

{

private static final Logger log=Logger.getLogger(BaseCtl.class);

public static final String OP\_SAVE = "Save";

public static final String OP\_CANCEL = "Cancel";

public static final String OP\_DELETE = "Delete";

public static final String OP\_LIST = "List";

public static final String OP\_SEARCH = "Search";

public static final String OP\_VIEW = "View";

public static final String OP\_NEXT = "Next";

public static final String OP\_PREVIOUS = "Previous";

public static final String OP\_NEW = "New";

public static final String OP\_GO = "Go";

public static final String OP\_BACK = "Back";

public static final String OP\_LOG\_OUT = "Logout";

public static final String OP\_RESET = "Reset";

public static final String OP\_PAYMENT = "Payment";

public static final String MSG\_SUCCESS = "success";

public static final String MSG\_ERROR = "error";

public BaseCtl() {

// TODO Auto-generated constructor stub

}

protected boolean validate(HttpServletRequest request) {

return true;

}

protected void preload(HttpServletRequest request) {

}

protected BaseBean populateBean(HttpServletRequest request) {

return null;

}

protected BaseBean populateDTO(BaseBean dto, HttpServletRequest request) {

log.debug("BaseCtl populate DTO method start");

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

String modifiedBy = null;

LoginBean userbean = (LoginBean) request.getSession().getAttribute("user");

if (userbean == null) {

// If record is created without login

createdBy = "root";

modifiedBy = "root";

} else {

modifiedBy = userbean.getLogin();

// If record is created first time

if ("null".equalsIgnoreCase(createdBy)|| DataValidator.isNull(createdBy)) {

createdBy = modifiedBy;

}

}

dto.setCreatedBy(createdBy);

dto.setModifiedBy(modifiedBy);

long cdt = DataUtility.getLong(request.getParameter("createdDatetime"));

if (cdt > 0) {

dto.setCreatedDatetime(DataUtility.getTimestamp(cdt));

} else {

dto.setCreatedDatetime(DataUtility.getCurrentTimestamp());

}

dto.setModifiedDatetime(DataUtility.getCurrentTimestamp());

log.debug("BaseCtl populate DTO method end");

return dto;

}

@Override

protected void service(HttpServletRequest request,HttpServletResponse response) throws ServletException, IOException {

log.debug("BaseCtl service method start");

// Load the preloaded data required to display at HTML form

preload(request);

String op = DataUtility.getString(request.getParameter("operation"));

// Check if operation is not DELETE, VIEW, CANCEL, and NULL then

// perform input data validation

System.out.println("operation ="+op);

if (DataValidator.isNotNull(op) && !OP\_CANCEL.equalsIgnoreCase(op)&& !OP\_VIEW.equalsIgnoreCase(op)&& !OP\_DELETE.equalsIgnoreCase(op)&&!OP\_RESET.equalsIgnoreCase(op)) {

// Check validation, If fail then send back to page with error

// messages

if (!validate(request)) {

BaseBean bean = (BaseBean) populateBean(request);

ServletUtility.setBean(bean, request);

ServletUtility.forward(getView(), request, response);

return;

}

}

log.debug("BaseCtl service method end");

super.service(request, response);

}

protected abstract String getView();

}package com.online.garments.deal.controller;

import java.io.IOException;

import java.util.List;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.bean.PaymentBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.model.PaymentModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "BookListCtl", urlPatterns = { "/ctl/BookListCtl" })

public class BookListCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

private static Logger log = Logger.getLogger(PaymentListCtl.class);

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("PaymentListCtl populateBean method start");

PaymentBean bean = new PaymentBean();

bean.setProductId(DataUtility.getLong(request.getParameter("Pro")));

log.debug("PaymentListCtl populateBean method end");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("PaymentListCtl doGet method start");

List list = null;

int pageNo = 1;

int pageSize = DataUtility.getInt(PropertyReader.getValue("page.size"));

long idd=DataUtility.getLong(request.getParameter("idd"));

PaymentModel model = new PaymentModel();

PaymentBean bean = (PaymentBean) populateBean(request);

try {

if(idd>0) {

PaymentBean pBean=new PaymentBean();

pBean.setId(idd);

model.delete(pBean);

ServletUtility.setSuccessMessage("Data Deleted Succsessfully", request);

}

HttpSession session=request.getSession();

LoginBean lBean=(LoginBean)session.getAttribute("user");

if(lBean.getRole()==3) {

bean.setLogin(lBean.getLogin());

}

list = model.search(bean, pageNo, pageSize);

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("No Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("PaymentListCtl doGet method end");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("PaymentListCtl doPost method start");

List list = null;

int pageNo = DataUtility.getInt(request.getParameter("pageNo"));

int pageSize = DataUtility.getInt(request.getParameter("pageSize"));

pageNo = (pageNo == 0) ? 1 : pageNo;

pageSize = (pageSize == 0) ? DataUtility.getInt(PropertyReader.getValue("page.size")) : pageSize;

PaymentBean bean = (PaymentBean) populateBean(request);

PaymentModel model = new PaymentModel();

String[] ids = request.getParameterValues("ids");

String op = DataUtility.getString(request.getParameter("operation"));

if (OP\_SEARCH.equalsIgnoreCase(op) || OP\_NEXT.equalsIgnoreCase(op) || OP\_PREVIOUS.equalsIgnoreCase(op)) {

if (OP\_SEARCH.equalsIgnoreCase(op)) {

pageNo = 1;

} else if (OP\_NEXT.equalsIgnoreCase(op)) {

pageNo++;

} else if (OP\_PREVIOUS.equalsIgnoreCase(op) && pageNo > 1) {

pageNo--;

}

} else if (OP\_NEW.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

return;

} else if (OP\_DELETE.equalsIgnoreCase(op)) {

pageNo = 1;

if (ids != null && ids.length > 0) {

PaymentBean deletebean = new PaymentBean();

for (String id : ids) {

deletebean.setId(DataUtility.getInt(id));

try {

model.delete(deletebean);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

}

ServletUtility.setSuccessMessage("Data Deleted Successfully", request);

} else {

ServletUtility.setErrorMessage("Select at least one record", request);

}

} else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.BOOK\_LIST\_CTL, request, response);

return;

}

try {

HttpSession session=request.getSession();

LoginBean lBean=(LoginBean)session.getAttribute("user");

if(lBean.getRole()==3) {

bean.setLogin(lBean.getLogin());

}

list = model.search(bean, pageNo, pageSize);

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("NO Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("PaymentListCtl doPost method end");

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.BOOK\_LIST\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.CustomerBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.bean.ManagerBean;

import com.online.garments.deal.bean.PaymentBean;

import com.online.garments.deal.bean.ProductBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.model.CustomerModel;

import com.online.garments.deal.model.LoginModel;

import com.online.garments.deal.model.ManagerModel;

import com.online.garments.deal.model.PaymentModel;

import com.online.garments.deal.model.ProductModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.DataValidator;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "CustomerCtl", urlPatterns = { "/CustomerCtl" })

public class CustomerCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

private static Logger log = Logger.getLogger(CustomerCtl.class);

@Override

protected boolean validate(HttpServletRequest request) {

log.debug("CustomerCtl validate method start");

boolean pass = true;

if (DataValidator.isNull(request.getParameter("name"))) {

request.setAttribute("name",PropertyReader.getValue("error.require", "Name"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("contact")))

{

request.setAttribute("contact",PropertyReader.getValue("error.require", "Contact No"));

pass = false;

}else if(!DataValidator.isPhoneNo(request.getParameter("contact")))

{

request.setAttribute("contact", PropertyReader.getValue("error.invalid","Contact No"));

pass=false;

}

if (DataValidator.isNull(request.getParameter("age")))

{

request.setAttribute("age",PropertyReader.getValue("error.require", "Age"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("address")))

{

request.setAttribute("address",PropertyReader.getValue("error.require", "Address"));

pass = false;

}

String op=DataUtility.getString(request.getParameter("operation"));

HttpSession session=request.getSession();

LoginBean lBean=(LoginBean)session.getAttribute("user");

if(lBean==null) {

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

if (DataValidator.isNull(login)) {

request.setAttribute("login", PropertyReader.getValue("error.require", "Login Id"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("password"))) {

request.setAttribute("password", PropertyReader.getValue("error.require", "Password"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("confirmPassword"))) {

request.setAttribute("confirmPassword", PropertyReader.getValue("error.require", "Confirm Password"));

pass = false;

}

if(!OP\_PAYMENT.equalsIgnoreCase(op)) {

if (!request.getParameter("password").equals(

request.getParameter("confirmPassword"))

&& !"".equals(request.getParameter("confirmPassword"))) {

/\*ServletUtility.setErrorMessage("Confirm Password did not match",

request);\*/

request.setAttribute("confirmPassword", PropertyReader.getValue("error.confirmPassword","Confirm Password"));

pass = false;

}

}

}

if(OP\_PAYMENT.equalsIgnoreCase(op)) {

pass=true;

}

log.debug("CustomerCtl validate method end");

return pass;

}

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("CustomerCtl Method populatebean Started");

CustomerBean bean = new CustomerBean();

bean.setId(DataUtility.getLong(request.getParameter("id")));

bean.setName(DataUtility.getString(request.getParameter("name")));

bean.setAge(DataUtility.getString(request.getParameter("age")));

bean.setContectNo(DataUtility.getString(request.getParameter("contact")));

bean.setAddress(DataUtility.getString(request.getParameter("address")));

HttpSession session=request.getSession();

LoginBean lBean=(LoginBean)session.getAttribute("user");

if(lBean==null) {

bean.setLogin(DataUtility.getString(request.getParameter("login")));

bean.setPassword(DataUtility.getString(request.getParameter("password")));

bean.setConfirmPassword(DataUtility.getString(request

.getParameter("confirmPassword")));

}else {

bean.setLogin(lBean.getLogin());

bean.setPassword(lBean.getPassword());

}

populateDTO(bean, request);

log.debug("CustomerCtl Method populatebean Ended");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("CustomerCtl Method doGet Started");

System.out.println("in do get");

String op = DataUtility.getString(request.getParameter("operation"));

long id = DataUtility.getLong(request.getParameter("id"));

HttpSession session=request.getSession(true);

long pid=DataUtility.getLong(request.getParameter("pid"));

if(pid==0) {

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

}else {

session.setAttribute("pid", pid);

ServletUtility.setOpration("Add", request);

CustomerModel model = new CustomerModel();

if (id > 0 || op != null) {

CustomerBean bean;

try {

bean = model.findByPk(id);

ServletUtility.setOpration("Edit", request);

ServletUtility.setBean(bean, request);

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

}

ServletUtility.forward(getView(), request, response);

}

log.debug("CustomerCtl Method doGett Ended");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("CustomerCtl Method doPost Started");

System.out.println("in do post");

String op = DataUtility.getString(request.getParameter("operation"));

// get model

HttpSession session=request.getSession();

CustomerModel model = new CustomerModel();

long id = DataUtility.getLong(request.getParameter("id"));

ServletUtility.setOpration("Add", request);

if (OP\_SAVE.equalsIgnoreCase(op)) {

CustomerBean bean = (CustomerBean) populateBean(request);

try {

if (id > 0) {

model.update(bean);

ServletUtility.setSuccessMessage("Data is successfully Updated",request);

ServletUtility.setOpration("Edit", request);

ServletUtility.setBean(bean, request);

} else {

session.setAttribute("cusBean", bean);

ServletUtility.forward(OGDView.PAYMENT\_VIEW, request, response);

return;

}

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

} catch (DuplicateRecordException e) {

ServletUtility.setBean(bean, request);

ServletUtility.setErrorMessage(e.getMessage(), request);

ServletUtility.forward(getView(), request, response);

return;

}

}

else if (OP\_DELETE.equalsIgnoreCase(op)) {

CustomerBean bean = (CustomerBean) populateBean(request);

try {

model.delete(bean);

ServletUtility.redirect(OGDView.CUSTOMER\_LIST\_CTL, request,response);

return;

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

} else if (OP\_CANCEL.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.CUSTOMER\_LIST\_CTL, request, response);

return;

}else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.CUSTOMER\_CTL, request, response);

return;

}else if(OP\_PAYMENT.equalsIgnoreCase(op)) {

CustomerBean cBean=(CustomerBean)session.getAttribute("cusBean");

ProductModel pModel=new ProductModel();

long pId=(long)session.getAttribute("pid");

try {

LoginBean lBean=(LoginBean)session.getAttribute("user");

if(lBean==null) {

LoginModel lModel=new LoginModel();

LoginBean lbBean=new LoginBean();

lbBean.setLogin(cBean.getLogin());

lbBean.setPassword(cBean.getPassword());

lbBean.setRole(3L);

lModel.add(lbBean);

}

ProductBean pBean=pModel.findByPk(pId);

cBean.setItemCode(pBean.getItemCode());

if(lBean!=null) {

cBean.setLogin(lBean.getLogin());

}

long pk=model.add(cBean);

PaymentModel pyModel=new PaymentModel();

PaymentBean pyBean=new PaymentBean();

pyBean.setCustomerId(pk);

pyBean.setProductId(pId);

pyBean.setAmount(pBean.getPrice());

pyBean.setLogin(cBean.getLogin());

if(lBean!=null) {

pyBean.setLogin(lBean.getLogin());

}

pyModel.add(pyBean);

ServletUtility.setSuccessMessage("Payment Done Successfully", request);

ServletUtility.forward(OGDView.SUCCESS\_VIEW, request, response);

return;

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

} catch (DuplicateRecordException e) {

ServletUtility.setErrorMessage(e.getMessage(), request);

}

}

log.debug("CustomerCtl Method doPost Ended");

ServletUtility.forward(getView(), request, response);

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.CUSTOMER\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.Filter;

import javax.servlet.FilterChain;

import javax.servlet.FilterConfig;

import javax.servlet.ServletException;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import javax.servlet.annotation.WebFilter;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.log4j.Logger;

import com.online.garments.deal.util.ServletUtility;

@WebFilter(filterName="FrontCtl",urlPatterns={"/ctl/\*","/doc/\*"})

public class FrontController implements Filter {

private static Logger log=Logger.getLogger(FrontController.class);

public void destroy() {

}

public void doFilter(ServletRequest req, ServletResponse resp,FilterChain chain) throws IOException, ServletException {

log.debug("FrontController doFilter method start");

HttpServletRequest request = (HttpServletRequest) req;

HttpServletResponse response = (HttpServletResponse) resp;

HttpSession session = request.getSession(true);

if (session.getAttribute("user") == null) {

ServletUtility.setErrorMessage("Your session has been expired. Please re-Login.", request);

String hitUri=request.getRequestURI();

req.setAttribute("uri", hitUri);

ServletUtility.forward("/LoginCtl", request, response);

} else {

chain.doFilter(req, resp);

System.out.println("respronse front coltoller");

}

log.debug("FrontController doFilter method end");

}

public void init(FilterConfig conf) throws ServletException {

// TODO Auto-generated method stub

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.model.LoginModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.DataValidator;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "LoginCtl", urlPatterns = { "/LoginCtl" })

public class LoginCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

public static final String OP\_REGISTER = "Register";

public static final String OP\_SIGN\_IN = "SignIn";

public static final String OP\_SIGN\_UP = "SignUp";

public static final String OP\_LOG\_OUT = "logout";

public static String HIT\_URI = null;

private static Logger log = Logger.getLogger(LoginCtl.class);

public LoginCtl() {

super();

// TODO Auto-generated constructor stub

}

@Override

protected boolean validate(HttpServletRequest request) {

log.debug("LoginCtl Method validate Started");

boolean pass = true;

String op = request.getParameter("operation");

if (OP\_SIGN\_UP.equals(op) || OP\_LOG\_OUT.equals(op)) {

return pass;

}

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

if (DataValidator.isNull(login)) {

request.setAttribute("login", PropertyReader.getValue("error.require", "Login Id"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("password"))) {

request.setAttribute("password", PropertyReader.getValue("error.require", "Password"));

pass = false;

}

log.debug("LoginCtl Method validate Ended");

return pass;

}

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("LoginCtl Method populateBean Started");

LoginBean bean = new LoginBean();

bean.setId(DataUtility.getLong(request.getParameter("id")));

bean.setLogin(DataUtility.getString(request.getParameter("login")));

bean.setPassword(DataUtility.getString(request.getParameter("password")));

log.debug("LOginCtl Method PopulatedBean End");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

log.debug("Method doGet Started");

HttpSession session = request.getSession(true);

String op = DataUtility.getString(request.getParameter("operation"));

LoginModel model = new LoginModel();

long id = DataUtility.getLong(request.getParameter("id"));

if (id > 0) {

LoginBean loginBean;

try {

loginBean = model.findByPK(id);

ServletUtility.setBean(loginBean, request);

} catch (Exception e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

} else if (OP\_LOG\_OUT.equals(op)) {

session = request.getSession(false);

session.invalidate();

ServletUtility.setSuccessMessage("You have been logged out successfully", request);

ServletUtility.forward(OGDView.LOGIN\_VIEW, request, response);

return;

}

if (session.getAttribute("user") != null) {

ServletUtility.redirect(OGDView.WELCOME\_CTL, request, response);

return;

}

ServletUtility.forward(getView(), request, response);

log.debug("Method doGet end");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

log.debug(" LoginCtl Method doPost Started");

HttpSession session = request.getSession(true);

String op = DataUtility.getString(request.getParameter("operation"));

// get Model

LoginModel model = new LoginModel();

long id = DataUtility.getLong(request.getParameter("id"));

if (OP\_SIGN\_IN.equalsIgnoreCase(op)) {

LoginBean bean = (LoginBean) populateBean(request);

try {

bean = model.authenticate(bean.getLogin(), bean.getPassword());

if (bean != null) {

session.setAttribute("user", bean);

// save state of session remember URL

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

if (uri == null || "null".equalsIgnoreCase(uri)) {

ServletUtility.redirect(OGDView.WELCOME\_CTL, request, response);

return;

} else {

ServletUtility.redirect(uri, request, response);

}

return;

} else {

bean = (LoginBean) populateBean(request);

ServletUtility.setBean(bean, request);

ServletUtility.setErrorMessage("Invalid LoginId And Password", request);

}

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

} else if (OP\_SIGN\_UP.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.USER\_REGISTRATION\_CTL, request, response);

return;

}

ServletUtility.forward(getView(), request, response);

log.debug("UserCtl Method doPost Ended");

}

@Override

protected String getView() {

return OGDView.LOGIN\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.bean.ManagerBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.model.LoginModel;

import com.online.garments.deal.model.ManagerModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.DataValidator;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "ManagerCtl", urlPatterns = { "/ctl/ManagerCtl" })

public class ManagerCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

private static Logger log = Logger.getLogger(ManagerCtl.class);

@Override

protected boolean validate(HttpServletRequest request) {

log.debug("ManagerCtl validate method start");

boolean pass = true;

if (DataValidator.isNull(request.getParameter("name"))) {

request.setAttribute("name",PropertyReader.getValue("error.require", "Name"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("contact")))

{

request.setAttribute("contact",PropertyReader.getValue("error.require", "Contact No"));

pass = false;

}else if(!DataValidator.isPhoneNo(request.getParameter("contact")))

{

request.setAttribute("contact", PropertyReader.getValue("error.invalid","Contact No"));

pass=false;

}

if (DataValidator.isNull(request.getParameter("email")))

{

request.setAttribute("email",PropertyReader.getValue("error.require", "Email "));

pass = false;

} else if (!DataValidator.isEmail(request.getParameter("email"))) {

request.setAttribute("email",PropertyReader.getValue("error.email", "Email "));

pass = false;

}

if (DataValidator.isNull(request.getParameter("address")))

{

request.setAttribute("address",PropertyReader.getValue("error.require", "Address"));

pass = false;

}

if("-----Select-----".equalsIgnoreCase(request.getParameter("role"))){

request.setAttribute("role",PropertyReader.getValue("error.require", "Role"));

pass=false;

}

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

if (DataValidator.isNull(login)) {

request.setAttribute("login",

PropertyReader.getValue("error.require", "Login Id"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("password"))) {

request.setAttribute("password",

PropertyReader.getValue("error.require", "Password"));

pass = false;

} else if (!DataValidator.isPassword(request.getParameter("password"))) {

request.setAttribute("password",

PropertyReader.getValue("error.password", "Password"));

return false;

}else if (!DataValidator.isPassword(request.getParameter("password"))) {

request.setAttribute("password",

PropertyReader.getValue("error.password", "Password"));

return false;

}

if (!request.getParameter("password").equals(

request.getParameter("confirmPassword"))

&& !"".equals(request.getParameter("confirmPassword"))) {

/\*ServletUtility.setErrorMessage("Confirm Password did not match",

request);\*/

request.setAttribute("confirmPassword", PropertyReader.getValue("error.confirmPassword","Confirm Password"));

pass = false;

}

log.debug("ManagerCtl validate method end");

return pass;

}

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("ManagerCtl Method populatebean Started");

ManagerBean bean = new ManagerBean();

bean.setId(DataUtility.getLong(request.getParameter("id")));

bean.setName(DataUtility.getString(request.getParameter("name")));

bean.setRole(DataUtility.getLong(request.getParameter("role")));

bean.setContactNo(DataUtility.getString(request.getParameter("contact")));

bean.setEmailId(DataUtility.getString(request.getParameter("email")));

bean.setAddress(DataUtility.getString(request.getParameter("address")));

bean.setLogin(DataUtility.getString(request.getParameter("login")));

bean.setPassword(DataUtility.getString(request.getParameter("password")));

bean.setConfirmPassword(DataUtility.getString(request

.getParameter("confirmPassword")));

populateDTO(bean, request);

log.debug("ManagerCtl Method populatebean Ended");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("ManagerCtl Method doGet Started");

System.out.println("in do get");

String op = DataUtility.getString(request.getParameter("operation"));

long id = DataUtility.getLong(request.getParameter("id"));

// get model

ServletUtility.setOpration("Add", request);

ManagerModel model = new ManagerModel();

if (id > 0 || op != null) {

ManagerBean bean;

try {

bean = model.findByPk(id);

ServletUtility.setOpration("Edit", request);

ServletUtility.setBean(bean, request);

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

}

ServletUtility.forward(getView(), request, response);

log.debug("ManagerCtl Method doGett Ended");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("ManagerCtl Method doPost Started");

System.out.println("in do post");

String op = DataUtility.getString(request.getParameter("operation"));

// get model

ManagerModel model = new ManagerModel();

long id = DataUtility.getLong(request.getParameter("id"));

ServletUtility.setOpration("Add", request);

if (OP\_SAVE.equalsIgnoreCase(op)) {

ManagerBean bean = (ManagerBean) populateBean(request);

LoginModel lModel=new LoginModel();

LoginBean lBean=new LoginBean();

try {

if (id > 0) {

LoginBean lbBean= lModel.findByLogin(bean.getLogin());

lbBean.setLogin(bean.getLogin());

lbBean.setPassword(bean.getPassword());

lbBean.setRole(bean.getRole());

lModel.update(lbBean);

model.update(bean);

ServletUtility.setSuccessMessage("Data is successfully Updated",request);

ServletUtility.setOpration("Edit", request);

ServletUtility.setBean(bean, request);

} else {

lBean.setLogin(bean.getLogin());

lBean.setPassword(bean.getPassword());

lBean.setRole(bean.getRole());

lModel.add(lBean);

long pk = model.add(bean);

ServletUtility.setSuccessMessage("Data is successfully saved",request);

}

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

} catch (DuplicateRecordException e) {

ServletUtility.setBean(bean, request);

ServletUtility.setErrorMessage(e.getMessage(), request);

}

ServletUtility.forward(getView(), request, response);

}

else if (OP\_DELETE.equalsIgnoreCase(op)) {

ManagerBean bean = (ManagerBean) populateBean(request);

try {

model.delete(bean);

ServletUtility.redirect(OGDView.MANAGER\_CTL, request,response);

return;

} catch (ApplicationException e) {

log.error(e);

ServletUtility.handleException(e, request, response);

return;

}

} else if (OP\_CANCEL.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.MANAGER\_CTL, request, response);

}else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.MANAGER\_CTL, request, response);

return;

}

log.debug("ManagerCtl Method doPost Ended");

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.MANAGER\_VIEW;

}

}**package** com.online.garments.deal.controller;

**public** **interface** OGDView {

**public** String ***APP\_CONTEXT*** = "/ONLINE-GARMENTS-DEAL";

**public** String ***LAYOUT\_VIEW*** = "/BaseLayout.jsp";

**public** String ***PAGE\_FOLDER*** = "/jsp";

**public** String ***JAVA\_DOC\_VIEW*** = ***APP\_CONTEXT*** + "/doc/index.html";

**public** String ***ERROR\_VIEW*** = ***PAGE\_FOLDER*** + "/Error.jsp";

**public** String ***USER\_VIEW*** = ***PAGE\_FOLDER*** + "/UserView.jsp";

**public** String ***USER\_LIST\_VIEW*** = ***PAGE\_FOLDER*** + "/UserListView.jsp";

**public** String ***USER\_REGISTRATION\_VIEW*** = ***PAGE\_FOLDER*** + "/UserRegistrationView.jsp";

**public** String ***PRODUCT\_VIEW*** = ***PAGE\_FOLDER*** + "/ProductView.jsp";

**public** String ***PRODUCT\_LIST\_VIEW*** = ***PAGE\_FOLDER*** + "/ProductListView.jsp";

**public** String ***PAYMENT\_LIST\_VIEW*** = ***PAGE\_FOLDER*** + "/PaymentListView.jsp";

**public** String ***CUSTOMER\_VIEW*** = ***PAGE\_FOLDER*** + "/CustomerView.jsp";

**public** String ***CUSTOMER\_LIST\_VIEW*** = ***PAGE\_FOLDER*** + "/CustomerListView.jsp";

**public** String ***BOOK\_LIST\_VIEW*** = ***PAGE\_FOLDER*** + "/BookListView.jsp";

**public** String ***LOGIN\_VIEW*** = ***PAGE\_FOLDER*** + "/LoginView.jsp";

**public** String ***WELCOME\_VIEW*** = ***PAGE\_FOLDER*** + "/Welcome.jsp";

**public** String ***CHANGE\_PASSWORD\_VIEW*** = ***PAGE\_FOLDER*** + "/ChangePasswordView.jsp";

**public** String ***MY\_PROFILE\_VIEW*** = ***PAGE\_FOLDER*** + "/MyProfileView.jsp";

**public** String ***FORGET\_PASSWORD\_VIEW*** = ***PAGE\_FOLDER*** + "/ForgetPasswordView.jsp";

**public** String ***MANAGER\_VIEW*** = ***PAGE\_FOLDER*** + "/ManagerView.jsp";

**public** String ***PAYMENT\_VIEW*** = ***PAGE\_FOLDER*** + "/PaymentView.jsp";

**public** String ***SUCCESS\_VIEW*** = ***PAGE\_FOLDER*** + "/SuccessView.jsp";

**public** String ***ERROR\_CTL*** = "/ctl/ErrorCtl";

**public** String ***USER\_CTL*** = ***APP\_CONTEXT*** + "/ctl/UserCtl";

**public** String ***USER\_LIST\_CTL*** = ***APP\_CONTEXT*** + "/ctl/UserListCtl";

**public** String ***MANAGER\_CTL*** = ***APP\_CONTEXT*** + "/ctl/ManagerCtl";

**public** String ***PRODUCT\_CTL*** = ***APP\_CONTEXT*** + "/ctl/ProductCtl";

**public** String ***PRODUCT\_LIST\_CTL*** = ***APP\_CONTEXT*** + "/ProductListCtl";

**public** String ***PAYMENT\_LIST\_CTL*** = ***APP\_CONTEXT*** + "/ctl/PaymentListCtl";

**public** String ***CUSTOMER\_CTL*** = ***APP\_CONTEXT*** + "/CustomerCtl";

**public** String ***CUSTOMER\_LIST\_CTL*** = ***APP\_CONTEXT*** + "/ctl/CustomerListCtl";

**public** String ***BOOK\_LIST\_CTL*** = ***APP\_CONTEXT*** + "/ctl/BookListCtl";

**public** String ***USER\_REGISTRATION\_CTL*** = ***APP\_CONTEXT*** + "/UserRegistrationCtl";

**public** String ***LOGIN\_CTL*** = ***APP\_CONTEXT*** + "/LoginCtl";

**public** String ***WELCOME\_CTL*** = ***APP\_CONTEXT*** + "/WelcomeCtl";

**public** String ***LOGOUT\_CTL*** = ***APP\_CONTEXT*** + "/LoginCtl";

**public** String ***GET\_MARKSHEET\_CTL*** = ***APP\_CONTEXT*** + "/ctl/GetMarksheetCtl";

**public** String ***CHANGE\_PASSWORD\_CTL*** = ***APP\_CONTEXT*** + "/ctl/ChangePasswordCtl";

**public** String ***MY\_PROFILE\_CTL*** = ***APP\_CONTEXT*** + "/ctl/MyProfileCtl";

**public** String ***FORGET\_PASSWORD\_CTL*** = ***APP\_CONTEXT*** + "/ForgetPasswordCtl";

**public** String ***MARKSHEET\_MERIT\_LIST\_CTL*** = ***APP\_CONTEXT*** + "/ctl/MarksheetMeritListCtl";

}

package com.online.garments.deal.controller;

import java.io.IOException;

import java.util.List;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.PaymentBean;

import com.online.garments.deal.bean.ProductBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.model.PaymentModel;

import com.online.garments.deal.model.ProductModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "PaymentListCtl", urlPatterns = { "/ctl/PaymentListCtl" })

public class PaymentListCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

private static Logger log = Logger.getLogger(PaymentListCtl.class);

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("PaymentListCtl populateBean method start");

PaymentBean bean = new PaymentBean();

bean.setProductId(DataUtility.getLong(request.getParameter("Pro")));

log.debug("PaymentListCtl populateBean method end");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("PaymentListCtl doGet method start");

List list = null;

int pageNo = 1;

int pageSize = DataUtility.getInt(PropertyReader.getValue("page.size"));

long idd=DataUtility.getLong(request.getParameter("idd"));

PaymentModel model = new PaymentModel();

PaymentBean bean = (PaymentBean) populateBean(request);

try {

if(idd>0) {

PaymentBean pBean=new PaymentBean();

pBean.setId(idd);

model.delete(pBean);

ServletUtility.setSuccessMessage("Data Deleted Succsessfully", request);

}

list = model.search(bean, pageNo, pageSize);

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("No Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("PaymentListCtl doGet method end");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("PaymentListCtl doPost method start");

List list = null;

int pageNo = DataUtility.getInt(request.getParameter("pageNo"));

int pageSize = DataUtility.getInt(request.getParameter("pageSize"));

pageNo = (pageNo == 0) ? 1 : pageNo;

pageSize = (pageSize == 0) ? DataUtility.getInt(PropertyReader.getValue("page.size")) : pageSize;

PaymentBean bean = (PaymentBean) populateBean(request);

PaymentModel model = new PaymentModel();

String[] ids = request.getParameterValues("ids");

String op = DataUtility.getString(request.getParameter("operation"));

if (OP\_SEARCH.equalsIgnoreCase(op) || OP\_NEXT.equalsIgnoreCase(op) || OP\_PREVIOUS.equalsIgnoreCase(op)) {

if (OP\_SEARCH.equalsIgnoreCase(op)) {

pageNo = 1;

} else if (OP\_NEXT.equalsIgnoreCase(op)) {

pageNo++;

} else if (OP\_PREVIOUS.equalsIgnoreCase(op) && pageNo > 1) {

pageNo--;

}

} else if (OP\_NEW.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

return;

} else if (OP\_DELETE.equalsIgnoreCase(op)) {

pageNo = 1;

if (ids != null && ids.length > 0) {

PaymentBean deletebean = new PaymentBean();

for (String id : ids) {

deletebean.setId(DataUtility.getInt(id));

try {

model.delete(deletebean);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

}

ServletUtility.setSuccessMessage("Data Deleted Successfully", request);

} else {

ServletUtility.setErrorMessage("Select at least one record", request);

}

} else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

return;

}

try {

list = model.search(bean, pageNo, pageSize);

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("NO Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("PaymentListCtl doPost method end");

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.PAYMENT\_LIST\_VIEW;

}

}**package** com.online.garments.deal.controller;

**import** java.io.File;

**import** java.io.IOException;

**import** java.nio.file.Paths;

**import** javax.servlet.ServletException;

**import** javax.servlet.annotation.MultipartConfig;

**import** javax.servlet.annotation.WebServlet;

**import** javax.servlet.http.HttpServlet;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

**import** javax.servlet.http.Part;

**import** org.apache.log4j.Logger;

**import** com.online.garments.deal.bean.BaseBean;

**import** com.online.garments.deal.bean.LoginBean;

**import** com.online.garments.deal.bean.ManagerBean;

**import** com.online.garments.deal.bean.ProductBean;

**import** com.online.garments.deal.exception.ApplicationException;

**import** com.online.garments.deal.exception.DuplicateRecordException;

**import** com.online.garments.deal.model.LoginModel;

**import** com.online.garments.deal.model.ManagerModel;

**import** com.online.garments.deal.model.ProductModel;

**import** com.online.garments.deal.util.DataUtility;

**import** com.online.garments.deal.util.DataValidator;

**import** com.online.garments.deal.util.PropertyReader;

**import** com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "ProductCtl", urlPatterns = { "/ctl/ProductCtl" })

@MultipartConfig(maxFileSize = 16177215)

**public** **class** ProductCtl **extends** BaseCtl {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**private** **static** Logger *log* = Logger.*getLogger*(ProductCtl.**class**);

@Override

**protected** **boolean** validate(HttpServletRequest request) {

*log*.debug("ProductCtl validate method start");

**boolean** pass = **true**;

**if** (DataValidator.*isNull*(request.getParameter("name"))) {

request.setAttribute("name", PropertyReader.*getValue*("error.require", "Product Name"));

pass = **false**;

}

**if** (DataValidator.*isNull*(request.getParameter("code"))) {

request.setAttribute("code", PropertyReader.*getValue*("error.require", "Item Code"));

pass = **false**;

}

**if** (DataValidator.*isNull*(request.getParameter("quantity"))) {

request.setAttribute("quantity", PropertyReader.*getValue*("error.require", "Quantity "));

pass = **false**;

}

**if** (DataValidator.*isNull*(request.getParameter("choice"))) {

request.setAttribute("choice", PropertyReader.*getValue*("error.require", "Choice"));

pass = **false**;

}

**if** (DataValidator.*isNull*(request.getParameter("price"))) {

request.setAttribute("price", PropertyReader.*getValue*("error.require", "Price"));

pass = **false**;

}**else** **if** (DataValidator.*isNotNull*(request.getParameter("price"))

&& !DataValidator.*isDouble*(request.getParameter("price"))) {

request.setAttribute("price", PropertyReader.*getValue*("error.double", "Price"));

pass = **false**;

}

Part part = **null**;

**try** {

part = request.getPart("photo");

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (ServletException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

String imgName = Paths.*get*(part.getSubmittedFileName()).getFileName().toString();

**if** (DataValidator.*isNull*(imgName)) {

request.setAttribute("photo", PropertyReader.*getValue*("error.require", "Image"));

pass = **false**;

}

*log*.debug("ProductCtl validate method end");

**return** pass;

}

@Override

**protected** BaseBean populateBean(HttpServletRequest request) {

*log*.debug("ProductCtl Method populatebean Started");

ProductBean bean = **new** ProductBean();

bean.setId(DataUtility.*getLong*(request.getParameter("id")));

bean.setProductName(DataUtility.*getString*(request.getParameter("name")));

bean.setProductChoice(DataUtility.*getString*(request.getParameter("choice")));

bean.setProductQuantity(DataUtility.*getString*(request.getParameter("quantity")));

bean.setItemCode(DataUtility.*getString*(request.getParameter("code")));

bean.setPrice(DataUtility.*getDouble*(request.getParameter("price")));

populateDTO(bean, request);

*log*.debug("ProductCtl Method populatebean Ended");

**return** bean;

}

**protected** **void** doGet(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException {

*log*.debug("ProductCtl Method doGet Started");

System.***out***.println("in do get");

String op = DataUtility.*getString*(request.getParameter("operation"));

**long** id = DataUtility.*getLong*(request.getParameter("id"));

// get model

ServletUtility.*setOpration*("Add", request);

ProductModel model = **new** ProductModel();

**if** (id > 0 || op != **null**) {

ProductBean bean;

**try** {

bean = model.findByPk(id);

ServletUtility.*setOpration*("Edit", request);

ServletUtility.*setBean*(bean, request);

} **catch** (ApplicationException e) {

*log*.error(e);

ServletUtility.*handleException*(e, request, response);

**return**;

}

}

ServletUtility.*forward*(getView(), request, response);

*log*.debug("ProductCtl Method doGett Ended");

}

**protected** **void** doPost(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException {

*log*.debug("ProductCtl Method doPost Started");

System.***out***.println("in do post");

String op = DataUtility.*getString*(request.getParameter("operation"));

// get model

ProductModel model = **new** ProductModel();

**long** id = DataUtility.*getLong*(request.getParameter("id"));

ServletUtility.*setOpration*("Add", request);

**if** (***OP\_SAVE***.equalsIgnoreCase(op)) {

ProductBean bean = (ProductBean) populateBean(request);

bean.setImage(SubImage(request, response));

**try** {

**if** (id > 0) {

model.update(bean);

ServletUtility.*setSuccessMessage*("Data is successfully Updated", request);

ServletUtility.*setOpration*("Edit", request);

ServletUtility.*setBean*(bean, request);

} **else** {

**long** pk = model.add(bean);

ServletUtility.*setSuccessMessage*("Data is successfully saved", request);

}

} **catch** (ApplicationException e) {

*log*.error(e);

ServletUtility.*handleException*(e, request, response);

**return**;

} **catch** (DuplicateRecordException e) {

ServletUtility.*setBean*(bean, request);

ServletUtility.*setErrorMessage*(e.getMessage(), request);

}

ServletUtility.*forward*(getView(), request, response);

}

**else** **if** (***OP\_DELETE***.equalsIgnoreCase(op)) {

ProductBean bean = (ProductBean) populateBean(request);

**try** {

model.delete(bean);

ServletUtility.*redirect*(OGDView.***PRODUCT\_LIST\_CTL***, request, response);

**return**;

} **catch** (ApplicationException e) {

*log*.error(e);

ServletUtility.*handleException*(e, request, response);

**return**;

}

} **else** **if** (***OP\_CANCEL***.equalsIgnoreCase(op)) {

ServletUtility.*redirect*(OGDView.***PRODUCT\_LIST\_CTL***, request, response);

} **else** **if** (***OP\_RESET***.equalsIgnoreCase(op)) {

ServletUtility.*redirect*(OGDView.***PRODUCT\_CTL***, request, response);

**return**;

}

*log*.debug("ProductCtl Method doPost Ended");

}

**protected** String SubImage(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException {

response.setContentType("");

String savePath = DataUtility.*getString*(PropertyReader.*getValue*("imagePath"));

File fileSaveDir = **new** File(savePath);

**if** (!fileSaveDir.exists()) {

fileSaveDir.mkdir();

}

Part part = request.getPart("photo");

String fileName = extractFileName(part);

part.write(savePath + File.***separator*** + fileName);

String filePath = fileName;

System.***out***.println("Path----" + savePath + File.***separator*** + fileName);

**return** fileName;

}

**private** String extractFileName(Part part) {

String contentDisp = part.getHeader("content-disposition");

String[] items = contentDisp.split(";");

**for** (String s : items) {

**if** (s.trim().startsWith("filename")) {

**return** s.substring(s.indexOf("=") + 2, s.length() - 1);

}

}

**return** "";

}

@Override

**protected** String getView() {

// **TODO** Auto-generated method stub

**return** OGDView.***PRODUCT\_VIEW***;

}

}

package com.online.garments.deal.controller;

import java.io.IOException;

import java.util.List;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.ProductBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.model.ProductModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "ProductListCtl", urlPatterns = { "/ProductListCtl" })

public class ProductListCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

private static Logger log = Logger.getLogger(ProductListCtl.class);

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("ProductListCtl populateBean method start");

ProductBean bean = new ProductBean();

bean.setProductName(DataUtility.getString(request.getParameter("name")));

bean.setItemCode(DataUtility.getString(request.getParameter("code")));

log.debug("ProductListCtl populateBean method end");

return bean;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("ProductListCtl doGet method start");

List list = null;

int pageNo = 1;

int pageSize = DataUtility.getInt(PropertyReader.getValue("page.size"));

long idd=DataUtility.getLong(request.getParameter("idd"));

ProductModel model = new ProductModel();

ProductBean bean = (ProductBean) populateBean(request);

try {

if(idd>0) {

ProductBean pBean=new ProductBean();

pBean.setId(idd);

model.delete(pBean);

ServletUtility.setSuccessMessage("Data Deleted Succsessfully", request);

}

list = model.list();

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("No Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("ProductListCtl doGet method end");

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("ProductListCtl doPost method start");

List list = null;

int pageNo = DataUtility.getInt(request.getParameter("pageNo"));

int pageSize = DataUtility.getInt(request.getParameter("pageSize"));

pageNo = (pageNo == 0) ? 1 : pageNo;

pageSize = (pageSize == 0) ? DataUtility.getInt(PropertyReader.getValue("page.size")) : pageSize;

ProductBean bean = (ProductBean) populateBean(request);

ProductModel model = new ProductModel();

String[] ids = request.getParameterValues("ids");

String op = DataUtility.getString(request.getParameter("operation"));

if (OP\_SEARCH.equalsIgnoreCase(op) || OP\_NEXT.equalsIgnoreCase(op) || OP\_PREVIOUS.equalsIgnoreCase(op)) {

if (OP\_SEARCH.equalsIgnoreCase(op)) {

pageNo = 1;

} else if (OP\_NEXT.equalsIgnoreCase(op)) {

pageNo++;

} else if (OP\_PREVIOUS.equalsIgnoreCase(op) && pageNo > 1) {

pageNo--;

}

} else if (OP\_NEW.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_CTL, request, response);

return;

} else if (OP\_DELETE.equalsIgnoreCase(op)) {

pageNo = 1;

if (ids != null && ids.length > 0) {

ProductBean deletebean = new ProductBean();

for (String id : ids) {

deletebean.setId(DataUtility.getInt(id));

try {

model.delete(deletebean);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

}

ServletUtility.setSuccessMessage("Data Deleted Successfully", request);

} else {

ServletUtility.setErrorMessage("Select at least one record", request);

}

} else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.PRODUCT\_LIST\_CTL, request, response);

return;

}

try {

list = model.list();

if (list == null || list.size() == 0) {

ServletUtility.setErrorMessage("NO Record Found", request);

}

ServletUtility.setList(list, request);

ServletUtility.setPageNo(pageNo, request);

ServletUtility.setPageSize(pageSize, request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

log.debug("ProductListCtl doPost method end");

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.PRODUCT\_LIST\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.model.LoginModel;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.DataValidator;

import com.online.garments.deal.util.PropertyReader;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "UserRegistrationCtl", urlPatterns = { "/UserRegistrationCtl" })

public class UserRegistrationCtl extends BaseCtl {

public static final String OP\_SIGN\_UP = "SignUp";

private static Logger log = Logger.getLogger(UserRegistrationCtl.class);

@Override

protected boolean validate(HttpServletRequest request) {

log.debug("UserRegistrationCtl Method validate Started");

boolean pass = true;

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

if (DataValidator.isNull(login)) {

request.setAttribute("login", PropertyReader.getValue("error.require", "Login Id"));

pass = false;

}

if (DataValidator.isNull(request.getParameter("password"))) {

request.setAttribute("password", PropertyReader.getValue("error.require", "Password"));

pass = false;

}

if (!request.getParameter("password").equals(

request.getParameter("confirmPassword"))

&& !"".equals(request.getParameter("confirmPassword"))) {

/\*ServletUtility.setErrorMessage("Confirm Password did not match",

request);\*/

request.setAttribute("confirmPassword", PropertyReader.getValue("error.confirmPassword","Confirm Password"));

pass = false;

}

log.debug("UserRegistrationCtl Method validate Ended");

return pass;

}

@Override

protected BaseBean populateBean(HttpServletRequest request) {

log.debug("UserRegistrationCtl Method populatebean Started");

LoginBean bean = new LoginBean();

bean.setId(DataUtility.getLong(request.getParameter("id")));

bean.setRole(3L);

bean.setLogin(DataUtility.getString(request.getParameter("login")));

bean.setPassword(DataUtility.getString(request.getParameter("password")));

bean.setConfirmPassword(DataUtility.getString(request

.getParameter("confirmPassword")));

populateDTO(bean, request);

log.debug("UserRegistrationCtl Method populatebean Ended");

return bean;

}

private static final long serialVersionUID = 1L;

public UserRegistrationCtl() {

super();

// TODO Auto-generated constructor stub

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

log.debug("UserRegistrationCtl Method doGet Started");

ServletUtility.forward(getView(), request, response);

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

System.out.println("in post method");

log.debug("UserRegistrationCtl Method doPost Started");

String op = DataUtility.getString(request.getParameter("operation"));

// get model

LoginModel model = new LoginModel();

long id = DataUtility.getLong(request.getParameter("id"));

if (OP\_SIGN\_UP.equalsIgnoreCase(op)) {

LoginBean bean = (LoginBean) populateBean(request);

try {

// System.out.println("in try sign up");

long pk = model.add(bean);

//System.out.println("register");

bean.setId(pk);

request.getSession().setAttribute("UserBean", bean);

ServletUtility.setSuccessMessage("User Successfully Registered", request);

ServletUtility.forward(OGDView.USER\_REGISTRATION\_VIEW, request, response);

return;

} catch (DuplicateRecordException e) {

log.error(e);

ServletUtility.setBean(bean, request);

ServletUtility.setErrorMessage(e.getMessage(),

request);

ServletUtility.forward(getView(), request, response);

} catch (ApplicationException e) {

ServletUtility.handleException(e, request, response);

e.printStackTrace();

return;

}

}else if (OP\_RESET.equalsIgnoreCase(op)) {

ServletUtility.redirect(OGDView.USER\_REGISTRATION\_CTL, request, response);

return;

}

log.debug("UserRegistrationCtl Method doPost Ended");

}

@Override

protected String getView() {

return OGDView.USER\_REGISTRATION\_VIEW;

}

}package com.online.garments.deal.controller;

import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.online.garments.deal.util.ServletUtility;

@WebServlet(name = "WelcomeCtl", urlPatterns = { "/WelcomeCtl" })

public class WelcomeCtl extends BaseCtl {

private static final long serialVersionUID = 1L;

/\*public WelcomeCtl() {

super();

// TODO Auto-generated constructor stub

}\*/

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

ServletUtility.forward(OGDView.WELCOME\_VIEW, request, response);

}

@Override

protected String getView() {

// TODO Auto-generated method stub

return OGDView.WELCOME\_VIEW;

}

}**package** com.online.garments.deal.exception;

**public** **class** ApplicationException **extends** Exception

{

**public** ApplicationException(String msg) {

**super**(msg);

}

}

**package** com.online.garments.deal.exception;

**public** **class** DatabaseException **extends** Exception

{

**public** DatabaseException(String msg) {

**super**(msg);

}

}

**package** com.online.garments.deal.exception;

**public** **class** DuplicateRecordException **extends** Exception

{

**public** DuplicateRecordException(String msg) {

**super**(msg);

}

}

**package** com.online.garments.deal.exception;

**public** **class** RecordNotFoundException **extends** Exception

{

**public** RecordNotFoundException(String msg) {

**super**(msg);

}

}

package com.online.garments.deal.util;

import java.sql.Timestamp;

import java.text.SimpleDateFormat;

import java.util.Date;

public class DataUtility

{

public static final String APP\_DATE\_FORMAT = "MM/dd/yyyy";

// public static final String APP\_TIME\_FORMAT = "MM/dd/yyyy HH:mm:ss";

private static final SimpleDateFormat formatter = new SimpleDateFormat(APP\_DATE\_FORMAT);

//private static final SimpleDateFormat timeFormatter = new SimpleDateFormat(APP\_TIME\_FORMAT);

public static String getString(String val) {

if (DataValidator.isNotNull(val)) {

return val.trim();

} else {

return val;

}

}

public static String getStringData(Object val) {

if (val != null) {

return val.toString();

} else {

return "";

}

}

public static int getInt(String val) {

if (DataValidator.isInteger(val)) {

return Integer.parseInt(val);

} else {

return 0;

}

}

public static long getLong(String val) {

if (DataValidator.isLong(val)) {

return Long.parseLong(val);

} else {

return 0;

}

}

public static double getDouble(String val) {

if (DataValidator.isDouble(val)) {

return Double.parseDouble(val);

} else {

return 0.0;

}

}

public static Date getDate(String val) {

Date date = null;

try {

date = formatter.parse(val);

} catch (Exception e) {

}

return date;

}

public static Date getDate1(String val) {

Date date = null;

try {

date = formatter.parse(val);

}catch(Exception e){}

return date;

}

public static String getDateString(Date date) {

try {

if(date!=null) {

return formatter.format(date);

}

else{

return "";

}

} catch (Exception e) {

return "";

}

}

public static Date getDate(Date date, int day) {

return null;

}

public static Timestamp getTimestamp(long l) {

Timestamp timeStamp = null;

try {

timeStamp = new Timestamp(l);

} catch (Exception e) {

return null;

}

return timeStamp;

}

public static Timestamp getTimestamp(String cdt) {

Timestamp timeStamp = null;

try {

// timeStamp = new Timestamp((timeFormatter.parse(cdt)).getTime());

} catch (Exception e) {

return null;

}

return timeStamp;

}

public static long getTimestamp(Timestamp tm) {

try {

return tm.getTime();

} catch (Exception e) {

return 0;

}

}

public static Timestamp getCurrentTimestamp() {

Timestamp timeStamp = null;

try {

timeStamp = new Timestamp(new Date().getTime());

} catch (Exception e) {

}

return timeStamp;

}

public static void main(String[] args)

{

DataUtility d=new DataUtility();

Date date=new Date();

//System.out.println(getDateString(date));

//System.out.println(d.getCurrentTimestamp());

/\*

\* String s1=""; System.out.println(getString("Ravi"));

\* System.out.println(s1.isEmpty());

\* System.out.println(getDate("21/01/1994")); System.out.println(new

\* Date());

\*/

//System.out.println(getDate("01/21/1998"));

System.out.println("formate date :"+getDate("12/09/1991"));

}

}

package com.online.garments.deal.util;

import java.text.ParseException;

import java.util.Date;

public class DataValidator {

public static boolean isName(String val) {

String name = "^[A-Za-z ]\*$";

/\*

\* if (isNotNull(val)) { return val.matches(name);

\*

\* } else { return false; }

\*/

if (val.matches(name)) {

return true;

} else {

return false;

}

}

public static boolean isRollNO(String val) {

String passregex = "^([0-9]{2}[A-Z]{2}[0-9]{1,})\\S$";

if (val.matches(passregex)) {

return true;

} else {

return false;

}

}

public static boolean isPassword(String val) {

String passregex = "^(?=.\*[A-Z])(?=.\*[a-z])(?=.\*[0-9])(?=.\*[\\S])[A-Za-z0-9\\S]{6,12}$";

if (val.matches(passregex)) {

return true;

} else {

return false;

}

}

public static boolean isPhoneNo(String val) {

String regex = "^[7-9][0-9]{9}$";

if (val.matches(regex)) {

return true;

} else {

return false;

}

}

public static boolean isNull(String val) {

if (val == null || val.trim().length() == 0) {

return true;

} else {

return false;

}

}

public static boolean isNotNull(String val) {

return !isNull(val);

}

public static boolean isInteger(String val) {

if (isNotNull(val)) {

try {

int i = Integer.parseInt(val);

return true;

} catch (NumberFormatException e) {

return false;

}

} else {

return false;

}

}

public static boolean isLong(String val) {

if (isNotNull(val)) {

try {

long i = Long.parseLong(val);

return true;

} catch (NumberFormatException e) {

return false;

}

} else {

return false;

}

}

public static boolean isDouble(String val) {

if (isNotNull(val)) {

try {

double i = Double.parseDouble(val);

return true;

} catch (NumberFormatException e) {

return false;

}

} else {

return false;

}

}

/\*public static boolean isIntegerName(String val) {

String match = "^[0-9]{3}$";

if (val.matches(match)) {

return true;

} else {

return false;

}

}\*/

public static boolean isEmail(String val) {

String emailreg = "^[\_A-Za-z0-9-]+(\\.[\_A-Za-z0-9-]+)\*@[A-Za-z0-9]+(\\.[A-Za-z0-9]+)\*(\\.[A-Za-z]{2,})$";

if (isNotNull(val)) {

try {

return val.matches(emailreg);

} catch (NumberFormatException e) {

return false;

}

} else {

return false;

}

}

public static boolean isDate(String val) {

Date d = null;

if (isNotNull(val)) {

d = DataUtility.getDate(val);

}

return d != null;

}

public static void main(String[] args) {

// System.out.println(isPhoneNo("1234567abc"));

/\*

\* System.out.println("Not Null 2" + isNotNull("ABC"));

\* System.out.println("Not Null 3" + isNotNull(null));

\* System.out.println("Not Null 4" + isNull("123"));

\*

\* System.out.println("Is Int " + isInteger(null)); System.out.println(

\* "Is Int " + isInteger("ABC1")); System.out.println("Is Int " +

\* isInteger("123")); System.out.println("Is Int " + isNotNull("123"));

\*/

// System.out.println(isPhoneNo("9926913693"));

// System.out.println(isName("rohan jain"));

System.out.println(isInteger("87"));

System.out.println("os password :"+isPassword("Manoj@123"));

System.out.println("IS ROLL NO :"+isRollNO("18CS01"));

}

}**package** com.online.garments.deal.util;

**import** java.util.HashMap;

**public** **class** EmailBuilder {

**public** **static** String getUserRegistrationMessage(HashMap<String, String> map) {

StringBuilder msg = **new** StringBuilder();

msg.append("<HTML><BODY>");

msg.append("Registration is successful for ORS Project SUNRAYS Technologies");

msg.append("<H1>Hi! Greetings from SUNRAYS Technologies!</H1>");

msg.append(

"<P>Congratulations for registering on ORS! You can now access your ORS account online - anywhere, anytime and enjoy the flexibility to check the Marksheet Details.</P>");

msg.append(

"<P>Log in today at <a href='http://ors.sunraystechnologies.com'>http://ors.sunraystechnologies.com</a> with your following credentials:</P>");

msg.append("<P><B>Login Id : " + map.get("login") + "<BR>" + " Password : " + map.get("password") + "</B></p>");

msg.append(

"<P> As a security measure, we recommended that you change your password after you first log in.</p>");

msg.append(

"<p>For any assistance, please feel free to call us at +91 98273 60504 or 0731-4249244 helpline numbers.</p>");

msg.append("<p>You may also write to us at hrd@sunrays.co.in.</p>");

msg.append(

"<p>We assure you the best service at all times and look forward to a warm and long-standing association with you.</p>");

msg.append("<P><a href='http://www.sunrays.co.in' >-SUNRAYS Technolgies</a></P>");

msg.append("</BODY></HTML>");

**return** msg.toString();

}

**public** **static** String getForgetPasswordMessage(HashMap<String, String> map) {

StringBuilder msg = **new** StringBuilder();

msg.append("<HTML><BODY>");

msg.append("<H1>Your password is reccovered !! " + map.get("firstName") + " " + map.get("lastName") + "</H1>");

msg.append("<P><B>To access account user Login Id : " + map.get("login") + "<BR>" + " Password : "

+ map.get("password") + "</B></p>");

msg.append("</BODY></HTML>");

**return** msg.toString();

}

**public** **static** String getChangePasswordMessage(HashMap<String, String> map) {

StringBuilder msg = **new** StringBuilder();

msg.append("<HTML><BODY>");

msg.append("<H1>Your Password has been changed Successfully !! " + map.get("firstName") + " "

+ map.get("lastName") + "</H1>");

msg.append("<P><B>To access account user Login Id : " + map.get("login") + "<BR>" + " Password : "

+ map.get("password") + "</B></p>");

msg.append("</BODY></HTML>");

**return** msg.toString();

}

**public** **static** **void** main(String[] args) {

HashMap<String, String> map = **new** HashMap<String, String>();

map.put("login", "Rohan123");

map.put("password", "268651");

System.***out***.println(EmailBuilder.*getUserRegistrationMessage*(map));

}

}

**package** com.online.garments.deal.util;

**public** **class** EmailMessage

{

**private** String to = **null**;

**private** String from = **null**;

**private** String cc = **null**;

**private** String bcc = **null**;

**private** String subject = **null**;

**private** String message = **null**;

**private** **int** messageType = ***TEXT\_MSG***;

**public** **static** **final** **int** ***HTML\_MSG*** = 1;

**public** **static** **final** **int** ***TEXT\_MSG*** = 2;

**public** String getTo() {

**return** to;

}

**public** **void** setTo(String to) {

**this**.to = to;

}

**public** String getFrom() {

**return** from;

}

**public** **void** setFrom(String from) {

**this**.from = from;

}

**public** String getCc() {

**return** cc;

}

**public** **void** setCc(String cc) {

**this**.cc = cc;

}

**public** String getBcc() {

**return** bcc;

}

**public** **void** setBcc(String bcc) {

**this**.bcc = bcc;

}

**public** String getSubject() {

**return** subject;

}

**public** **void** setSubject(String subject) {

**this**.subject = subject;

}

**public** String getMessage() {

**return** message;

}

**public** **void** setMessage(String message) {

**this**.message = message;

}

**public** **int** getMessageType() {

**return** messageType;

}

**public** **void** setMessageType(**int** messageType) {

**this**.messageType = messageType;

}

}

package com.online.garments.deal.util;

import java.util.Properties;

import java.util.ResourceBundle;

import javax.mail.Message;

import javax.mail.PasswordAuthentication;

import javax.mail.Session;

import javax.mail.Transport;

import javax.mail.internet.InternetAddress;

import javax.mail.internet.MimeMessage;

import com.online.garments.deal.exception.ApplicationException;

public class EmailUtility

{

static ResourceBundle rb = ResourceBundle.getBundle("com.online.garments.deal.bundle.system");

private static final String SMTP\_HOST\_NAME = rb.getString("smtp.server");

private static final String SMTP\_PORT = rb.getString("smtp.port");

private static final String SSL\_FACTORY = "javax.net.ssl.SSLSocketFactory";

private static final String emailFromAddress = rb.getString("email.login");

private static final String emailPassword = rb.getString("email.pwd");

private static Properties props = new Properties();

static {

props.put("mail.smtp.host", SMTP\_HOST\_NAME);

props.put("mail.smtp.auth", "true");

props.put("mail.debug", "true");

props.put("mail.smtp.port", SMTP\_PORT);

props.put("mail.smtp.socketFactory.port", SMTP\_PORT);

props.put("mail.smtp.socketFactory.class", SSL\_FACTORY);

props.put("mail.smtp.socketFactory.fallback", "false");

props.put("mail.smtp.starttls.enable", "true");

}

public static void sendMail(EmailMessage emailMessageDTO) throws ApplicationException{

try {

// Connection to Mail Server

Session session = Session.getDefaultInstance(props,

new javax.mail.Authenticator() {

protected PasswordAuthentication getPasswordAuthentication() {

return new PasswordAuthentication(emailFromAddress,emailPassword);

}

});

// Make debug mode true to display debug messages at console

session.setDebug(true);

// Create a message

Message msg = new MimeMessage(session);

InternetAddress addressFrom = new InternetAddress(emailFromAddress);

msg.setFrom(addressFrom);

// Set TO addresses

String[] emailIds = new String[0];

if (emailMessageDTO.getTo() != null) {

emailIds = emailMessageDTO.getTo().split(",");

}

// Set CC addresses

String[] emailIdsCc = new String[0];

if (emailMessageDTO.getCc() != null) {

emailIdsCc = emailMessageDTO.getCc().split(",");

}

// Set BCC addresses

String[] emailIdsBcc = new String[0];

if (emailMessageDTO.getBcc() != null) {

emailIdsBcc = emailMessageDTO.getBcc().split(",");

}

InternetAddress[] addressTo = new InternetAddress[emailIds.length];

for (int i = 0; i < emailIds.length; i++) {

addressTo[i] = new InternetAddress(emailIds[i]);

}

InternetAddress[] addressCc = new InternetAddress[emailIdsCc.length];

for (int i = 0; i < emailIdsCc.length; i++) {

addressCc[i] = new InternetAddress(emailIdsCc[i]);

}

InternetAddress[] addressBcc = new InternetAddress[emailIdsBcc.length];

for (int i = 0; i < emailIdsBcc.length; i++) {

addressBcc[i] = new InternetAddress(emailIdsBcc[i]);

}

if (addressTo.length > 0) {

msg.setRecipients(Message.RecipientType.TO, addressTo);

}

if (addressCc.length > 0) {

msg.setRecipients(Message.RecipientType.CC, addressCc);

}

if (addressBcc.length > 0) {

msg.setRecipients(Message.RecipientType.BCC, addressBcc);

}

// Setting the Subject and Content Type

msg.setSubject(emailMessageDTO.getSubject());

// Set message MIME type

switch (emailMessageDTO.getMessageType()) {

case EmailMessage.HTML\_MSG:

msg.setContent(emailMessageDTO.getMessage(), "text/html");

break;

case EmailMessage.TEXT\_MSG:

msg.setContent(emailMessageDTO.getMessage(), "text/plain");

break;

}

// Send the mail

Transport.send(msg);

} catch (Exception ex) {

throw new ApplicationException("Email " + ex.getMessage());

}

}

}package com.online.garments.deal.util;

import java.util.Collections;

import java.util.HashMap;

import java.util.LinkedHashMap;

import java.util.List;

import java.util.Map;

import java.util.Set;

import java.util.TreeMap;

import com.online.garments.deal.bean.DropdownListBean;

public class HTMLUtility {

public static String getList(String name, String selectedVal,HashMap<String, String> map) {

/\*Map<String, String> treeMap = new LinkedHashMap<String, String>(map);\*/

StringBuffer sb = new StringBuffer( "<select class='form-control' name='" + name + "'>");

Set<String> keys = map.keySet();

String val = null;

String select = "-----Select-----";

sb.append("<option selected value='" + select + "'>" + select+ "</option>");

for (String key : keys) {

val = map.get(key);

if (key.trim().equals(selectedVal)) {

sb.append("<option selected value='" + key + "'>" + val

+ "</option>");

} else {

sb.append("<option value='" + key + "'>" + val + "</option>");

}

}

sb.append("</select>");

return sb.toString();

}

public static String getList(String name, String selectedVal, List list) {

Collections.sort(list);

List<DropdownListBean> dd = (List<DropdownListBean>) list;

StringBuffer sb = new StringBuffer("<select class='form-control' name='" + name + "' id='" + name+ "'>");

String key = null;

String val = null;

String select = "-----Select-----";

sb.append("<option selected value='" + select + "'>" + select+ "</option>");

for (DropdownListBean obj : dd) {

key = obj.getKey();

val = obj.getValue();

if (key.trim().equals(selectedVal)) {

sb.append("<option selected value='" + key + "'>" + val+ "</option>");

} else {

sb.append("<option value='" + key + "'>" + val + "</option>");

}

}

sb.append("</select>");

return sb.toString();

}

}package com.online.garments.deal.util;

import java.sql.Connection;

import java.sql.SQLException;

import java.util.ResourceBundle;

import com.mchange.v2.c3p0.ComboPooledDataSource;

import com.online.garments.deal.exception.ApplicationException;

public class JDBCDataSource

{

private static JDBCDataSource datasource;

private JDBCDataSource() {

}

private ComboPooledDataSource cpds = null;

public static JDBCDataSource getInstance() {

if (datasource == null) {

ResourceBundle rb = ResourceBundle.getBundle("com.online.garments.deal.bundle.system");

datasource = new JDBCDataSource();

datasource.cpds = new ComboPooledDataSource();

try {

datasource.cpds.setDriverClass(rb.getString("driver"));

} catch (Exception e) {

e.printStackTrace();

}

datasource.cpds.setJdbcUrl(rb.getString("url"));

datasource.cpds.setUser(rb.getString("username"));

datasource.cpds.setPassword(rb.getString("password"));

datasource.cpds.setInitialPoolSize(new Integer((String) rb .getString("initialPoolSize")));

datasource.cpds.setAcquireIncrement(new Integer((String) rb.getString("acquireIncrement")));

datasource.cpds.setMaxPoolSize(new Integer((String) rb.getString("maxPoolSize")));

datasource.cpds.setMaxIdleTime(DataUtility.getInt(rb.getString("timeout")));

datasource.cpds.setMinPoolSize(new Integer((String) rb.getString("minPoolSize")));

}

return datasource;

}

public static Connection getConnection() throws Exception {

return getInstance().cpds.getConnection();

}

public static void closeConnection(Connection connection) {

if (connection != null) {

try {

connection.close();

} catch (Exception e) {

}

}

}

public static void trnRollback(Connection connection)

throws ApplicationException {

if (connection != null) {

try {

connection.rollback();

} catch (SQLException ex) {

throw new ApplicationException(ex.toString());

}

}

}

}**package** com.online.garments.deal.util;

**import** java.util.ResourceBundle;

**public** **class** PropertyReader

{

**private** **static** ResourceBundle *rb* = ResourceBundle.*getBundle*("com.online.garments.deal.bundle.system");

**public** **static** String getValue(String key) {

String val = **null**;

**try** {

val = *rb*.getString(key);

} **catch** (Exception e) {

val = key;

}

**return** val;

}

**public** **static** String getValue(String key, String param) {

String msg = *getValue*(key);

msg = msg.replace("{0}", param);

**return** msg;

}

**public** **static** String getValue(String key, String[] params) {

String msg = *getValue*(key);

**for** (**int** i = 0; i < params.length; i++) {

msg = msg.replace("{" + i + "}", params[i]);

}

**return** msg;

}

**public** **static** **void** main(String[] args) {

String params = "email" ;

System.***out***.println(PropertyReader.*getValue*("requires", params));

}

}

package com.online.garments.deal.util;

import java.io.IOException;

import java.util.List;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.online.garments.deal.bean.BaseBean;

import com.online.garments.deal.controller.BaseCtl;

import com.online.garments.deal.controller.OGDView;

public class ServletUtility {

public static void forward(String page, HttpServletRequest request, HttpServletResponse response)

throws IOException, ServletException {

RequestDispatcher rd = request.getRequestDispatcher(page);

System.out.println(page);

rd.forward(request, response);

}

public static void redirect(String page, HttpServletRequest request, HttpServletResponse response)

throws IOException, ServletException {

response.sendRedirect(page);

}

public static void handleException(Exception e, HttpServletRequest request, HttpServletResponse response)

throws IOException, ServletException {

request.setAttribute("exception", e);

ServletUtility.forward(OGDView.ERROR\_CTL, request, response);

e.printStackTrace();

}

public static String getErrorMessage(String property, HttpServletRequest request) {

String val = (String) request.getAttribute(property);

if (val == null) {

return "";

} else {

return val;

}

}

public static String getMessage(String property, HttpServletRequest request) {

String val = (String) request.getAttribute(property);

if (val == null) {

return "";

} else {

return val;

}

}

public static void setErrorMessage(String msg, HttpServletRequest request) {

request.setAttribute(BaseCtl.MSG\_ERROR, msg);

}

public static String getErrorMessage(HttpServletRequest request) {

String val = (String) request.getAttribute(BaseCtl.MSG\_ERROR);

if (val == null) {

return "";

} else {

return val;

}

}

public static void setSuccessMessage(String msg, HttpServletRequest request) {

request.setAttribute(BaseCtl.MSG\_SUCCESS, msg);

}

public static String getSuccessMessage(HttpServletRequest request) {

String val = (String) request.getAttribute(BaseCtl.MSG\_SUCCESS);

if (val == null) {

return "";

} else {

return val;

}

}

public static void setBean(BaseBean bean, HttpServletRequest request) {

request.setAttribute("bean", bean);

}

public static BaseBean getBean(HttpServletRequest request) {

return (BaseBean) request.getAttribute("bean");

}

public static String getParameter(String property, HttpServletRequest request) {

String val = (String) request.getParameter(property);

if (val == null) {

return "";

} else {

return val;

}

}

public static void setList(List list, HttpServletRequest request) {

request.setAttribute("list", list);

}

public static List getList(HttpServletRequest request) {

return (List) request.getAttribute("list");

}

public static void setPageNo(int pageNo, HttpServletRequest request) {

request.setAttribute("pageNo", pageNo);

}

public static int getPageNo(HttpServletRequest request) {

return (Integer) request.getAttribute("pageNo");

}

public static void setPageSize(int pageSize, HttpServletRequest request) {

request.setAttribute("pageSize", pageSize);

}

public static int getPageSize(HttpServletRequest request) {

return (Integer) request.getAttribute("pageSize");

}

public static void setOpration(String msg, HttpServletRequest request) {

request.setAttribute("Opration", msg);

}

public static String getOpration(HttpServletRequest request) {

String val = (String) request.getAttribute("Opration");

if (val == null) {

return "";

} else {

return val;

}

}

}**package** com.online.garments.deal.model;

**import** java.sql.Connection;

**import** java.sql.PreparedStatement;

**import** java.sql.ResultSet;

**import** java.util.ArrayList;

**import** java.util.List;

**import** org.apache.log4j.Logger;

**import** com.online.garments.deal.bean.CustomerBean;

**import** com.online.garments.deal.bean.LoginBean;

**import** com.online.garments.deal.exception.ApplicationException;

**import** com.online.garments.deal.exception.DatabaseException;

**import** com.online.garments.deal.exception.DuplicateRecordException;

**import** com.online.garments.deal.util.JDBCDataSource;

**public** **class** CustomerModel {

**private** **static** Logger *log* = Logger.*getLogger*(CustomerModel.**class**);

**public** Integer nextPK() **throws** DatabaseException {

*log*.debug("Model nextPK Started");

Connection conn = **null**;

**int** pk = 0;

**try** {

conn = JDBCDataSource.*getConnection*();

PreparedStatement pstmt = conn.prepareStatement("SELECT MAX(ID) FROM Customer");

ResultSet rs = pstmt.executeQuery();

**while** (rs.next()) {

pk = rs.getInt(1);

}

rs.close();

} **catch** (Exception e) {

*log*.error("Database Exception..", e);

**throw** **new** DatabaseException("Exception : Exception in getting PK");

} **finally** {

JDBCDataSource.*closeConnection*(conn);

}

*log*.debug("Model nextPK End");

**return** pk + 1;

}

**public** CustomerBean findByName(String name) **throws** ApplicationException {

*log*.debug("Model findBy Name Started");

StringBuffer sql = **new** StringBuffer("SELECT \* FROM Customer WHERE name=?");

CustomerBean bean = **null**;

Connection conn = **null**;

**try** {

conn = JDBCDataSource.*getConnection*();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, name);

ResultSet rs = pstmt.executeQuery();

**while** (rs.next()) {

bean = **new** CustomerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setAge(rs.getString(3));

bean.setAddress(rs.getString(4));

bean.setContectNo(rs.getString(5));

bean.setProductChoice(rs.getString(6));

bean.setItemCode(rs.getString(7));

bean.setMultipleItems(rs.getString(8));

bean.setCreatedBy(rs.getString(9));

bean.setModifiedBy(rs.getString(10));

bean.setCreatedDatetime(rs.getTimestamp(11));

bean.setModifiedDatetime(rs.getTimestamp(12));

bean.setLogin(rs.getString(13));

}

rs.close();

} **catch** (Exception e) {

*log*.error("Database Exception..", e);

**throw** **new** ApplicationException("Exception : Exception in getting Customer by Name");

} **finally** {

JDBCDataSource.*closeConnection*(conn);

}

*log*.debug("Model findBy Name End");

**return** bean;

}

**public** CustomerBean findByContactNo(String contact) **throws** ApplicationException {

*log*.debug("Model findBy contact Started");

StringBuffer sql = **new** StringBuffer("SELECT \* FROM Customer WHERE contactNo=?");

CustomerBean bean = **null**;

Connection conn = **null**;

**try** {

conn = JDBCDataSource.*getConnection*();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, contact);

ResultSet rs = pstmt.executeQuery();

**while** (rs.next()) {

bean = **new** CustomerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setAge(rs.getString(3));

bean.setAddress(rs.getString(4));

bean.setContectNo(rs.getString(5));

bean.setProductChoice(rs.getString(6));

bean.setItemCode(rs.getString(7));

bean.setMultipleItems(rs.getString(8));

bean.setCreatedBy(rs.getString(9));

bean.setModifiedBy(rs.getString(10));

bean.setCreatedDatetime(rs.getTimestamp(11));

bean.setModifiedDatetime(rs.getTimestamp(12));

bean.setLogin(rs.getString(13));

}

rs.close();

} **catch** (Exception e) {

*log*.error("Database Exception..", e);

**throw** **new** ApplicationException("Exception : Exception in getting Customer by contact");

} **finally** {

JDBCDataSource.*closeConnection*(conn);

}

*log*.debug("Model findBy contact End");

**return** bean;

}

**public** CustomerBean findByPk(**long** id) **throws** ApplicationException {

*log*.debug("Model findBy PK Started");

StringBuffer sql = **new** StringBuffer("SELECT \* FROM Customer WHERE ID=?");

CustomerBean bean = **null**;

Connection conn = **null**;

**try** {

conn = JDBCDataSource.*getConnection*();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setLong(1, id);

ResultSet rs = pstmt.executeQuery();

**while** (rs.next()) {

bean = **new** CustomerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setAge(rs.getString(3));

bean.setAddress(rs.getString(4));

bean.setContectNo(rs.getString(5));

bean.setProductChoice(rs.getString(6));

bean.setItemCode(rs.getString(7));

bean.setMultipleItems(rs.getString(8));

bean.setCreatedBy(rs.getString(9));

bean.setModifiedBy(rs.getString(10));

bean.setCreatedDatetime(rs.getTimestamp(11));

bean.setModifiedDatetime(rs.getTimestamp(12));

bean.setLogin(rs.getString(13));

}

rs.close();

} **catch** (Exception e) {

*log*.error("Database Exception..", e);

**throw** **new** ApplicationException("Exception : Exception in getting Customer by Pk");

} **finally** {

JDBCDataSource.*closeConnection*(conn);

}

*log*.debug("Model findBy Pk End");

**return** bean;

}

**public** **long** add(CustomerBean bean) **throws** ApplicationException, DuplicateRecordException {

*log*.debug("Model add Started");

Connection conn = **null**;

**int** pk = 0;

CustomerBean duplicataLogin = findByContactNo(bean.getContectNo());

**if** (duplicataLogin != **null**) {

**throw** **new** DuplicateRecordException("Contact No Id already exists");

}

**try** {

conn = JDBCDataSource.*getConnection*();

pk = nextPK();

// Get auto-generated next primary key

System.***out***.println(pk + " in ModelJDBC");

conn.setAutoCommit(**false**); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("INSERT INTO Customer VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?)");

pstmt.setInt(1, pk);

pstmt.setString(2, bean.getName());

pstmt.setString(3, bean.getAge());

pstmt.setString(4, bean.getAddress());

pstmt.setString(5, bean.getContectNo());

pstmt.setString(6, bean.getProductChoice());

pstmt.setString(7, bean.getItemCode());

pstmt.setString(8, bean.getMultipleItems());

pstmt.setString(9, bean.getCreatedBy());

pstmt.setString(10, bean.getModifiedBy());

pstmt.setTimestamp(11, bean.getCreatedDatetime());

pstmt.setTimestamp(12, bean.getModifiedDatetime());

pstmt.setString(13,bean.getLogin());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} **catch** (Exception e) {

e.printStackTrace();

*log*.error("Database Exception..", e);

**try** {

conn.rollback();

} **catch** (Exception ex) {

**throw** **new** ApplicationException("Exception : add rollback exception " + ex.getMessage());

}

**throw** **new** ApplicationException("Exception : Exception in add Customer");

} **finally** {

JDBCDataSource.*closeConnection*(conn);

}

*log*.debug("Model add End");

**return** pk;

}

**public** **void** delete(CustomerBean bean) **throws** ApplicationException {

*log*.debug("Model delete Started");

Connection conn = **null**;

**try** {

conn = JDBCDataSource.*getConnection*();

conn.setAutoCommit(**false**); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("DELETE FROM Customer WHERE ID=?");

pstmt.setLong(1, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} **catch** (Exception e) {

// log.error("Database Exception..", e);

**try** {

conn.rollback();

} **catch** (Exception ex) {

**throw** **new** ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

**throw** **new** ApplicationException("Exception : Exception in delete Customer");

} **finally** {

JDBCDataSource.*closeConnection*(conn);

}

*log*.debug("Model delete Started");

}

**public** **void** update(CustomerBean bean) **throws** ApplicationException, DuplicateRecordException {

*log*.debug("Model update Started");

Connection conn = **null**;

CustomerBean duplicataCustomer = findByContactNo(bean.getContectNo());

// Check if updated Role already exist

**if** (duplicataCustomer != **null** && duplicataCustomer.getId() != bean.getId()) {

**throw** **new** DuplicateRecordException("Contact No already exists");

}

**try** {

conn = JDBCDataSource.*getConnection*();

conn.setAutoCommit(**false**); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement(

"UPDATE Customer SET Name=?,age=?,Address=?,ContectNo=?,ProductChoice=?,ItemCode=?,MultipleItems=?,CREATEDBY=?,MODIFIEDBY=?,CREATEDDATETIME=?,MODIFIEDDATETIME=?,Login=? WHERE ID=?");

pstmt.setString(1, bean.getName());

pstmt.setString(2, bean.getAge());

pstmt.setString(3, bean.getAddress());

pstmt.setString(4, bean.getContectNo());

pstmt.setString(5, bean.getProductChoice());

pstmt.setString(6, bean.getItemCode());

pstmt.setString(7, bean.getMultipleItems());

pstmt.setString(8, bean.getCreatedBy());

pstmt.setString(9, bean.getModifiedBy());

pstmt.setTimestamp(10, bean.getCreatedDatetime());

pstmt.setTimestamp(11, bean.getModifiedDatetime());

pstmt.setString(12,bean.getLogin());

pstmt.setLong(13, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} **catch** (Exception e) {

*log*.error("Database Exception..", e);

**try** {

conn.rollback();

} **catch** (Exception ex) {

**throw** **new** ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

**throw** **new** ApplicationException("Exception in updating Customer ");

} **finally** {

JDBCDataSource.*closeConnection*(conn);

}

*log*.debug("Model update End");

}

**public** List search(CustomerBean bean) **throws** ApplicationException {

**return** search(bean, 0, 0);

}

**public** List search(CustomerBean bean, **int** pageNo, **int** pageSize)

**throws** ApplicationException {

*log*.debug("Model search Started");

StringBuffer sql = **new** StringBuffer("SELECT \* FROM Customer WHERE 1=1");

**if** (bean != **null**) {

**if** (bean.getId() > 0) {

sql.append(" AND id = " + bean.getId());

}

**if** (bean.getContectNo() != **null** && bean.getContectNo().length() > 0) {

sql.append(" AND ContectNo LIKE '" + bean.getContectNo() + "%'");

}

**if** (bean.getItemCode() != **null** && bean.getItemCode().length() > 0) {

sql.append(" AND ItemCode LIKE '" + bean.getItemCode() + "%'");

}

**if** (bean.getLogin() != **null** && bean.getLogin().length() > 0) {

sql.append(" AND Login LIKE '" + bean.getLogin() + "%'");

}

}

// if page size is greater than zero then apply pagination

**if** (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" Limit " + pageNo + ", " + pageSize);

// sql.append(" limit " + pageNo + "," + pageSize);

}

ArrayList list = **new** ArrayList();

Connection conn = **null**;

**try** {

conn = JDBCDataSource.*getConnection*();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

**while** (rs.next()) {

bean = **new** CustomerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setAge(rs.getString(3));

bean.setAddress(rs.getString(4));

bean.setContectNo(rs.getString(5));

bean.setProductChoice(rs.getString(6));

bean.setItemCode(rs.getString(7));

bean.setMultipleItems(rs.getString(8));

bean.setCreatedBy(rs.getString(9));

bean.setModifiedBy(rs.getString(10));

bean.setCreatedDatetime(rs.getTimestamp(11));

bean.setModifiedDatetime(rs.getTimestamp(12));

bean.setLogin(rs.getString(13));

list.add(bean);

}

rs.close();

} **catch** (Exception e) {

*log*.error("Database Exception..", e);

**throw** **new** ApplicationException(

"Exception : Exception in search Customer");

} **finally** {

JDBCDataSource.*closeConnection*(conn);

}

*log*.debug("Model search End");

**return** list;

}

**public** List list() **throws** ApplicationException {

**return** list(0, 0);

}

**public** List list(**int** pageNo, **int** pageSize) **throws** ApplicationException {

*log*.debug("Model list Started");

ArrayList list = **new** ArrayList();

StringBuffer sql = **new** StringBuffer("select \* from Customer");

// if page size is greater than zero then apply pagination

**if** (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" limit " + pageNo + "," + pageSize);

}

Connection conn = **null**;

**try** {

conn = JDBCDataSource.*getConnection*();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

**while** (rs.next()) {

CustomerBean bean = **new** CustomerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setAge(rs.getString(3));

bean.setAddress(rs.getString(4));

bean.setContectNo(rs.getString(5));

bean.setProductChoice(rs.getString(6));

bean.setItemCode(rs.getString(7));

bean.setMultipleItems(rs.getString(8));

bean.setCreatedBy(rs.getString(9));

bean.setModifiedBy(rs.getString(10));

bean.setCreatedDatetime(rs.getTimestamp(11));

bean.setModifiedDatetime(rs.getTimestamp(12));

bean.setLogin(rs.getString(13));

list.add(bean);

}

rs.close();

} **catch** (Exception e) {

// log.error("Database Exception..", e);

**throw** **new** ApplicationException(

"Exception : Exception in getting list of Customer");

} **finally** {

JDBCDataSource.*closeConnection*(conn);

}

*log*.debug("Model list End");

**return** list;

}

}

package com.online.garments.deal.model;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.util.ArrayList;

import java.util.List;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.LoginBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DatabaseException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.JDBCDataSource;

public class LoginModel {

private static Logger log = Logger.getLogger(LoginModel.class);

public Integer nextPK() throws DatabaseException {

log.debug("Model nextPK Started");

Connection conn = null;

int pk = 0;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement("SELECT MAX(ID) FROM Login");

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

pk = rs.getInt(1);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new DatabaseException("Exception : Exception in getting PK");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model nextPK End");

return pk + 1;

}

public LoginBean findByLogin(String login) throws ApplicationException {

log.debug("Model findBy LoginId Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Login WHERE Login=?");

LoginBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, login);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new LoginBean();

bean.setId(rs.getLong(1));

bean.setLogin(rs.getString(2));

bean.setPassword(rs.getString(3));

bean.setCreatedBy(rs.getString(4));

bean.setModifiedBy(rs.getString(5));

bean.setCreatedDatetime(rs.getTimestamp(6));

bean.setModifiedDatetime(rs.getTimestamp(7));

bean.setRole(rs.getLong(8));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting User by LoginId");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy EmailId End");

return bean;

}

public LoginBean findByPK(long pk) throws ApplicationException {

log.debug("Model findByPK Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Login WHERE ID=?");

LoginBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setLong(1, pk);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new LoginBean();

bean.setId(rs.getLong(1));

bean.setLogin(rs.getString(2));

bean.setPassword(rs.getString(3));

bean.setCreatedBy(rs.getString(4));

bean.setModifiedBy(rs.getString(5));

bean.setCreatedDatetime(rs.getTimestamp(6));

bean.setModifiedDatetime(rs.getTimestamp(7));

bean.setRole(rs.getLong(8));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Login by pk");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findByPK End");

return bean;

}

public long add(LoginBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model add Started");

Connection conn = null;

int pk = 0;

LoginBean duplicataLogin = findByLogin(bean.getLogin());

if (duplicataLogin != null) {

throw new DuplicateRecordException("Login Id already exists");

}

try {

conn = JDBCDataSource.getConnection();

pk = nextPK();

// Get auto-generated next primary key

System.out.println(pk + " in ModelJDBC");

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("INSERT INTO Login VALUES(?,?,?,?,?,?,?,?)");

pstmt.setInt(1, pk);

pstmt.setString(2, bean.getLogin());

pstmt.setString(3, bean.getPassword());

pstmt.setString(4, bean.getCreatedBy());

pstmt.setString(5, bean.getModifiedBy());

pstmt.setTimestamp(6, DataUtility.getCurrentTimestamp());

pstmt.setTimestamp(7, DataUtility.getCurrentTimestamp());

pstmt.setLong(8,bean.getRole());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

e.printStackTrace();

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : add rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in add Login");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model add End");

return pk;

}

public void delete(LoginBean bean) throws ApplicationException {

log.debug("Model delete Started");

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("DELETE FROM Login WHERE ID=?");

pstmt.setLong(1, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in delete Role");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model delete Started");

}

public void update(LoginBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model update Started");

Connection conn = null;

LoginBean duplicataRole = findByLogin(bean.getLogin());

// Check if updated Role already exist

if (duplicataRole != null && duplicataRole.getId() != bean.getId()) {

throw new DuplicateRecordException("Login Id already exists");

}

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement(

"UPDATE login SET Login=?,Password=?,CREATEDBY=?,MODIFIEDBY=?,CREATEDDATETIME=?,MODIFIEDDATETIME=?,Role=? WHERE ID=?");

pstmt.setString(1, bean.getLogin());

pstmt.setString(2, bean.getPassword());

pstmt.setString(3, bean.getCreatedBy());

pstmt.setString(4, bean.getModifiedBy());

pstmt.setTimestamp(5, bean.getCreatedDatetime());

pstmt.setTimestamp(6, bean.getModifiedDatetime());

pstmt.setLong(7,bean.getRole());

pstmt.setLong(8, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception in updating Login ");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model update End");

}

public List search(LoginBean bean) throws ApplicationException {

return search(bean, 0, 0);

}

public List search(LoginBean bean, int pageNo, int pageSize)

throws ApplicationException {

log.debug("Model search Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Login WHERE 1=1");

if (bean != null) {

if (bean.getId() > 0) {

sql.append(" AND id = " + bean.getId());

}

if (bean.getLogin() != null && bean.getLogin().length() > 0) {

sql.append(" AND LOGIN LIKE '" + bean.getLogin() + "%'");

}

}

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" Limit " + pageNo + ", " + pageSize);

// sql.append(" limit " + pageNo + "," + pageSize);

}

ArrayList list = new ArrayList();

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new LoginBean();

bean.setId(rs.getLong(1));

bean.setLogin(rs.getString(2));

bean.setPassword(rs.getString(3));

bean.setCreatedBy(rs.getString(4));

bean.setModifiedBy(rs.getString(5));

bean.setCreatedDatetime(rs.getTimestamp(6));

bean.setModifiedDatetime(rs.getTimestamp(7));

bean.setRole(rs.getLong(8));

list.add(bean);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in search Login");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model search End");

return list;

}

public List list() throws ApplicationException {

return list(0, 0);

}

public List list(int pageNo, int pageSize) throws ApplicationException {

log.debug("Model list Started");

ArrayList list = new ArrayList();

StringBuffer sql = new StringBuffer("select \* from Login");

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" limit " + pageNo + "," + pageSize);

}

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

LoginBean bean = new LoginBean();

bean.setId(rs.getLong(1));

bean.setLogin(rs.getString(2));

bean.setPassword(rs.getString(3));

bean.setCreatedBy(rs.getString(4));

bean.setModifiedBy(rs.getString(5));

bean.setCreatedDatetime(rs.getTimestamp(6));

bean.setModifiedDatetime(rs.getTimestamp(7));

bean.setRole(rs.getLong(8));

list.add(bean);

}

rs.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in getting list of Login");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model list End");

return list;

}

public LoginBean authenticate(String login, String password) throws ApplicationException {

log.debug("Model authenticate Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Login WHERE LOGIN = ? AND PASSWORD = ?");

LoginBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, login);

pstmt.setString(2, password);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new LoginBean();

bean.setId(rs.getLong(1));

bean.setLogin(rs.getString(2));

bean.setPassword(rs.getString(3));

bean.setCreatedBy(rs.getString(4));

bean.setModifiedBy(rs.getString(5));

bean.setCreatedDatetime(rs.getTimestamp(6));

bean.setModifiedDatetime(rs.getTimestamp(7));

bean.setRole(rs.getLong(8));

System.out.println("Usermodel here");

}

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in get Authenticate Login");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model authenticate End");

return bean;

}

}package com.online.garments.deal.model;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.util.ArrayList;

import java.util.List;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.CustomerBean;

import com.online.garments.deal.bean.ManagerBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DatabaseException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.util.JDBCDataSource;

public class ManagerModel {

private static Logger log = Logger.getLogger(ManagerModel.class);

public Integer nextPK() throws DatabaseException {

log.debug("Model nextPK Started");

Connection conn = null;

int pk = 0;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement("SELECT MAX(ID) FROM Manager");

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

pk = rs.getInt(1);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new DatabaseException("Exception : Exception in getting PK");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model nextPK End");

return pk + 1;

}

public ManagerBean findByLogin(String login) throws ApplicationException {

log.debug("Model findBy Name Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM manager WHERE Login=?");

ManagerBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, login);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ManagerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setEmailId(rs.getString(3));

bean.setContactNo(rs.getString(4));

bean.setAddress(rs.getString(5));

bean.setLogin(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Manager by Login");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy Name End");

return bean;

}

public ManagerBean findByPk(long id) throws ApplicationException {

log.debug("Model findBy PK Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Manager WHERE ID=?");

ManagerBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setLong(1, id);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ManagerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setEmailId(rs.getString(3));

bean.setContactNo(rs.getString(4));

bean.setAddress(rs.getString(5));

bean.setLogin(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Manager by Pk");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy Pk End");

return bean;

}

public long add(ManagerBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model add Started");

Connection conn = null;

int pk = 0;

ManagerBean duplicataLogin = findByLogin(bean.getLogin());

if (duplicataLogin != null) {

throw new DuplicateRecordException("Login Id already exists");

}

try {

conn = JDBCDataSource.getConnection();

pk = nextPK();

// Get auto-generated next primary key

System.out.println(pk + " in ModelJDBC");

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("INSERT INTO Manager VALUES(?,?,?,?,?,?,?,?,?,?)");

pstmt.setInt(1, pk);

pstmt.setString(2, bean.getName());

pstmt.setString(3, bean.getEmailId());

pstmt.setString(4, bean.getContactNo());

pstmt.setString(5, bean.getAddress());

pstmt.setString(6, bean.getLogin());

pstmt.setString(7, bean.getCreatedBy());

pstmt.setString(8, bean.getModifiedBy());

pstmt.setTimestamp(9, bean.getCreatedDatetime());

pstmt.setTimestamp(10, bean.getModifiedDatetime());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

e.printStackTrace();

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : add rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in add Manager");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model add End");

return pk;

}

public void delete(ManagerBean bean) throws ApplicationException {

log.debug("Model delete Started");

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("DELETE FROM Manager WHERE ID=?");

pstmt.setLong(1, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in delete Manager");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model delete Started");

}

public void update(ManagerBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model update Started");

Connection conn = null;

ManagerBean duplicataCustomer = findByLogin(bean.getLogin());

// Check if updated Role already exist

if (duplicataCustomer != null && duplicataCustomer.getId() != bean.getId()) {

throw new DuplicateRecordException("Login Id already exists");

}

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement(

"UPDATE Manager SET Name=?,EmailId=?,ContactNo=?,Address=?,Login=?,CREATEDBY=?,MODIFIEDBY=?,CREATEDDATETIME=?,MODIFIEDDATETIME=? WHERE ID=?");

pstmt.setString(1, bean.getName());

pstmt.setString(2, bean.getEmailId());

pstmt.setString(3, bean.getContactNo());

pstmt.setString(4, bean.getAddress());

pstmt.setString(5, bean.getLogin());

pstmt.setString(6, bean.getCreatedBy());

pstmt.setString(7, bean.getModifiedBy());

pstmt.setTimestamp(8, bean.getCreatedDatetime());

pstmt.setTimestamp(9, bean.getModifiedDatetime());

pstmt.setLong(10, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception in updating Manager ");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model update End");

}

public List search(ManagerBean bean) throws ApplicationException {

return search(bean, 0, 0);

}

public List search(ManagerBean bean, int pageNo, int pageSize)

throws ApplicationException {

log.debug("Model search Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Manager WHERE 1=1");

if (bean != null) {

if (bean.getId() > 0) {

sql.append(" AND id = " + bean.getId());

}

if (bean.getName() != null && bean.getName().length() > 0) {

sql.append(" AND Name LIKE '" + bean.getName() + "%'");

}

if (bean.getLogin() != null && bean.getLogin().length() > 0) {

sql.append(" AND Login LIKE '" + bean.getLogin() + "%'");

}

}

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" Limit " + pageNo + ", " + pageSize);

// sql.append(" limit " + pageNo + "," + pageSize);

}

ArrayList list = new ArrayList();

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ManagerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setEmailId(rs.getString(3));

bean.setContactNo(rs.getString(4));

bean.setAddress(rs.getString(5));

bean.setLogin(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

list.add(bean);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in search Manager");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model search End");

return list;

}

public List list() throws ApplicationException {

return list(0, 0);

}

public List list(int pageNo, int pageSize) throws ApplicationException {

log.debug("Model list Started");

ArrayList list = new ArrayList();

StringBuffer sql = new StringBuffer("select \* from Manager");

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" limit " + pageNo + "," + pageSize);

}

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

ManagerBean bean = new ManagerBean();

bean.setId(rs.getLong(1));

bean.setName(rs.getString(2));

bean.setEmailId(rs.getString(3));

bean.setContactNo(rs.getString(4));

bean.setAddress(rs.getString(5));

bean.setLogin(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

list.add(bean);

}

rs.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in getting list of Manager");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model list End");

return list;

}

}package com.online.garments.deal.model;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.util.ArrayList;

import java.util.List;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.PaymentBean;

import com.online.garments.deal.bean.ProductBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DatabaseException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.util.DataUtility;

import com.online.garments.deal.util.JDBCDataSource;

public class PaymentModel {

private static Logger log = Logger.getLogger(PaymentModel.class);

public Integer nextPK() throws DatabaseException {

log.debug("Model nextPK Started");

Connection conn = null;

int pk = 0;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement("SELECT MAX(ID) FROM payment");

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

pk = rs.getInt(1);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new DatabaseException("Exception : Exception in getting PK");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model nextPK End");

return pk + 1;

}

public Integer nextPaymentId() throws DatabaseException {

log.debug("Model nextPK Started");

Connection conn = null;

int pk = 0;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement("SELECT MAX(paymentId) FROM payment");

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

pk = rs.getInt(1);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new DatabaseException("Exception : Exception in getting PK");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model nextPK End");

return pk + 1;

}

public PaymentBean findByPk(long Id) throws ApplicationException {

log.debug("Model findBy Id Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM payment WHERE Id=? ");

PaymentBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setLong(1, Id);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new PaymentBean();

bean.setId(rs.getLong(1));

bean.setPaymentId(rs.getLong(2));

bean.setCustomerId(rs.getLong(3));

bean.setProductId(rs.getLong(4));

bean.setPaymentDate(rs.getDate(5));

bean.setAmount(rs.getDouble(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setLogin(rs.getString(11));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Payment by Id");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy ID End");

return bean;

}

public PaymentBean findByProductAndCustomer(long productId,long customerId) throws ApplicationException {

log.debug("Model findBy Id Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM payment WHERE productId=? and customerId=? ");

PaymentBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setLong(1, productId);

pstmt.setLong(2, customerId);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new PaymentBean();

bean.setId(rs.getLong(1));

bean.setPaymentId(rs.getLong(2));

bean.setCustomerId(rs.getLong(3));

bean.setProductId(rs.getLong(4));

bean.setPaymentDate(rs.getDate(5));

bean.setAmount(rs.getDouble(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setLogin(rs.getString(11));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Customer And product by Id");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy ID End");

return bean;

}

public long add(PaymentBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model add Started");

Connection conn = null;

int pk = 0;

int paymentId = 0;

PaymentBean duplicataLogin = findByProductAndCustomer(bean.getPaymentId(), bean.getCustomerId());

if (duplicataLogin != null) {

throw new DuplicateRecordException("Payment Done Already ");

}

try {

conn = JDBCDataSource.getConnection();

pk = nextPK();

paymentId=nextPaymentId();

// Get auto-generated next primary key

System.out.println(pk + " in ModelJDBC");

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("INSERT INTO payment VALUES(?,?,?,?,?,?,?,?,?,?,?)");

pstmt.setInt(1, pk);

pstmt.setLong(2, paymentId);

pstmt.setLong(3, bean.getCustomerId());

pstmt.setLong(4, bean.getProductId());

pstmt.setDate(5, new java.sql.Date(new java.util.Date().getTime()));

pstmt.setDouble(6, bean.getAmount());

pstmt.setString(7, bean.getCreatedBy());

pstmt.setString(8, bean.getModifiedBy());

pstmt.setTimestamp(9, DataUtility.getCurrentTimestamp());

pstmt.setTimestamp(10, DataUtility.getCurrentTimestamp());

pstmt.setString(11,bean.getLogin());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

e.printStackTrace();

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : add rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in add Payment");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model add End");

return pk;

}

public void delete(PaymentBean bean) throws ApplicationException {

log.debug("Model delete Started");

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("DELETE FROM Payment WHERE ID=?");

pstmt.setLong(1, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in delete Payment");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model delete Started");

}

public void update(PaymentBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model update Started");

Connection conn = null;

PaymentBean duplicataCustomer = findByProductAndCustomer(bean.getPaymentId(), bean.getCustomerId());

// Check if updated Role already exist

if (duplicataCustomer != null && duplicataCustomer.getId() != bean.getId()) {

throw new DuplicateRecordException("Payment is already Done");

}

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement(

"UPDATE payment SET PaymentId=?,CustomerId=?,ProductId=?,paymentDate=?,Amount=?,CREATEDBY=?,MODIFIEDBY=?,CREATEDDATETIME=?,MODIFIEDDATETIME=?,login=? WHERE ID=?");

pstmt.setLong(1, bean.getPaymentId());

pstmt.setLong(2, bean.getCustomerId());

pstmt.setLong(3, bean.getProductId());

pstmt.setDate(4, new java.sql.Date( bean.getPaymentDate().getTime()));

pstmt.setDouble(5, bean.getAmount());

pstmt.setString(6, bean.getCreatedBy());

pstmt.setString(7, bean.getModifiedBy());

pstmt.setTimestamp(8, bean.getCreatedDatetime());

pstmt.setTimestamp(9, bean.getModifiedDatetime());

pstmt.setString(10, bean.getLogin());

pstmt.setLong(11, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception in updating Payment");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model update End");

}

public List search(PaymentBean bean) throws ApplicationException {

return search(bean, 0, 0);

}

public List search(PaymentBean bean, int pageNo, int pageSize)

throws ApplicationException {

log.debug("Model search Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Payment WHERE 1=1");

if (bean != null) {

if (bean.getId() > 0) {

sql.append(" AND id = " + bean.getId());

}

if (bean.getPaymentId() > 0) {

sql.append(" AND PaymentId = " + bean.getPaymentId());

}

if (bean.getCustomerId() > 0) {

sql.append(" AND CustomerId = " + bean.getCustomerId());

}

if (bean.getProductId() > 0) {

sql.append(" AND ProductId = " + bean.getProductId());

}

if (bean.getLogin() != null && bean.getLogin().length() > 0) {

sql.append(" AND Login LIKE '" + bean.getLogin() + "%'");

}

}

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" Limit " + pageNo + ", " + pageSize);

// sql.append(" limit " + pageNo + "," + pageSize);

}

ArrayList list = new ArrayList();

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new PaymentBean();

bean.setId(rs.getLong(1));

bean.setPaymentId(rs.getLong(2));

bean.setCustomerId(rs.getLong(3));

bean.setProductId(rs.getLong(4));

bean.setPaymentDate(rs.getDate(5));

bean.setAmount(rs.getDouble(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setLogin(rs.getString(11));

list.add(bean);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in search Payment");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model search End");

return list;

}

public List list() throws ApplicationException {

return list(0, 0);

}

public List list(int pageNo, int pageSize) throws ApplicationException {

log.debug("Model list Started");

ArrayList list = new ArrayList();

StringBuffer sql = new StringBuffer("select \* from Payment");

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" limit " + pageNo + "," + pageSize);

}

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

PaymentBean bean = new PaymentBean();

bean.setId(rs.getLong(1));

bean.setPaymentId(rs.getLong(2));

bean.setCustomerId(rs.getLong(3));

bean.setProductId(rs.getLong(4));

bean.setPaymentDate(rs.getDate(5));

bean.setAmount(rs.getDouble(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setLogin(rs.getString(11));

list.add(bean);

}

rs.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in getting list of Payment");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model list End");

return list;

}

}package com.online.garments.deal.model;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.util.ArrayList;

import java.util.List;

import org.apache.log4j.Logger;

import com.online.garments.deal.bean.CustomerBean;

import com.online.garments.deal.bean.ProductBean;

import com.online.garments.deal.exception.ApplicationException;

import com.online.garments.deal.exception.DatabaseException;

import com.online.garments.deal.exception.DuplicateRecordException;

import com.online.garments.deal.util.JDBCDataSource;

public class ProductModel {

private static Logger log = Logger.getLogger(ProductModel.class);

public Integer nextPK() throws DatabaseException {

log.debug("Model nextPK Started");

Connection conn = null;

int pk = 0;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement("SELECT MAX(ID) FROM Product");

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

pk = rs.getInt(1);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new DatabaseException("Exception : Exception in getting PK");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model nextPK End");

return pk + 1;

}

public ProductBean findByName(String name) throws ApplicationException {

log.debug("Model findByName Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Product WHERE name=?");

ProductBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, name);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ProductBean();

bean.setId(rs.getLong(1));

bean.setProductName(rs.getString(2));

bean.setProductQuantity(rs.getString(3));

bean.setProductChoice(rs.getString(4));

bean.setItemCode(rs.getString(5));

bean.setImage(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setPrice(rs.getDouble(11));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Product by Name");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findByName End");

return bean;

}

public ProductBean findByItemCode(String code) throws ApplicationException {

log.debug("Model findByName Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Product WHERE ItemCode=?");

ProductBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, code);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ProductBean();

bean.setId(rs.getLong(1));

bean.setProductName(rs.getString(2));

bean.setProductQuantity(rs.getString(3));

bean.setProductChoice(rs.getString(4));

bean.setItemCode(rs.getString(5));

bean.setImage(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setPrice(rs.getDouble(11));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Product by Code");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findByName End");

return bean;

}

public ProductBean findByNameAndCode(String name,String code) throws ApplicationException {

log.debug("Model findByNameCode Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Product WHERE Productname=? and itemCode=?");

ProductBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setString(1, name);

pstmt.setString(2, code);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ProductBean();

bean.setId(rs.getLong(1));

bean.setProductName(rs.getString(2));

bean.setProductQuantity(rs.getString(3));

bean.setProductChoice(rs.getString(4));

bean.setItemCode(rs.getString(5));

bean.setImage(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setPrice(rs.getDouble(11));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Product by NameAndCode");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy Name End");

return bean;

}

public ProductBean findByPk(long Id) throws ApplicationException {

log.debug("Model findBy Id Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Product WHERE Id=? ");

ProductBean bean = null;

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

pstmt.setLong(1, Id);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ProductBean();

bean.setId(rs.getLong(1));

bean.setProductName(rs.getString(2));

bean.setProductQuantity(rs.getString(3));

bean.setProductChoice(rs.getString(4));

bean.setItemCode(rs.getString(5));

bean.setImage(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setPrice(rs.getDouble(11));

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException("Exception : Exception in getting Product by Id");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model findBy ID End");

return bean;

}

public long add(ProductBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model add Started");

Connection conn = null;

int pk = 0;

ProductBean duplicataLogin = findByNameAndCode(bean.getProductName(), bean.getItemCode());

if (duplicataLogin != null) {

throw new DuplicateRecordException(" Product is already exists");

}

try {

conn = JDBCDataSource.getConnection();

pk = nextPK();

// Get auto-generated next primary key

System.out.println(pk + " in ModelJDBC");

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("INSERT INTO product VALUES(?,?,?,?,?,?,?,?,?,?,?)");

pstmt.setInt(1, pk);

pstmt.setString(2, bean.getProductName());

pstmt.setString(3, bean.getProductQuantity());

pstmt.setString(4, bean.getProductChoice());

pstmt.setString(5, bean.getItemCode());

pstmt.setString(6, bean.getImage());

pstmt.setString(7, bean.getCreatedBy());

pstmt.setString(8, bean.getModifiedBy());

pstmt.setTimestamp(9, bean.getCreatedDatetime());

pstmt.setTimestamp(10, bean.getModifiedDatetime());

pstmt.setDouble(11, bean.getPrice());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

e.printStackTrace();

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : add rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in add Product");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model add End");

return pk;

}

public void delete(ProductBean bean) throws ApplicationException {

log.debug("Model delete Started");

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement("DELETE FROM Product WHERE ID=?");

pstmt.setLong(1, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception : Exception in delete Customer");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model delete Started");

}

public void update(ProductBean bean) throws ApplicationException, DuplicateRecordException {

log.debug("Model update Started");

Connection conn = null;

ProductBean duplicataCustomer = findByNameAndCode(bean.getProductName(), bean.getItemCode());

// Check if updated Role already exist

if (duplicataCustomer != null && duplicataCustomer.getId() != bean.getId()) {

throw new DuplicateRecordException("Product is already exists");

}

try {

conn = JDBCDataSource.getConnection();

conn.setAutoCommit(false); // Begin transaction

PreparedStatement pstmt = conn.prepareStatement(

"UPDATE Product SET ProductName=?,ProductQuantity=?,ProductChoice=?,ItemCode=?,Image=?,CREATEDBY=?,MODIFIEDBY=?,CREATEDDATETIME=?,MODIFIEDDATETIME=?,price=? WHERE ID=?");

pstmt.setString(1, bean.getProductName());

pstmt.setString(2, bean.getProductQuantity());

pstmt.setString(3, bean.getProductChoice());

pstmt.setString(4, bean.getItemCode());

pstmt.setString(5, bean.getImage());

pstmt.setString(6, bean.getCreatedBy());

pstmt.setString(7, bean.getModifiedBy());

pstmt.setTimestamp(8, bean.getCreatedDatetime());

pstmt.setTimestamp(9, bean.getModifiedDatetime());

pstmt.setDouble(10, bean.getPrice());

pstmt.setLong(11, bean.getId());

pstmt.executeUpdate();

conn.commit(); // End transaction

pstmt.close();

} catch (Exception e) {

log.error("Database Exception..", e);

try {

conn.rollback();

} catch (Exception ex) {

throw new ApplicationException("Exception : Delete rollback exception " + ex.getMessage());

}

throw new ApplicationException("Exception in updating Product ");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model update End");

}

public List search(ProductBean bean) throws ApplicationException {

return search(bean, 0, 0);

}

public List search(ProductBean bean, int pageNo, int pageSize)

throws ApplicationException {

log.debug("Model search Started");

StringBuffer sql = new StringBuffer("SELECT \* FROM Product WHERE 1=1");

if (bean != null) {

if (bean.getId() > 0) {

sql.append(" AND id = " + bean.getId());

}

if (bean.getProductName() != null && bean.getProductName().length() > 0) {

sql.append(" AND ProductName LIKE '" + bean.getProductName() + "%'");

}

if (bean.getItemCode() != null && bean.getItemCode().length() > 0) {

sql.append(" AND ItemCode LIKE '" + bean.getItemCode() + "%'");

}

}

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" Limit " + pageNo + ", " + pageSize);

// sql.append(" limit " + pageNo + "," + pageSize);

}

ArrayList list = new ArrayList();

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

bean = new ProductBean();

bean.setId(rs.getLong(1));

bean.setProductName(rs.getString(2));

bean.setProductQuantity(rs.getString(3));

bean.setProductChoice(rs.getString(4));

bean.setItemCode(rs.getString(5));

bean.setImage(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setPrice(rs.getDouble(11));

list.add(bean);

}

rs.close();

} catch (Exception e) {

log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in search Product");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model search End");

return list;

}

public List list() throws ApplicationException {

return list(0, 0);

}

public List list(int pageNo, int pageSize) throws ApplicationException {

log.debug("Model list Started");

ArrayList list = new ArrayList();

StringBuffer sql = new StringBuffer("select \* from Product");

// if page size is greater than zero then apply pagination

if (pageSize > 0) {

// Calculate start record index

pageNo = (pageNo - 1) \* pageSize;

sql.append(" limit " + pageNo + "," + pageSize);

}

Connection conn = null;

try {

conn = JDBCDataSource.getConnection();

PreparedStatement pstmt = conn.prepareStatement(sql.toString());

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

ProductBean bean = new ProductBean();

bean.setId(rs.getLong(1));

bean.setProductName(rs.getString(2));

bean.setProductQuantity(rs.getString(3));

bean.setProductChoice(rs.getString(4));

bean.setItemCode(rs.getString(5));

bean.setImage(rs.getString(6));

bean.setCreatedBy(rs.getString(7));

bean.setModifiedBy(rs.getString(8));

bean.setCreatedDatetime(rs.getTimestamp(9));

bean.setModifiedDatetime(rs.getTimestamp(10));

bean.setPrice(rs.getDouble(11));

list.add(bean);

}

rs.close();

} catch (Exception e) {

// log.error("Database Exception..", e);

throw new ApplicationException(

"Exception : Exception in getting list of Product");

} finally {

JDBCDataSource.closeConnection(conn);

}

log.debug("Model list End");

return list;

}

}*.modal-content* {

position: *relative*;

background-color: *#ffffff*;

border: *1px* *solid* *#999999*;

border: *1px* *solid* *rgba(0,* *0,* *0,* *0.2)*;

border-radius: *0*;

outline: *none*;

-webkit-box-shadow: *0* *3px* *9px* *rgba(0,* *0,* *0,* *0.5)*;

box-shadow: *0* *3px* *9px* *rgba(0,* *0,* *0,* *0.5)*;

background-clip: *padding-box*;

}

*.modal-header* {

border-bottom: *1px* *solid* *#48a4ac*;

background: *#61bdc5*;

min-height: *16.4286px*;

padding: *10px* *15px*;

}

*.close* {

float: *right*;

font-size: *21px*;

font-weight: *700*;

line-height: *1*;

color: *#000*;

text-shadow: *0* *1px* *0* *#fff*;

opacity: *.2*;

filter: *alpha(opacity*=*20)*;

}

*.modal-body* {

position: *relative*;

padding: *20px*;

}

*.modal-footer* {

padding: *19px* *20px* *20px*;

margin-top: *0*;

text-align: *right*;

border-top: *1px* *solid* *#e5e5e5*;

}**h3***.h3*{text-align:*center*;margin:*1em*;text-transform:*capitalize*;font-size:*1.7em*;}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* shopping Demo-1 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

*.product-grid*{font-family:*Raleway,sans-serif*;text-align:*center*;padding:*0* *0* *72px*;border:*1px* *solid* *rgba(0,0,0,.1)*;overflow:*hidden*;position:*relative*;z-index:*1*}

*.product-grid* *.product-image*{position:*relative*;transition:*all* *.3s* *ease* *0s*}

*.product-grid* *.product-image* **a**{display:*block*}

*.product-grid* *.product-image* **img**{width:*100%*;height:*auto*}

*.product-grid* *.pic-1*{opacity:*1*;transition:*all* *.3s* *ease-out* *0s*}

*.product-grid:hover* *.pic-1*{opacity:*1*}

*.product-grid* *.pic-2*{opacity:*0*;position:*absolute*;top:*0*;left:*0*;transition:*all* *.3s* *ease-out* *0s*}

*.product-grid:hover* *.pic-2*{opacity:*1*}

*.product-grid* *.social*{width:*150px*;padding:*0*;margin:*0*;list-style:*none*;opacity:*0*;transform:*translateY(-50%)* *translateX(-50%)*;position:*absolute*;top:*60%*;left:*50%*;z-index:*1*;transition:*all* *.3s* *ease* *0s*}

*.product-grid:hover* *.social*{opacity:*1*;top:*50%*}

*.product-grid* *.social* **li**{display:*inline-block*}

*.product-grid* *.social* **li** **a**{color:*#fff*;background-color:*#333*;font-size:*16px*;line-height:*40px*;text-align:*center*;height:*40px*;width:*40px*;margin:*0* *2px*;display:*block*;position:*relative*;transition:*all* *.3s* *ease-in-out*}

*.product-grid* *.social* **li** **a***:hover*{color:*#fff*;background-color:*#ef5777*}

*.product-grid* *.social* **li** **a***:after***,***.product-grid* *.social* **li** **a***:before*{content:*attr(data-tip)*;color:*#fff*;background-color:*#000*;font-size:*12px*;letter-spacing:*1px*;line-height:*20px*;padding:*1px* *5px*;white-space:*nowrap*;opacity:*0*;transform:*translateX(-50%)*;position:*absolute*;left:*50%*;top:*-30px*}

*.product-grid* *.social* **li** **a***:after*{content:*''*;height:*15px*;width:*15px*;border-radius:*0*;transform:*translateX(-50%)* *rotate(45deg)*;top:*-20px*;z-index:*-1*}

*.product-grid* *.social* **li** **a***:hover:after***,***.product-grid* *.social* **li** **a***:hover:before*{opacity:*1*}

*.product-grid* *.product-discount-label***,***.product-grid* *.product-new-label*{color:*#fff*;background-color:*#ef5777*;font-size:*12px*;text-transform:*uppercase*;padding:*2px* *7px*;display:*block*;position:*absolute*;top:*10px*;left:*0*}

*.product-grid* *.product-discount-label*{background-color:*#333*;left:*auto*;right:*0*}

*.product-grid* *.rating*{color:*#FFD200*;font-size:*12px*;padding:*12px* *0* *0*;margin:*0*;list-style:*none*;position:*relative*;z-index:*-1*}

*.product-grid* *.rating* **li***.disable*{color:*rgba(0,0,0,.2)*}

*.product-grid* *.product-content*{background-color:*#fff*;text-align:*center*;padding:*12px* *0*;margin:*0* *auto*;position:*absolute*;left:*0*;right:*0*;bottom:*-27px*;z-index:*1*;transition:*all* *.3s*}

*.product-grid:hover* *.product-content*{bottom:*0*}

*.product-grid* *.title*{font-size:*13px*;font-weight:*400*;letter-spacing:*.5px*;text-transform:*capitalize*;margin:*0* *0* *10px*;transition:*all* *.3s* *ease* *0s*}

*.product-grid* *.title* **a**{color:*#828282*}

*.product-grid* *.title* **a***:hover***,***.product-grid:hover* *.title* **a**{color:*#ef5777*}

*.product-grid* *.price*{color:*#333*;font-size:*17px*;font-family:*Montserrat,sans-serif*;font-weight:*700*;letter-spacing:*.6px*;margin-bottom:*8px*;text-align:*center*;transition:*all* *.3s*}

*.product-grid* *.price* **span**{color:*#999*;font-size:*13px*;font-weight:*400*;text-decoration:*line-through*;margin-left:*3px*;display:*inline-block*}

*.product-grid* *.add-to-cart*{color:*#000*;font-size:*13px*;font-weight:*600*}

@media only screen and (max-width:990px){*.product-grid*{margin-bottom:*30px*}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Shopping Demo-2 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

*.demo*{padding:*45px* *0*}

*.product-grid2*{font-family:*'Open Sans',sans-serif*;position:*relative*}

*.product-grid2* *.product-image2*{overflow:*hidden*;position:*relative*}

*.product-grid2* *.product-image2* **a**{display:*block*}

*.product-grid2* *.product-image2* **img**{width:*100%*;height:*auto*}

*.product-image2* *.pic-1*{opacity:*1*;transition:*all* *.5s*}

*.product-grid2:hover* *.product-image2* *.pic-1*{opacity:*0*}

*.product-image2* *.pic-2*{width:*100%*;height:*100%*;opacity:*0*;position:*absolute*;top:*0*;left:*0*;transition:*all* *.5s*}

*.product-grid2:hover* *.product-image2* *.pic-2*{opacity:*1*}

*.product-grid2* *.social*{padding:*0*;margin:*0*;position:*absolute*;bottom:*50px*;right:*25px*;z-index:*1*}

*.product-grid2* *.social* **li**{margin:*0* *0* *10px*;display:*block*;transform:*translateX(100px)*;transition:*all* *.5s*}

*.product-grid2:hover* *.social* **li**{transform:*translateX(0)*}

*.product-grid2:hover* *.social* **li***:nth-child*(*2*){transition-delay:*.15s*}

*.product-grid2:hover* *.social* **li***:nth-child*(*3*){transition-delay:*.25s*}

*.product-grid2* *.social* **li** **a**{color:*#505050*;background-color:*#fff*;font-size:*17px*;line-height:*45px*;text-align:*center*;height:*45px*;width:*45px*;border-radius:*50%*;display:*block*;transition:*all* *.3s* *ease* *0s*}

*.product-grid2* *.social* **li** **a***:hover*{color:*#fff*;background-color:*#3498db*;box-shadow:*0* *0* *10px* *rgba(0,0,0,.5)*}

*.product-grid2* *.social* **li** **a***:after***,***.product-grid2* *.social* **li** **a***:before*{content:*attr(data-tip)*;color:*#fff*;background-color:*#000*;font-size:*12px*;line-height:*22px*;border-radius:*3px*;padding:*0* *5px*;white-space:*nowrap*;opacity:*0*;transform:*translateX(-50%)*;position:*absolute*;left:*50%*;top:*-30px*}

*.product-grid2* *.social* **li** **a***:after*{content:*''*;height:*15px*;width:*15px*;border-radius:*0*;transform:*translateX(-50%)* *rotate(45deg)*;top:*-22px*;z-index:*-1*}

*.product-grid2* *.social* **li** **a***:hover:after***,***.product-grid2* *.social* **li** **a***:hover:before*{opacity:*1*}

*.product-grid2* *.add-to-cart*{color:*#fff*;background-color:*#404040*;font-size:*15px*;text-align:*center*;width:*100%*;padding:*10px* *0*;display:*block*;position:*absolute*;left:*0*;bottom:*-100%*;transition:*all* *.3s*}

*.product-grid2* *.add-to-cart:hover*{background-color:*#3498db*;text-decoration:*none*}

*.product-grid2:hover* *.add-to-cart*{bottom:*0*}

*.product-grid2* *.product-new-label*{background-color:*#3498db*;color:*#fff*;font-size:*17px*;padding:*5px* *10px*;position:*absolute*;right:*0*;top:*0*;transition:*all* *.3s*}

*.product-grid2:hover* *.product-new-label*{opacity:*0*}

*.product-grid2* *.product-content*{padding:*20px* *10px*;text-align:*center*}

*.product-grid2* *.title*{font-size:*17px*;margin:*0* *0* *7px*}

*.product-grid2* *.title* **a**{color:*#303030*}

*.product-grid2* *.title* **a***:hover*{color:*#3498db*}

*.product-grid2* *.price*{color:*#303030*;font-size:*15px*}

@media screen and (max-width:990px){*.product-grid2*{margin-bottom:*30px*}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Shopping Demo-3 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

*.product-grid3*{font-family:*Roboto,sans-serif*;text-align:*center*;position:*relative*;z-index:*1*}

*.product-grid3:before*{content:*""*;height:*81%*;width:*100%*;background:*#fff*;border:*1px* *solid* *rgba(0,0,0,.1)*;opacity:*0*;position:*absolute*;top:*0*;left:*0*;z-index:*-1*;transition:*all* *.5s* *ease* *0s*}

*.product-grid3:hover:before*{opacity:*1*;height:*100%*}

*.product-grid3* *.product-image3*{position:*relative*}

*.product-grid3* *.product-image3* **a**{display:*block*}

*.product-grid3* *.product-image3* **img**{width:*100%*;height:*auto*}

*.product-grid3* *.pic-1*{opacity:*1*;transition:*all* *.5s* *ease-out* *0s*}

*.product-grid3:hover* *.pic-1*{opacity:*0*}

*.product-grid3* *.pic-2*{position:*absolute*;top:*0*;left:*0*;opacity:*0*;transition:*all* *.5s* *ease-out* *0s*}

*.product-grid3:hover* *.pic-2*{opacity:*1*}

*.product-grid3* *.social*{width:*120px*;padding:*0*;margin:*0* *auto*;list-style:*none*;opacity:*0*;position:*absolute*;right:*0*;left:*0*;bottom:*-23px*;transform:*scale(0)*;transition:*all* *.3s* *ease* *0s*}

*.product-grid3:hover* *.social*{opacity:*1*;transform:*scale(1)*}

*.product-grid3:hover* *.product-discount-label***,***.product-grid3:hover* *.product-new-label***,***.product-grid3:hover* *.title*{opacity:*0*}

*.product-grid3* *.social* **li**{display:*inline-block*}

*.product-grid3* *.social* **li** **a**{color:*#e67e22*;background:*#fff*;font-size:*18px*;line-height:*50px*;width:*50px*;height:*50px*;border:*1px* *solid* *rgba(0,0,0,.1)*;border-radius:*50%*;margin:*0* *2px*;display:*block*;transition:*all* *.3s* *ease* *0s*}

*.product-grid3* *.social* **li** **a***:hover*{background:*#e67e22*;color:*#fff*}

*.product-grid3* *.product-discount-label***,***.product-grid3* *.product-new-label*{background-color:*#e67e22*;color:*#fff*;font-size:*17px*;padding:*2px* *10px*;position:*absolute*;right:*10px*;top:*10px*;transition:*all* *.3s*}

*.product-grid3* *.product-content*{z-index:*-1*;padding:*15px*;text-align:*left*}

*.product-grid3* *.title*{font-size:*14px*;text-transform:*capitalize*;margin:*0* *0* *7px*;transition:*all* *.3s* *ease* *0s*}

*.product-grid3* *.title* **a**{color:*#414141*}

*.product-grid3* *.price*{color:*#000*;font-size:*16px*;letter-spacing:*1px*;font-weight:*600*;margin-right:*2px*;display:*inline-block*}

*.product-grid3* *.price* **span**{color:*#909090*;font-size:*14px*;font-weight:*500*;letter-spacing:*0*;text-decoration:*line-through*;text-align:*left*;display:*inline-block*;margin-top:*-2px*}

*.product-grid3* *.rating*{padding:*0*;margin:*-22px* *0* *0*;list-style:*none*;text-align:*right*;display:*block*}

*.product-grid3* *.rating* **li**{color:*#ffd200*;font-size:*13px*;display:*inline-block*}

*.product-grid3* *.rating* **li***.disable*{color:*#dcdcdc*}

@media only screen and (max-width:1200px){*.product-grid3* *.rating*{margin:*0*}

}

@media only screen and (max-width:990px){*.product-grid3*{margin-bottom:*30px*}

*.product-grid3* *.rating*{margin:*-22px* *0* *0*}

}

@media only screen and (max-width:359px){*.product-grid3* *.rating*{margin:*0*}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Shopping Demo-4 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

*.product-grid4***,***.product-grid4* *.product-image4*{position:*relative*}

*.product-grid4*{font-family:*Poppins,sans-serif*;text-align:*center*;border-radius:*5px*;overflow:*hidden*;z-index:*1*;transition:*all* *.3s* *ease* *0s*}

*.product-grid4:hover*{box-shadow:*0* *0* *10px* *rgba(0,0,0,.2)*}

*.product-grid4* *.product-image4* **a**{display:*block*}

*.product-grid4* *.product-image4* **img**{width:*100%*;height:*auto*}

*.product-grid4* *.pic-1*{opacity:*1*;transition:*all* *.5s* *ease-out* *0s*}

*.product-grid4:hover* *.pic-1*{opacity:*0*}

*.product-grid4* *.pic-2*{position:*absolute*;top:*0*;left:*0*;opacity:*0*;transition:*all* *.5s* *ease-out* *0s*}

*.product-grid4:hover* *.pic-2*{opacity:*1*}

*.product-grid4* *.social*{width:*180px*;padding:*0*;margin:*0* *auto*;list-style:*none*;position:*absolute*;right:*0*;left:*0*;top:*50%*;transform:*translateY(-50%)*;transition:*all* *.3s* *ease* *0s*}

*.product-grid4* *.social* **li**{display:*inline-block*;opacity:*0*;transition:*all* *.7s*}

*.product-grid4* *.social* **li***:nth-child*(*1*){transition-delay:*.15s*}

*.product-grid4* *.social* **li***:nth-child*(*2*){transition-delay:*.3s*}

*.product-grid4* *.social* **li***:nth-child*(*3*){transition-delay:*.45s*}

*.product-grid4:hover* *.social* **li**{opacity:*1*}

*.product-grid4* *.social* **li** **a**{color:*#222*;background:*#fff*;font-size:*17px*;line-height:*36px*;width:*40px*;height:*36px*;border-radius:*2px*;margin:*0* *5px*;display:*block*;transition:*all* *.3s* *ease* *0s*}

*.product-grid4* *.social* **li** **a***:hover*{color:*#fff*;background:*#16a085*}

*.product-grid4* *.social* **li** **a***:after***,***.product-grid4* *.social* **li** **a***:before*{content:*attr(data-tip)*;color:*#fff*;background-color:*#000*;font-size:*12px*;line-height:*20px*;border-radius:*3px*;padding:*0* *5px*;white-space:*nowrap*;opacity:*0*;transform:*translateX(-50%)*;position:*absolute*;left:*50%*;top:*-30px*}

*.product-grid4* *.social* **li** **a***:after*{content:*''*;height:*15px*;width:*15px*;border-radius:*0*;transform:*translateX(-50%)* *rotate(45deg)*;top:*-22px*;z-index:*-1*}

*.product-grid4* *.social* **li** **a***:hover:after***,***.product-grid4* *.social* **li** **a***:hover:before*{opacity:*1*}

*.product-grid4* *.product-discount-label***,***.product-grid4* *.product-new-label*{color:*#fff*;background-color:*#16a085*;font-size:*13px*;font-weight:*800*;text-transform:*uppercase*;line-height:*45px*;height:*45px*;width:*45px*;border-radius:*50%*;position:*absolute*;left:*10px*;top:*15px*;transition:*all* *.3s*}

*.product-grid4* *.product-discount-label*{left:*auto*;right:*10px*;background-color:*#d7292a*}

*.product-grid4:hover* *.product-new-label*{opacity:*0*}

*.product-grid4* *.product-content*{padding:*25px*}

*.product-grid4* *.title*{font-size:*15px*;font-weight:*400*;text-transform:*capitalize*;margin:*0* *0* *7px*;transition:*all* *.3s* *ease* *0s*}

*.product-grid4* *.title* **a**{color:*#222*}

*.product-grid4* *.title* **a***:hover*{color:*#16a085*}

*.product-grid4* *.price*{color:*#16a085*;font-size:*17px*;font-weight:*700*;margin:*0* *2px* *15px* *0*;display:*block*}

*.product-grid4* *.price* **span**{color:*#909090*;font-size:*13px*;font-weight:*400*;letter-spacing:*0*;text-decoration:*line-through*;text-align:*left*;vertical-align:*middle*;display:*inline-block*}

*.product-grid4* *.add-to-cart*{border:*1px* *solid* *#e5e5e5*;display:*inline-block*;padding:*10px* *20px*;color:*#888*;font-weight:*600*;font-size:*14px*;border-radius:*4px*;transition:*all* *.3s*}

*.product-grid4:hover* *.add-to-cart*{border:*1px* *solid* *transparent*;background:*#16a085*;color:*#fff*}

*.product-grid4* *.add-to-cart:hover*{background-color:*#505050*;box-shadow:*0* *0* *10px* *rgba(0,0,0,.5)*}

@media only screen and (max-width:990px){*.product-grid4*{margin-bottom:*30px*}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Shopping Demo-5 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

*.product-image5* *.pic-1***,***.product-image5* *.pic-2*{backface-visibility:*hidden*;transition:*all* *.5s* *ease* *0s*}

*.product-grid5*{font-family:*Poppins,sans-serif*;position:*relative*}

*.product-grid5* *.product-image5*{overflow:*hidden*;position:*relative*}

*.product-grid5* *.product-image5* **a**{display:*block*}

*.product-grid5* *.product-image5* **img**{width:*100%*;height:*auto*}

*.product-image5* *.pic-1*{opacity:*1*}

*.product-grid5:hover* *.product-image5* *.pic-1*{opacity:*0*}

*.product-image5* *.pic-2*{width:*100%*;height:*100%*;opacity:*0*;position:*absolute*;top:*0*;left:*0*}

*.product-grid5:hover* *.product-image5* *.pic-2*{opacity:*1*}

*.product-grid5* *.social*{padding:*0*;margin:*0*;position:*absolute*;top:*10px*;right:*10px*}

*.product-grid5* *.social* **li**{display:*block*;margin:*0* *0* *10px*;transition:*all* *.5s*}

*.product-grid5* *.social* **li***:nth-child*(*2*){opacity:*0*;transform:*translateY(-50px)*}

*.product-grid5:hover* *.social* **li***:nth-child*(*2*){opacity:*1*;transform:*translateY(0)*}

*.product-grid5* *.social* **li***:nth-child*(*3*){opacity:*0*;transform:*translateY(-50px)*}

*.product-grid5:hover* *.social* **li***:nth-child*(*3*){opacity:*1*;transform:*translateY(0)*;transition-delay:*.2s*}

*.product-grid5* *.social* **li** **a**{color:*#888*;background:*#fff*;font-size:*14px*;text-align:*center*;line-height:*40px*;height:*40px*;width:*40px*;border-radius:*50%*;display:*block*;transition:*.5s* *ease* *0s*}

*.product-grid5* *.social* **li** **a***:hover*{color:*#fff*;background:*#1e3799*}

*.product-grid5* *.select-options*{color:*#777*;background-color:*#fff*;font-size:*13px*;font-weight:*400*;text-align:*center*;text-transform:*uppercase*;padding:*15px* *5px*;margin:*0* *auto*;opacity:*0*;display:*block*;position:*absolute*;width:*92%*;left:*0*;bottom:*-100px*;right:*0*;transition:*.5s* *ease* *0s*}

*.product-grid5* *.select-options:hover*{color:*#fff*;background-color:*#1e3799*;text-decoration:*none*}

*.product-grid5:hover* *.select-options*{opacity:*1*;bottom:*10px*}

*.product-grid5* *.product-content*{padding:*20px* *10px*}

*.product-grid5* *.title*{font-size:*15px*;font-weight:*600*;text-transform:*capitalize*;margin:*0* *0* *10px*;transition:*all* *.3s* *ease* *0s*}

*.product-grid5* *.title* **a**{color:*#222*}

*.product-grid5* *.title* **a***:hover*{color:*#1e3799*}

*.product-grid5* *.price*{color:*#222*;font-size:*13px*;font-weight:*500*;letter-spacing:*1px*}

@media only screen and (max-width:990px){*.product-grid5*{margin-bottom:*30px*}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Shopping Demo-6 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

*.product-grid6***,***.product-grid6* *.product-image6*{overflow:*hidden*}

*.product-grid6*{font-family:*'Open Sans',sans-serif*;text-align:*center*;position:*relative*;transition:*all* *.5s* *ease* *0s*}

*.product-grid6:hover*{box-shadow:*0* *0* *10px* *rgba(0,0,0,.3)*}

*.product-grid6* *.product-image6* **a**{display:*block*}

*.product-grid6* *.product-image6* **img**{width:*100%*;height:*auto*;transition:*all* *.5s* *ease* *0s*}

*.product-grid6:hover* *.product-image6* **img**{transform:*scale(1.1)*}

*.product-grid6* *.product-content*{padding:*12px* *12px* *15px*;transition:*all* *.5s* *ease* *0s*}

*.product-grid6:hover* *.product-content*{opacity:*0*}

*.product-grid6* *.title*{font-size:*20px*;font-weight:*600*;text-transform:*capitalize*;margin:*0* *0* *10px*;transition:*all* *.3s* *ease* *0s*}

*.product-grid6* *.title* **a**{color:*#000*}

*.product-grid6* *.title* **a***:hover*{color:*#2e86de*}

*.product-grid6* *.price*{font-size:*18px*;font-weight:*600*;color:*#2e86de*}

*.product-grid6* *.price* **span**{color:*#999*;font-size:*15px*;font-weight:*400*;text-decoration:*line-through*;margin-left:*7px*;display:*inline-block*}

*.product-grid6* *.social*{background-color:*#fff*;width:*100%*;padding:*0*;margin:*0*;list-style:*none*;opacity:*0*;transform:*translateX(-50%)*;position:*absolute*;bottom:*-50%*;left:*50%*;z-index:*1*;transition:*all* *.5s* *ease* *0s*}

*.product-grid6:hover* *.social*{opacity:*1*;bottom:*20px*}

*.product-grid6* *.social* **li**{display:*inline-block*}

*.product-grid6* *.social* **li** **a**{color:*#909090*;font-size:*16px*;line-height:*45px*;text-align:*center*;height:*45px*;width:*45px*;margin:*0* *7px*;border:*1px* *solid* *#909090*;border-radius:*50px*;display:*block*;position:*relative*;transition:*all* *.3s* *ease-in-out*}

*.product-grid6* *.social* **li** **a***:hover*{color:*#fff*;background-color:*#2e86de*;width:*80px*}

*.product-grid6* *.social* **li** **a***:after***,***.product-grid6* *.social* **li** **a***:before*{content:*attr(data-tip)*;color:*#fff*;background-color:*#2e86de*;font-size:*12px*;letter-spacing:*1px*;line-height:*20px*;padding:*1px* *5px*;border-radius:*5px*;white-space:*nowrap*;opacity:*0*;transform:*translateX(-50%)*;position:*absolute*;left:*50%*;top:*-30px*}

*.product-grid6* *.social* **li** **a***:after*{content:*''*;height:*15px*;width:*15px*;border-radius:*0*;transform:*translateX(-50%)* *rotate(45deg)*;top:*-20px*;z-index:*-1*}

*.product-grid6* *.social* **li** **a***:hover:after***,***.product-grid6* *.social* **li** **a***:hover:before*{opacity:*1*}

@media only screen and (max-width:990px){*.product-grid6*{margin-bottom:*30px*}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Shopping Demo-7 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

*.product-grid7*{font-family:*'Roboto Slab',serif*;position:*relative*;z-index:*1*}

*.product-grid7* *.product-image7*{border:*1px* *solid* *rgba(0,0,0,.1)*;overflow:*hidden*;perspective:*1500px*;position:*relative*;transition:*all* *.3s* *ease* *0s*}

*.product-grid7* *.product-image7* **a**{display:*block*}

*.product-grid7* *.product-image7* **img**{width:*100%*;height:*auto*}

*.product-grid7* *.pic-1*{opacity:*1*;transition:*all* *.5s* *ease-out* *0s*}

*.product-grid7* *.pic-2*{opacity:*0*;transform:*rotateY(-90deg)*;position:*absolute*;top:*0*;left:*0*;transition:*all* *.5s* *ease-out* *0s*}

*.product-grid7:hover* *.pic-2*{opacity:*1*;transform:*rotateY(0)*}

*.product-grid7* *.social*{padding:*0*;margin:*0*;list-style:*none*;position:*absolute*;bottom:*3px*;left:*-20%*;z-index:*1*;transition:*all* *.5s* *ease* *0s*}

*.product-grid7:hover* *.social*{left:*17px*}

*.product-grid7* *.social* **li** **a**{color:*#fff*;background-color:*#333*;font-size:*16px*;line-height:*40px*;text-align:*center*;height:*40px*;width:*40px*;margin:*15px* *0*;border-radius:*50%*;display:*block*;transition:*all* *.5s* *ease-in-out*}

*.product-grid7* *.social* **li** **a***:hover*{color:*#fff*;background-color:*#78e08f*}

*.product-grid7* *.product-new-label*{color:*#fff*;background-color:*#333*;padding:*5px* *10px*;border-radius:*5px*;display:*block*;position:*absolute*;top:*10px*;left:*10px*}

*.product-grid7* *.product-content*{text-align:*center*;padding:*20px* *0* *0*}

*.product-grid7* *.title*{font-size:*15px*;font-weight:*600*;text-transform:*capitalize*;margin:*0* *0* *10px*;transition:*all* *.3s* *ease* *0s*}

*.product-grid7* *.title* **a**{color:*#333*}

*.product-grid7* *.title* **a***:hover*{color:*#78e08f*}

*.product-grid7* *.rating*{color:*#78e08f*;font-size:*12px*;padding:*0*;margin:*0* *0* *10px*;list-style:*none*}

*.product-grid7* *.price*{color:*#333*;font-size:*20px*;font-family:*Lora,serif*;font-weight:*700*;margin-bottom:*8px*;text-align:*center*;transition:*all* *.3s*}

*.product-grid7* *.price* **span**{color:*#999*;font-size:*14px*;font-weight:*700*;text-decoration:*line-through*;margin-left:*7px*;display:*inline-block*}

@media only screen and (max-width:990px){*.product-grid7*{margin-bottom:*30px*}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Shopping Demo-8 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

*.product-grid8*{font-family:*Poppins,sans-serif*;position:*relative*;z-index:*1*}

*.product-grid8* *.product-image8*{border:*1px* *solid* *#e4e9ef*;position:*relative*;transition:*all* *.3s* *ease* *0s*}

*.product-grid8:hover* *.product-image8*{box-shadow:*0* *0* *10px* *rgba(0,0,0,.15)*}

*.product-grid8* *.product-image8* **a**{display:*block*}

*.product-grid8* *.product-image8* **img**{width:*100%*;height:*auto*}

*.product-grid8* *.pic-1*{opacity:*1*;transition:*all* *.5s* *ease-out* *0s*}

*.product-grid8:hover* *.pic-1*{opacity:*0*}

*.product-grid8* *.pic-2*{opacity:*0*;position:*absolute*;top:*0*;left:*0*;transition:*all* *.5s* *ease-out* *0s*}

*.product-grid8:hover* *.pic-2*{opacity:*1*}

*.product-grid8* *.social*{padding:*0*;margin:*0*;list-style:*none*;position:*absolute*;bottom:*13px*;right:*13px*;z-index:*1*}

*.product-grid8* *.social* **li**{opacity:*0*;transform:*translateY(3px)*;transition:*all* *.5s* *ease* *0s*}

*.product-grid8:hover* *.social* **li**{margin:*0* *0* *10px*;opacity:*1*;transform:*translateY(0)*}

*.product-grid8:hover* *.social* **li***:nth-child*(*1*){transition-delay:*.1s*}

*.product-grid8:hover* *.social* **li***:nth-child*(*2*){transition-delay:*.2s*}

*.product-grid8:hover* *.social* **li***:nth-child*(*3*){transition-delay:*.4s*}

*.product-grid8* *.social* **li** **a**{color:*grey*;font-size:*17px*;line-height:*40px*;text-align:*center*;height:*40px*;width:*40px*;border:*1px* *solid* *grey*;display:*block*;transition:*all* *.5s* *ease-in-out*}

*.product-grid8* *.social* **li** **a***:hover*{color:*#000*;border-color:*#000*}

*.product-grid8* *.product-discount-label*{display:*block*;padding:*4px* *15px* *4px* *30px*;color:*#fff*;background-color:*#0081c2*;position:*absolute*;top:*10px*;right:*0*;-webkit-clip-path:*polygon(34%* *0,100%* *0,100%* *100%,0* *100%)*;clip-path:*polygon(34%* *0,100%* *0,100%* *100%,0* *100%)*}

*.product-grid8* *.product-content*{padding:*20px* *0* *0*}

*.product-grid8* *.price*{color:*#000*;font-size:*19px*;font-weight:*400*;margin-bottom:*8px*;text-align:*left*;transition:*all* *.3s*}

*.product-grid8* *.price* **span**{color:*#999*;font-size:*14px*;font-weight:*500*;text-decoration:*line-through*;margin-left:*7px*;display:*inline-block*}

*.product-grid8* *.product-shipping*{color:*rgba(0,0,0,.5)*;font-size:*15px*;padding-left:*35px*;margin:*0* *0* *15px*;display:*block*;position:*relative*}

*.product-grid8* *.product-shipping:before*{content:*''*;height:*1px*;width:*25px*;background-color:*rgba(0,0,0,.5)*;transform:*translateY(-50%)*;position:*absolute*;top:*50%*;left:*0*}

*.product-grid8* *.title*{font-size:*16px*;font-weight:*400*;text-transform:*capitalize*;margin:*0* *0* *30px*;transition:*all* *.3s* *ease* *0s*}

*.product-grid8* *.title* **a**{color:*#000*}

*.product-grid8* *.title* **a***:hover*{color:*#0081c2*}

*.product-grid8* *.all-deals*{display:*block*;color:*#fff*;background-color:*#2e353b*;font-size:*15px*;letter-spacing:*1px*;text-align:*center*;text-transform:*uppercase*;padding:*22px* *5px*;transition:*all* *.5s* *ease* *0s*}

*.product-grid8* *.all-deals* *.icon*{margin-left:*7px*}

*.product-grid8* *.all-deals:hover*{background-color:*#0081c2*}

@media only screen and (max-width:990px){*.product-grid8*{margin-bottom:*30px*}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Shopping Demo-9 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

*.product-grid9***,***.product-grid9* *.product-image9*{position:*relative*}

*.product-grid9*{font-family:*Poppins,sans-serif*;z-index:*1*}

*.product-grid9* *.product-image9* **a**{display:*block*}

*.product-grid9* *.product-image9* **img**{width:*100%*;height:*auto*}

*.product-grid9* *.pic-1*{opacity:*1*;transition:*all* *.5s* *ease-out* *0s*}

*.product-grid9:hover* *.pic-1*{opacity:*0*}

*.product-grid9* *.pic-2*{position:*absolute*;top:*0*;left:*0*;opacity:*0*;transition:*all* *.5s* *ease-out* *0s*}

*.product-grid9:hover* *.pic-2*{opacity:*1*}

*.product-grid9* *.product-full-view*{color:*#505050*;background-color:*#fff*;font-size:*16px*;height:*45px*;width:*45px*;padding:*18px*;border-radius:*100px* *0* *0*;display:*block*;opacity:*0*;position:*absolute*;right:*0*;bottom:*0*;transition:*all* *.3s* *ease* *0s*}

*.product-grid9* *.product-full-view:hover*{color:*#c0392b*}

*.product-grid9:hover* *.product-full-view*{opacity:*1*}

*.product-grid9* *.product-content*{padding:*12px* *12px* *0*;overflow:*hidden*;position:*relative*}

*.product-content* *.rating*{padding:*0*;margin:*0* *0* *7px*;list-style:*none*}

*.product-grid9* *.rating* **li**{font-size:*12px*;color:*#ffd200*;transition:*all* *.3s* *ease* *0s*}

*.product-grid9* *.rating* **li***.disable*{color:*rgba(0,0,0,.2)*}

*.product-grid9* *.title*{font-size:*16px*;font-weight:*400*;text-transform:*capitalize*;margin:*0* *0* *3px*;transition:*all* *.3s* *ease* *0s*}

*.product-grid9* *.title* **a**{color:*rgba(0,0,0,.5)*}

*.product-grid9* *.title* **a***:hover*{color:*#c0392b*}

*.product-grid9* *.price*{color:*#000*;font-size:*17px*;margin:*0*;display:*block*;transition:*all* *.5s* *ease* *0s*}

*.product-grid9:hover* *.price*{opacity:*0*}

*.product-grid9* *.add-to-cart*{display:*block*;color:*#c0392b*;font-weight:*600*;font-size:*14px*;opacity:*0*;position:*absolute*;left:*10px*;bottom:*-20px*;transition:*all* *.5s* *ease* *0s*}

*.product-grid9:hover* *.add-to-cart*{opacity:*1*;bottom:*0*}

@media only screen and (max-width:990px){*.product-grid9*{margin-bottom:*30px*}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*globals\*\*\*\*\*\*\*\*\*\*\*\*\*/

**body** {

font-family: *'open sans'*;

overflow-x: *hidden*; }

**img** {

max-width: *100%*; }

*.preview* {

display: *-webkit-box*;

display: *-webkit-flex*;

display: *-ms-flexbox*;

display: *flex*;

-webkit-box-orient: *vertical*;

-webkit-box-direction: *normal*;

-webkit-flex-direction: *column*;

-ms-flex-direction: *column*;

flex-direction: *column*; }

@media screen and (max-width: 996px) {

*.preview* {

margin-bottom: *20px*; } }

*.preview-pic* {

-webkit-box-flex: *1*;

-webkit-flex-grow: *1*;

-ms-flex-positive: *1*;

flex-grow: *1*; }

*.preview-thumbnail.nav-tabs* {

border: *none*;

margin-top: *15px*; }

*.preview-thumbnail.nav-tabs* **li** {

width: *18%*;

margin-right: *2.5%*; }

*.preview-thumbnail.nav-tabs* **li** **img** {

max-width: *100%*;

display: *block*; }

*.preview-thumbnail.nav-tabs* **li** **a** {

padding: *0*;

margin: *0*; }

*.preview-thumbnail.nav-tabs* **li***:last-of-type* {

margin-right: *0*; }

*.tab-content* {

overflow: *hidden*; }

*.tab-content* **img** {

width: *100%*;

-webkit-animation-name: *opacity*;

animation-name: *opacity*;

-webkit-animation-duration: *.3s*;

animation-duration: *.3s*; }

*.card* {

margin-top: *50px*;

background: *#eee*;

padding: *3em*;

line-height: *1.5em*; }

@media screen and (min-width: 997px) {

*.wrapper* {

display: *-webkit-box*;

display: *-webkit-flex*;

display: *-ms-flexbox*;

display: *flex*; } }

*.details* {

display: *-webkit-box*;

display: *-webkit-flex*;

display: *-ms-flexbox*;

display: *flex*;

-webkit-box-orient: *vertical*;

-webkit-box-direction: *normal*;

-webkit-flex-direction: *column*;

-ms-flex-direction: *column*;

flex-direction: *column*; }

*.colors* {

-webkit-box-flex: *1*;

-webkit-flex-grow: *1*;

-ms-flex-positive: *1*;

flex-grow: *1*; }

*.product-title***,** *.price***,** *.sizes***,** *.colors* {

text-transform: *UPPERCASE*;

font-weight: *bold*; }

*.checked***,** *.price* **span** {

color: *#ff9f1a*; }

*.product-title***,** *.rating***,** *.product-description***,** *.price***,** *.vote***,** *.sizes* {

margin-bottom: *15px*; }

*.product-title* {

margin-top: *0*; }

*.size* {

margin-right: *10px*; }

*.size:first-of-type* {

margin-left: *40px*; }

*.color* {

display: *inline-block*;

vertical-align: *middle*;

margin-right: *10px*;

height: *2em*;

width: *2em*;

border-radius: *2px*; }

*.color:first-of-type* {

margin-left: *20px*; }

*.add-to-cart***,** *.like* {

background: *#ff9f1a*;

padding: *1.2em* *1.5em*;

border: *none*;

text-transform: *UPPERCASE*;

font-weight: *bold*;

color: *#fff*;

-webkit-transition: *background* *.3s* *ease*;

transition: *background* *.3s* *ease*; }

*.add-to-cart:hover***,** *.like:hover* {

background: *#b36800*;

color: *#fff*; }

*.not-available* {

text-align: *center*;

line-height: *2em*; }

*.not-available:before* {

font-family: *fontawesome*;

content: *"\f00d"*;

color: *#fff*; }

*.orange* {

background: *#ff9f1a*; }

*.green* {

background: *#85ad00*; }

*.blue* {

background: *#0076ad*; }

*.tooltip-inner* {

padding: *1.3em*; }

@**-webkit-keyframes** **opacity** {

0% {

opacity: *0*;

-webkit-transform: *scale(3)*;

transform: *scale(3)*; }

*100*% {

opacity: *1*;

-webkit-transform: *scale(1)*;

transform: *scale(1)*; } }

@**keyframes** **opacity** {

0% {

opacity: *0*;

-webkit-transform: *scale(3)*;

transform: *scale(3)*; }

*100*% {

opacity: *1*;

-webkit-transform: *scale(1)*;

transform: *scale(1)*; } }

/\*# sourceMappingURL=style.css.map \*/

<%@page import=*"com.online.garments.deal.util.PropertyReader"*%>

<%@page import=*"com.online.garments.deal.bean.CustomerBean"*%>

<%@page import=*"com.online.garments.deal.model.CustomerModel"*%>

<%@page import=*"com.online.garments.deal.bean.ProductBean"*%>

<%@page import=*"com.online.garments.deal.model.ProductModel"*%>

<%@page import=*"java.util.Iterator"*%>

<%@page import=*"java.util.List"*%>

<%@page import=*"com.online.garments.deal.bean.PaymentBean"*%>

<%@page import=*"com.online.garments.deal.util.ServletUtility"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Book List</title>

</head>

<body>

<%@ include file=*"Header.jsp"* %>

<br>

<nav aria-label=*"breadcrumb"* role=*"navigation"*> <nav

aria-label=*"breadcrumb"* role=*"navigation"*>

<ol class=*"breadcrumb"*>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.WELCOME\_CTL%>*"*>Home</a></li>

<li class=*"breadcrumb-item active"* aria-current=*"page"*>Book List</li>

</ol>

</nav>

<center>

<b><font color=*"red"*><%=ServletUtility.getErrorMessage(request)%></font></b>

<b><font color=*"green"*><%=ServletUtility.getSuccessMessage(request)%></font></b>

</center>

<hr>

<form action=*"*<%=OGDView.BOOK\_LIST\_CTL%>*"* method=*"post"*>

<div class=*"container"*>

<h3 class=*"h3"*>Book List</h3>

<div class=*"row"*>

<%

**int** pageNo = ServletUtility.getPageNo(request);

**int** pageSize = ServletUtility.getPageSize(request);

**int** index = ((pageNo - 1) \* pageSize) + 1;

PaymentBean bean = **null**;

List list = ServletUtility.getList(request);

Iterator<PaymentBean> it = list.iterator();

**while** (it.hasNext()) {

bean = it.next();

ProductModel pModel=**new** ProductModel();

ProductBean pBean=pModel.findByPk(bean.getProductId());

CustomerModel cModel=**new** CustomerModel();

CustomerBean cBean=cModel.findByPk(bean.getCustomerId());

%>

<div class=*"col-md-3 col-sm-6"*>

<div class=*"product-grid4"*>

<div class=*"product-image4"*>

<a href=*"#"*>

<img class=*"pic-1"* src=*"*<%=OGDView.APP\_CONTEXT%>*/image/*<%=pBean.getImage()%>*"*>

<img class=*"pic-2"* src=*"*<%=OGDView.APP\_CONTEXT%>*/image/*<%=pBean.getImage()%>*"*>

</a>

</div>

<div class=*"product-content"*>

<h3 class=*"title"*><a href=*"#"*><%=pBean.getProductName()%></a></h3>

<div class=*"price"*>

<%=PropertyReader.getValue("cur")%><%=bean.getAmount()%>

<!-- <span>$16.00</span> -->

</div>

</div>

</div>

</div>

<%} %>

</div>

</div>

<input type=*"hidden"* name=*"pageNo"* value=*"*<%=pageNo%>*"*> <input

type=*"hidden"* name=*"pageSize"* value=*"*<%=pageSize%>*"*>

</form>

<hr>

<%@ include file=*"Footer.jsp"* %>

</body>

</html><%@page import=*"com.online.garments.deal.model.CustomerModel"*%>

<%@page import=*"com.online.garments.deal.controller.CustomerListCtl"*%>

<%@page import=*"com.online.garments.deal.controller.CustomerCtl"*%>

<%@page import=*"java.util.Iterator"*%>

<%@page import=*"java.util.List"*%>

<%@page import=*"com.online.garments.deal.bean.CustomerBean"*%>

<%@page import=*"com.online.garments.deal.util.ServletUtility"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>CustomerList</title>

</head>

<body>

<%@ include file=*"Header.jsp"* %>

<br>

<nav aria-label=*"breadcrumb"* role=*"navigation"*> <nav

aria-label=*"breadcrumb"* role=*"navigation"*>

<ol class=*"breadcrumb"*>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.WELCOME\_CTL%>*"*>Home</a></li>

<li class=*"breadcrumb-item active"* aria-current=*"page"*>Customer</li>

</ol>

</nav>

</nav>

<center>

<h3>Customer</h3>

<b><font color=*"red"*><%=ServletUtility.getErrorMessage(request)%></font></b>

<b><font color=*"green"*><%=ServletUtility.getSuccessMessage(request)%></font></b>

</center>

<form action=*"*<%=OGDView.CUSTOMER\_LIST\_CTL%>*"* method=*"post"*>

<table class=*"table"*>

<thead class=*"thead-dark"*>

<tr>

<th scope=*"col"*>S No.</th>

<th scope=*"col"*>Name</th>

<th scope=*"col"*>Item Code</th>

<th scope=*"col"*>Contact No</th>

<th scope=*"col"*>Age</th>

<th scope=*"col"*>Address</th>

</tr>

</thead>

<tbody>

<%

**int** pageNo = ServletUtility.getPageNo(request);

**int** pageSize = ServletUtility.getPageSize(request);

**int** index = ((pageNo - 1) \* pageSize) + 1;

CustomerBean bean = **null**;

List list = ServletUtility.getList(request);

Iterator<CustomerBean> iterator = list.iterator();

**while** (iterator.hasNext()) {

bean = iterator.next();

%>

<tr>

<th scope=*"row"*><%=index++%></th>

<td><%=bean.getName()%></td>

<td><%=bean.getItemCode() %></td>

<td><%=bean.getContectNo() %></td>

<td><%=bean.getAge()%></td>

<td><%=bean.getAddress() %></td>

</tr>

<%} %>

</tbody>

</table>

<table >

<thead >

<tr>

<th ><input type=*"submit"* name=*"operation"* class=*"btn btn-primary pull-center"* value=*"*<%=CustomerListCtl.OP\_PREVIOUS%>*"* <%=(pageNo == 1) ? "disabled" : ""%>></th>

<%

CustomerModel model = **new** CustomerModel();

%>

<th ><input type=*"submit"* name=*"operation"* class=*"btn btn-primary pull-center"* value=*"*<%=CustomerListCtl.OP\_NEXT%>*"*

<%=((list.size() < pageSize) || model.nextPK() - 1 == bean.getId()) ? "disabled" : ""%> ></th>

</tr>

</thead>

</table>

<input type=*"hidden"* name=*"pageNo"* value=*"*<%=pageNo%>*"*> <input

type=*"hidden"* name=*"pageSize"* value=*"*<%=pageSize%>*"*>

</form>

<br>

<div style="margin-top: *193px*">

<%@ include file=*"Footer.jsp"* %>

</div>

</body>

</html><%@page import=*"com.online.garments.deal.bean.ProductBean"*%>

<%@page import=*"com.online.garments.deal.model.ProductModel"*%>

<%@page import=*"com.online.garments.deal.controller.CustomerCtl"*%>

<%@page import=*"com.online.garments.deal.util.ServletUtility"*%>

<%@page import=*"com.online.garments.deal.util.DataUtility"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Customer</title>

</head>

<body>

<%@ include file=*"Header.jsp"* %>

<br>

<nav aria-label=*"breadcrumb"* role=*"navigation"*> <nav

aria-label=*"breadcrumb"* role=*"navigation"*>

<ol class=*"breadcrumb"*>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.WELCOME\_CTL%>*"*>Home</a></li>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.PRODUCT\_LIST\_CTL%>*"*>Products</a></li>

<li class=*"breadcrumb-item active"* aria-current=*"page"*>Customer</li>

</ol>

</nav>

<div class=*"container"*>

<div class=*"card"*>

<div class=*"container-fliud"*>

<div class=*"wrapper row"*>

<div class=*"preview col-md-6"*>

<% Long pid=(Long)session.getAttribute("pid");

ProductModel pModel=**new** ProductModel();

ProductBean pBean=pModel.findByPk(pid);

%>

<div class=*"preview-pic tab-content"*>

<div class=*"tab-pane active"* id=*"pic-1"*><img src=*"*<%=OGDView.APP\_CONTEXT%>*/image/*<%=pBean.getImage()%>*"* /></div>

</div>

</div>

<div class=*"details col-md-6"*>

<h3 class=*"product-title"*><%=pBean.getProductName() %></h3>

<div class=*"rating"*>

<div class=*"stars"*>

<span class=*"fa fa-star checked"*></span>

<span class=*"fa fa-star checked"*></span>

<span class=*"fa fa-star checked"*></span>

<span class=*"fa fa-star"*></span>

<span class=*"fa fa-star"*></span>

</div>

<span class=*"review-no"*>41 reviews</span>

</div>

<h4 class=*"price"*>current price: <span><%=pBean.getPrice()%></span></h4>

</div>

</div>

</div>

</div>

</div>

<div col-md-5 img-thumbnail">

<div id=*"feedback"*> <div class=*"container"*>

<div class=*"col-md-5"*>

<div class=*"form-area"*>

<form role=*"form"* action=*"*<%=OGDView.CUSTOMER\_CTL%>*"* method=*"post"* >

<jsp:useBean id=*"bean"* class=*"com.online.garments.deal.bean.CustomerBean"*

scope=*"request"*></jsp:useBean>

<input type=*"hidden"* name=*"id"* value=*"*<%=bean.getId()%>*"*>

<input type=*"hidden"* name=*"createdBy"* value=*"*<%=bean.getCreatedBy()%>*"*>

<input type=*"hidden"* name=*"modifiedBy"* value=*"*<%=bean.getModifiedBy()%>*"*>

<input type=*"hidden"* name=*"createdDatetime"* value=*"*<%=DataUtility.getTimestamp(bean.getCreatedDatetime())%>*"*>

<input type=*"hidden"* name=*"modifiedDatetime"* value=*"*<%=DataUtility.getTimestamp(bean.getModifiedDatetime())%>*"*>

<br style="clear:*both*">

<h3 style="margin-bottom: *15px*; text-align: *left*: ;">Add Detail</h3>

<b><font color=*"red"*> <%=ServletUtility.getErrorMessage(request)%>

</font></b>

<b><font color=*"Green"*> <%=ServletUtility.getSuccessMessage(request)%>

</font></b>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"name"*

placeholder=*"Name"* value=*"*<%=DataUtility.getStringData(bean.getName())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("name", request)%></font>

</div>

<%**if**(!userLoggedIn){%>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"login"*

placeholder=*"Login Id"* value=*"*<%=DataUtility.getStringData(bean.getLogin())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("name", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"password"* class=*"form-control"* name=*"password"*

placeholder=*"Password"* value=*"*<%=DataUtility.getStringData(bean.getPassword())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("password", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"password"* class=*"form-control"* name=*"confirmPassword"*

placeholder=*"Confirm Password"* value=*"*<%=DataUtility.getStringData(bean.getConfirmPassword())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("confirmPassword", request)%></font>

</div>

<%} %>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"age"*

placeholder=*"Age"* value=*"*<%=DataUtility.getStringData(bean.getAge())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("age", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"contact"*

placeholder=*"Contact No"* value=*"*<%=DataUtility.getStringData(bean.getContectNo())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("contact", request)%></font>

</div>

<div class=*"form-group"*>

<textarea class=*"form-control"* name=*"address"* placeholder=*"Address"* rows=*"4"*><%=DataUtility.getStringData(bean.getAddress()) %></textarea>

<font color=*"red"*><%=ServletUtility.getErrorMessage("address", request)%></font>

</div>

<input type=*"submit"* name=*"operation"*

class=*"btn btn-primary pull-right"* value=*"*<%=CustomerCtl.OP\_SAVE%>*"*>

<input type=*"submit"* name=*"operation"*

class=*"btn btn-primary pull-right"* value=*"*<%=CustomerCtl.OP\_RESET%>*"*>

</form>

</div>

</div>

</div> </div>

<!--feedback-->

<br>

<div style="margin-top: *83px*">

<%@ include file=*"Footer.jsp"*%>

</div>

</body>

</html><%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Insert title here</title>

<style type=*"text/css"*>

**a***:hover*{

text-decoration:*none*;

}

*.section-padding* {

padding: *14px 0*;

}

*.bottom* {

background-color: *#0a1c2e*;

}

*.bottom* *.copyright* {

color: *#e5e5e5*;

font-weight: *600*;

}

*.copyright* **a** {

color: *#f2ff49*;

margin-left: *3px*;

padding-right: *3px*;

}

*.bottom* **p** {

margin-bottom: *0*;

line-height: *50px*;

font-size: *16px*;

font-weight: *400*;

}

*.copyright* **p** **span** {

color: *#d1caca*;

}

*.bottom* *.copyright* **p,** *.bottom* *.copyright* **a***:hover*{

color: *#6c6d83*;

}

</style>

</head>

<body>

<div class=*"bottom section-padding"*>

<div class=*"container"*>

<div class=*"row"*>

<div class=*"col-md-12 text-center"*>

<div class=*"copyright"*>

<p>© <span>2021</span> <a href=*"#"* class=*"transition"*>Sporty Shoes Online</a> Developed by @Mohanish for Simplylearn</p>

</div>

</div>

</div>

</div>

</div>

</body>

</html><%@page import=*"com.online.garments.deal.controller.LoginCtl"*%>

<%@page import=*"com.online.garments.deal.controller.OGDView"*%>

<%@page import=*"com.online.garments.deal.bean.LoginBean"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Insert title here</title>

<link

href=*"//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"*

rel=*"stylesheet"* id=*"bootstrap-css"*>

<script

src=*"//maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"*></script>

<script src=*"//code.jquery.com/jquery-1.11.1.min.js"*></script>

<!-- <link href="/ONLINE-GARMENTS-DEAL/css/main.css" rel="stylesheet"> -->

<!-- <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css" /> -->

<link href=*"/ONLINE-GARMENTS-DEAL/css/product.css"* rel=*"stylesheet"*>

</head>

<body>

<%

LoginBean userBean = (LoginBean) session.getAttribute("user");

**boolean** userLoggedIn = userBean != **null**;

String welcomeMsg = "Hi, ";

**if** (userLoggedIn) {

welcomeMsg += userBean.getLogin();

} **else** {

welcomeMsg += "Guest";

}

%>

<nav class=*"navbar navbar-expand-lg navbar-dark bg-dark"*> <a

class=*"navbar-brand"* href=*"#"*>Sporty Shoes Online</a>

<button class=*"navbar-toggler"* type=*"button"* data-toggle=*"collapse"*

data-target=*"#navbarSupportedContent"*

aria-controls=*"navbarSupportedContent"* aria-expanded=*"false"*

aria-label=*"Toggle navigation"*>

<span class=*"navbar-toggler-icon"*></span>

</button>

<div class=*"collapse navbar-collapse"* id=*"navbarSupportedContent"*>

<ul class=*"navbar-nav mr-auto"*>

<li class=*"nav-item active"*><a class=*"nav-link"*

href=*"*<%=OGDView.WELCOME\_CTL%>*"*>Home <span class=*"sr-only"*>(current)</span></a>

</li>

<%

**if** (userLoggedIn) {

%>

<%

**if** (userBean.getRole() == 1) {

%>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.MANAGER\_CTL%>*"*>Add Manager</a></li>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.PRODUCT\_CTL%>*"*>Add Product</a></li>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.PRODUCT\_LIST\_CTL%>*"*>Products</a></li>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.CUSTOMER\_LIST\_CTL%>*"*>Customer</a></li>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.PAYMENT\_LIST\_CTL%>*"*>Payments</a></li>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.BOOK\_LIST\_CTL%>*"*>Book List</a></li>

<%

} **else** **if** (userBean.getRole() == 2) {

%>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.PRODUCT\_CTL%>*"*>Add Product</a></li>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.PRODUCT\_LIST\_CTL%>*"*>Products</a></li>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.CUSTOMER\_LIST\_CTL%>*"*>Customer</a></li>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.PAYMENT\_LIST\_CTL%>*"*>Payments</a></li>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.BOOK\_LIST\_CTL%>*"*>Book List</a></li>

<%

} **else** **if** (userBean.getRole() == 3) {

%>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.PRODUCT\_LIST\_CTL%>*"*>Products</a></li>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.BOOK\_LIST\_CTL%>*"*>Book List</a></li>

<%

%>

<%

}} **else** {

%>

<li class=*"nav-item"*><a class=*"nav-link"*

href=*"*<%=OGDView.PRODUCT\_LIST\_CTL%>*"*>Products</a></li>

<!-- <li class="nav-item"><a class="nav-link" href="#">Contact Us</a></li>

<li class="nav-item"><a class="nav-link" href="#">About Us</a></li> -->

<%

}

%>

</ul>

<form class=*"form-inline my-2 my-lg-0"*>

<%

**if** (userLoggedIn) {

%>

<a class=*"nav-link"* class=*"btn btn-outline-success my-2 my-sm-0"*

href=*"*<%=OGDView.LOGIN\_CTL%>*?operation=*<%=LoginCtl.OP\_LOG\_OUT%>*"*>Logout</a>

<%

} **else** {

%>

<a class=*"nav-link"* class=*"btn btn-outline-success my-2 my-sm-0"*

href=*"*<%=OGDView.LOGIN\_CTL%>*"*>Login</a>

<%

}

%>

</form>

</div>

</nav>

</body>

</html><%@page import=*"com.online.garments.deal.controller.LoginCtl"*%>

<%@page import=*"com.online.garments.deal.util.ServletUtility"*%>

<%@page import=*"com.online.garments.deal.util.DataUtility"*%>

<%@page import=*"com.online.garments.deal.controller.OGDView"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Login</title>

</head>

<body>

<%@ include file=*"Header.jsp"*%>

<br>

<nav aria-label=*"breadcrumb"* role=*"navigation"*> <nav

aria-label=*"breadcrumb"* role=*"navigation"*>

<ol class=*"breadcrumb"*>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.WELCOME\_CTL%>*"*>Home</a></li>

<li class=*"breadcrumb-item active"* aria-current=*"page"*>Login</li>

</ol>

</nav>

<br>

<div col-md-5img-thumbnail">

<div id=*"feedback"*>

<div class=*"container"*>

<div class=*"col-md-5"*>

<div class=*"form-area"*>

<form role=*"form"* action=*"*<%=OGDView.LOGIN\_CTL%>*"* method=*"post"* >

<br style="clear: *both*">

<jsp:useBean id=*"bean"* class=*"com.online.garments.deal.bean.LoginBean"*

scope=*"request"*></jsp:useBean>

<% String uri=(String)request.getAttribute("uri");%>

<input type=*"hidden"* name=*"uri"* value=*"*<%=uri%>*"*>

<input type=*"hidden"* name=*"id"* value=*"*<%=bean.getId()%>*"*>

<input type=*"hidden"* name=*"createdBy"* value=*"*<%=bean.getCreatedBy()%>*"*>

<input type=*"hidden"* name=*"modifiedBy"* value=*"*<%=bean.getModifiedBy()%>*"*>

<input type=*"hidden"* name=*"createdDatetime"* value=*"*<%=DataUtility.getTimestamp(bean.getCreatedDatetime())%>*"*>

<input type=*"hidden"* name=*"modifiedDatetime"* value=*"*<%=DataUtility.getTimestamp(bean.getModifiedDatetime())%>*"*>

<h3 style="margin-bottom: *15px*; text-align: *left*: ;">Login</h3>

<b><font color=*"red"*> <%=ServletUtility.getErrorMessage(request)%>

</font></b>

<b><font color=*"Green"*> <%=ServletUtility.getSuccessMessage(request)%>

</font></b>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"login"*

placeholder=*"Login Id"* value=*"*<%=DataUtility.getStringData(bean.getLogin())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("login", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"password"* class=*"form-control"*

name=*"password"* placeholder=*"Password"* value=*"*<%=DataUtility.getStringData(bean.getPassword()) %>*"* >

<font color=*"red"*> <%=ServletUtility.getErrorMessage("password", request)%></font>

</div>

<input type=*"submit"* name=*"operation"*

class=*"btn btn-primary pull-right"* value=*"*<%=LoginCtl.OP\_SIGN\_IN %>*"*>

<%-- <input type="submit" name="operation"

class="btn btn-primary pull-right" value="<%=LoginCtl.OP\_SIGN\_UP %>"> --%>

<%-- <a href="<%=OGDView.FORGET\_PASSWORD\_CTL%>"><b>Forget my password</b></a> --%>

</form>

</div>

</div>

</div>

</div>

<!--feedback-->

<br>

<div style="margin-top: *147px*">

<%@ include file=*"Footer.jsp"*%>

</div>

</body>

</html><%@page import=*"com.online.garments.deal.controller.ManagerCtl"*%>

<%@page import=*"com.online.garments.deal.util.HTMLUtility"*%>

<%@page import=*"java.util.HashMap"*%>

<%@page import=*"com.online.garments.deal.util.ServletUtility"*%>

<%@page import=*"com.online.garments.deal.util.DataUtility"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Manager</title>

</head>

<body>

<%@ include file=*"Header.jsp"* %>

<br>

<nav aria-label=*"breadcrumb"* role=*"navigation"*> <nav

aria-label=*"breadcrumb"* role=*"navigation"*>

<ol class=*"breadcrumb"*>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.WELCOME\_CTL%>*"*>Home</a></li>

<li class=*"breadcrumb-item active"* aria-current=*"page"*>Manager</li>

</ol>

</nav>

<div col-md-5 img-thumbnail">

<div id=*"feedback"*> <div class=*"container"*>

<div class=*"col-md-5"*>

<div class=*"form-area"*>

<form role=*"form"* action=*"*<%=OGDView.MANAGER\_CTL%>*"* method=*"post"* >

<jsp:useBean id=*"bean"* class=*"com.online.garments.deal.bean.ManagerBean"*

scope=*"request"*></jsp:useBean>

<input type=*"hidden"* name=*"id"* value=*"*<%=bean.getId()%>*"*>

<input type=*"hidden"* name=*"createdBy"* value=*"*<%=bean.getCreatedBy()%>*"*>

<input type=*"hidden"* name=*"modifiedBy"* value=*"*<%=bean.getModifiedBy()%>*"*>

<input type=*"hidden"* name=*"createdDatetime"* value=*"*<%=DataUtility.getTimestamp(bean.getCreatedDatetime())%>*"*>

<input type=*"hidden"* name=*"modifiedDatetime"* value=*"*<%=DataUtility.getTimestamp(bean.getModifiedDatetime())%>*"*>

<br style="clear:*both*">

<h3 style="margin-bottom: *15px*; text-align: *left*: ;">Manager</h3>

<b><font color=*"red"*> <%=ServletUtility.getErrorMessage(request)%>

</font></b>

<b><font color=*"Green"*> <%=ServletUtility.getSuccessMessage(request)%>

</font></b>

<%

HashMap map = **new** HashMap();

map.put("1", "Admin");

map.put("2", "Manager");

%>

<div class=*"form-group"*>

<%=HTMLUtility.getList("role", String.valueOf(bean.getRole()), map)%>

<font color=*"red"*><%=ServletUtility.getErrorMessage("role", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"name"*

placeholder=*"Name"* value=*"*<%=DataUtility.getStringData(bean.getName())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("name", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"login"*

placeholder=*"Login Id"* value=*"*<%=DataUtility.getStringData(bean.getLogin())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("login", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"password"* class=*"form-control"*

name=*"password"* placeholder=*"Password"* value=*"*<%=DataUtility.getStringData(bean.getPassword()) %>*"* >

<font color=*"red"*> <%=ServletUtility.getErrorMessage("password", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"password"* class=*"form-control"*

name=*"confirmPassword"* placeholder=*"Confirm Password"* value=*"*<%=DataUtility.getStringData(bean.getConfirmPassword()) %>*"* >

<font color=*"red"*> <%=ServletUtility.getErrorMessage("confirmPassword", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"email"*

placeholder=*"Email Id"* value=*"*<%=DataUtility.getStringData(bean.getEmailId())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("email", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"contact"*

placeholder=*"Contact No"* value=*"*<%=DataUtility.getStringData(bean.getContactNo())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("contact", request)%></font>

</div>

<div class=*"form-group"*>

<textarea class=*"form-control"* name=*"address"* placeholder=*"Address"* rows=*"4"*><%=DataUtility.getStringData(bean.getAddress()) %></textarea>

<font color=*"red"*><%=ServletUtility.getErrorMessage("address", request)%></font>

</div>

<input type=*"submit"* name=*"operation"*

class=*"btn btn-primary pull-right"* value=*"*<%=ManagerCtl.OP\_SAVE%>*"*>

<input type=*"submit"* name=*"operation"*

class=*"btn btn-primary pull-right"* value=*"*<%=ManagerCtl.OP\_RESET%>*"*>

</form>

</div>

</div>

</div> </div> <!--feedback-->

<br>

<%@ include file=*"Footer.jsp"*%>

</body>

</html><%@page import=*"com.online.garments.deal.model.CustomerModel"*%>

<%@page import=*"com.online.garments.deal.bean.ProductBean"*%>

<%@page import=*"com.online.garments.deal.model.ProductModel"*%>

<%@page import=*"com.online.garments.deal.util.DataUtility"*%>

<%@page import=*"com.online.garments.deal.controller.PaymentListCtl"*%>

<%@page import=*"com.online.garments.deal.model.PaymentModel"*%>

<%@page import=*"com.online.garments.deal.bean.PaymentBean"*%>

<%@page import=*"java.util.Iterator"*%>

<%@page import=*"java.util.List"*%>

<%@page import=*"com.online.garments.deal.bean.CustomerBean"*%>

<%@page import=*"com.online.garments.deal.util.ServletUtility"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Payment List</title>

</head>

<body>

<%@ include file=*"Header.jsp"* %>

<br>

<nav aria-label=*"breadcrumb"* role=*"navigation"*> <nav

aria-label=*"breadcrumb"* role=*"navigation"*>

<ol class=*"breadcrumb"*>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.WELCOME\_CTL%>*"*>Home</a></li>

<li class=*"breadcrumb-item active"* aria-current=*"page"*>Payment List</li>

</ol>

</nav>

</nav>

<center>

<h3>Payments List</h3>

<b><font color=*"red"*><%=ServletUtility.getErrorMessage(request)%></font></b>

<b><font color=*"green"*><%=ServletUtility.getSuccessMessage(request)%></font></b>

</center>

<form action=*"*<%=OGDView.CUSTOMER\_LIST\_CTL%>*"* method=*"post"*>

<table class=*"table"*>

<thead class=*"thead-dark"*>

<tr>

<th scope=*"col"*>S No.</th>

<th scope=*"col"*>Payment Id</th>

<th scope=*"col"*>Product Code</th>

<th scope=*"col"*>Customer Name</th>

<th scope=*"col"*>Date</th>

<th scope=*"col"*>Amount</th>

</tr>

</thead>

<tbody>

<%

**int** pageNo = ServletUtility.getPageNo(request);

**int** pageSize = ServletUtility.getPageSize(request);

**int** index = ((pageNo - 1) \* pageSize) + 1;

PaymentBean bean = **null**;

List list = ServletUtility.getList(request);

Iterator<PaymentBean> iterator = list.iterator();

**while** (iterator.hasNext()) {

bean = iterator.next();

%>

<tr>

<th scope=*"row"*><%=index++%></th>

<td><%=bean.getPaymentId()%></td>

<% ProductModel pMModel=**new** ProductModel();

ProductBean pBean=pMModel.findByPk(bean.getProductId());

%>

<td><%=pBean.getItemCode()%></td>

<% CustomerModel cModel=**new** CustomerModel();

CustomerBean cBean=cModel.findByPk(bean.getCustomerId());

%>

<td><%=cBean.getName()%></td>

<td><%=DataUtility.getDateString(bean.getPaymentDate())%></td>

<td><%=bean.getAmount() %></td>

</tr>

<%} %>

</tbody>

</table>

<table >

<thead >

<tr>

<th ><input type=*"submit"* name=*"operation"* class=*"btn btn-primary pull-center"* value=*"*<%=PaymentListCtl.OP\_PREVIOUS%>*"* <%=(pageNo == 1) ? "disabled" : ""%>></th>

<%

PaymentModel model = **new** PaymentModel();

%>

<th ><input type=*"submit"* name=*"operation"* class=*"btn btn-primary pull-center"* value=*"*<%=PaymentListCtl.OP\_NEXT%>*"*

<%=((list.size() < pageSize) || model.nextPK() - 1 == bean.getId()) ? "disabled" : ""%> ></th>

</tr>

</thead>

</table>

<input type=*"hidden"* name=*"pageNo"* value=*"*<%=pageNo%>*"*> <input

type=*"hidden"* name=*"pageSize"* value=*"*<%=pageSize%>*"*>

</form>

<br>

<div style="margin-top: *193px*">

<%@ include file=*"Footer.jsp"* %>

</div>

</body>

</html><%@page import=*"com.online.garments.deal.controller.CustomerCtl"*%>

<%@page import=*"com.online.garments.deal.util.ServletUtility"*%>

<%@page import=*"com.online.garments.deal.util.DataUtility"*%>

<%@page import=*"com.online.garments.deal.bean.ProductBean"*%>

<%@page import=*"com.online.garments.deal.model.ProductModel"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Payment</title>

</head>

<body>

<%@ include file=*"Header.jsp"* %>

<br>

<nav aria-label=*"breadcrumb"* role=*"navigation"*> <nav

aria-label=*"breadcrumb"* role=*"navigation"*>

<ol class=*"breadcrumb"*>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.WELCOME\_CTL%>*"*>Home</a></li>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.PRODUCT\_LIST\_CTL%>*"*>Products</a></li>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.CUSTOMER\_CTL%>*"*>Customer</a></li>

<li class=*"breadcrumb-item active"* aria-current=*"page"*>Payment</li>

</ol>

</nav>

<div class=*"container"*>

<div class=*"card"*>

<div class=*"container-fliud"*>

<div class=*"wrapper row"*>

<div class=*"preview col-md-6"*>

<% Long pid=(Long)session.getAttribute("pid");

ProductModel pModel=**new** ProductModel();

ProductBean pBean=pModel.findByPk(pid);

%>

<div class=*"preview-pic tab-content"*>

<div class=*"tab-pane active"* id=*"pic-1"*><img src=*"*<%=OGDView.APP\_CONTEXT%>*/image/*<%=pBean.getImage()%>*"* /></div>

</div>

</div>

<div class=*"details col-md-6"*>

<h3 class=*"product-title"*><%=pBean.getProductName() %></h3>

<div class=*"rating"*>

<div class=*"stars"*>

<span class=*"fa fa-star checked"*></span>

<span class=*"fa fa-star checked"*></span>

<span class=*"fa fa-star checked"*></span>

<span class=*"fa fa-star"*></span>

<span class=*"fa fa-star"*></span>

</div>

<span class=*"review-no"*>41 reviews</span>

</div>

<h4 class=*"price"*>current price: <span><%=pBean.getPrice()%></span></h4>

</div>

</div>

</div>

</div>

</div>

<div col-md-5 img-thumbnail">

<div id=*"feedback"*> <div class=*"container"*>

<div class=*"col-md-5"*>

<div class=*"form-area"*>

<form role=*"form"* action=*"*<%=OGDView.CUSTOMER\_CTL%>*"* method=*"post"* >

<jsp:useBean id=*"bean"* class=*"com.online.garments.deal.bean.PaymentBean"*

scope=*"request"*></jsp:useBean>

<input type=*"hidden"* name=*"id"* value=*"*<%=bean.getId()%>*"*>

<input type=*"hidden"* name=*"createdBy"* value=*"*<%=bean.getCreatedBy()%>*"*>

<input type=*"hidden"* name=*"modifiedBy"* value=*"*<%=bean.getModifiedBy()%>*"*>

<input type=*"hidden"* name=*"createdDatetime"* value=*"*<%=DataUtility.getTimestamp(bean.getCreatedDatetime())%>*"*>

<input type=*"hidden"* name=*"modifiedDatetime"* value=*"*<%=DataUtility.getTimestamp(bean.getModifiedDatetime())%>*"*>

<br style="clear:*both*">

<h3 style="margin-bottom: *15px*; text-align: *left*: ;">Add Payment Detail</h3>

<b><font color=*"red"*> <%=ServletUtility.getErrorMessage(request)%>

</font></b>

<b><font color=*"Green"*> <%=ServletUtility.getSuccessMessage(request)%>

</font></b>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"CName"*

placeholder=*"Card Holder Name"* value=*""* >

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"CNum"*

placeholder=*"Card Number "* >

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"month"*

placeholder=*"Month"* >

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"year"*

placeholder=*"Year"* >

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"Cvv"*

placeholder=*"CVV"* >

</div>

<input type=*"submit"* name=*"operation"*

class=*"btn btn-primary pull-right"* value=*"*<%=CustomerCtl.OP\_PAYMENT%>*"*>

</form>

</div>

</div>

</div> </div>

<!--feedback-->

<br>

<div style="margin-top: *83px*">

<%@ include file=*"Footer.jsp"*%>

</div>

</body>

</html><%@page import=*"com.online.garments.deal.util.PropertyReader"*%>

<%@page import=*"java.util.Iterator"*%>

<%@page import=*"java.util.List"*%>

<%@page import=*"com.online.garments.deal.bean.ProductBean"*%>

<%@page import=*"com.online.garments.deal.util.ServletUtility"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Products</title>

</head>

<body>

<%@ include file=*"Header.jsp"* %>

<br>

<nav aria-label=*"breadcrumb"* role=*"navigation"*> <nav

aria-label=*"breadcrumb"* role=*"navigation"*>

<ol class=*"breadcrumb"*>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.WELCOME\_CTL%>*"*>Home</a></li>

<li class=*"breadcrumb-item active"* aria-current=*"page"*>Products</li>

</ol>

</nav>

<center>

<b><font color=*"red"*><%=ServletUtility.getErrorMessage(request)%></font></b>

<b><font color=*"green"*><%=ServletUtility.getSuccessMessage(request)%></font></b>

</center>

<hr>

<form action=*"*<%=OGDView.PRODUCT\_LIST\_CTL%>*"* method=*"post"*>

<div class=*"container"*>

<h3 class=*"h3"*>Products</h3>

<div class=*"row"*>

<%

**int** pageNo = ServletUtility.getPageNo(request);

**int** pageSize = ServletUtility.getPageSize(request);

**int** index = ((pageNo - 1) \* pageSize) + 1;

ProductBean bean = **null**;

List list = ServletUtility.getList(request);

Iterator<ProductBean> it = list.iterator();

**while** (it.hasNext()) {

bean = it.next();

%>

<div class=*"col-md-3 col-sm-6"*>

<div class=*"product-grid4"*>

<div class=*"product-image4"*>

<a href=*"#"*>

<img class=*"pic-1"* src=*"*<%=OGDView.APP\_CONTEXT%>*/image/*<%=bean.getImage()%>*"*>

<img class=*"pic-2"* src=*"*<%=OGDView.APP\_CONTEXT%>*/image/*<%=bean.getImage()%>*"*>

</a>

</div>

<div class=*"product-content"*>

<h3 class=*"title"*><a href=*"#"*><%=bean.getProductName()%></a></h3>

<div class=*"price"*>

<%=PropertyReader.getValue("cur")%><%=bean.getPrice()%>

<!-- <span>$16.00</span> -->

</div>

<%**if**(userLoggedIn){%>

<%**if**(userBean.getRole()!=3){ %>

<a class=*"add-to-cart"* href=*"ProductListCtl?idd=*<%=bean.getId()%>*"*>Delete</a>

<a class=*"add-to-cart"* href=*"*<%=OGDView.APP\_CONTEXT%>*/ctl/ProductCtl?id=*<%=bean.getId()%>*"*>Edit</a>

<%}**else**{ %>

<a class=*"add-to-cart"* href=*"CustomerCtl?pid=*<%=bean.getId()%>*"*>Book</a>

<%} %>

<%}**else**{ %>

<a class=*"add-to-cart"* href=*"CustomerCtl?pid=*<%=bean.getId()%>*"*>Book</a>

<%} %>

</div>

</div>

</div>

<%} %>

</div>

</div>

<input type=*"hidden"* name=*"pageNo"* value=*"*<%=pageNo%>*"*> <input

type=*"hidden"* name=*"pageSize"* value=*"*<%=pageSize%>*"*>

</form>

<hr>

<%@ include file=*"Footer.jsp"* %>

</body>

<%@page import=*"com.online.garments.deal.controller.ProductCtl"*%>

<%@page import=*"com.online.garments.deal.util.ServletUtility"*%>

<%@page import=*"com.online.garments.deal.util.DataUtility"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Product</title>

</head>

<body>

<%@ include file=*"Header.jsp"* %>

<br>

<nav aria-label=*"breadcrumb"* role=*"navigation"*> <nav

aria-label=*"breadcrumb"* role=*"navigation"*>

<ol class=*"breadcrumb"*>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.WELCOME\_CTL%>*"*>Home</a></li>

<li class=*"breadcrumb-item active"* aria-current=*"page"*>Product</li>

</ol>

</nav>

<div col-md-5 img-thumbnail">

<div id=*"feedback"*> <div class=*"container"*>

<div class=*"col-md-5"*>

<div class=*"form-area"*>

<form role=*"form"* action=*"*<%=OGDView.PRODUCT\_CTL%>*"* method=*"post"* enctype=*"multipart/form-data"* >

<jsp:useBean id=*"bean"* class=*"com.online.garments.deal.bean.ProductBean"*

scope=*"request"*></jsp:useBean>

<input type=*"hidden"* name=*"id"* value=*"*<%=bean.getId()%>*"*>

<input type=*"hidden"* name=*"createdBy"* value=*"*<%=bean.getCreatedBy()%>*"*>

<input type=*"hidden"* name=*"modifiedBy"* value=*"*<%=bean.getModifiedBy()%>*"*>

<input type=*"hidden"* name=*"createdDatetime"* value=*"*<%=DataUtility.getTimestamp(bean.getCreatedDatetime())%>*"*>

<input type=*"hidden"* name=*"modifiedDatetime"* value=*"*<%=DataUtility.getTimestamp(bean.getModifiedDatetime())%>*"*>

<br style="clear:*both*">

<h3 style="margin-bottom: *15px*; text-align: *left*: ;">Product</h3>

<b><font color=*"red"*> <%=ServletUtility.getErrorMessage(request)%>

</font></b>

<b><font color=*"Green"*> <%=ServletUtility.getSuccessMessage(request)%>

</font></b>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"code"*

placeholder=*"Item Code"* value=*"*<%=DataUtility.getStringData(bean.getItemCode())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("code", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"name"*

placeholder=*"Product Name"* value=*"*<%=DataUtility.getStringData(bean.getProductName())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("name", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"quantity"*

placeholder=*"Product Quantity"* value=*"*<%=DataUtility.getStringData(bean.getProductQuantity())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("quantity", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"choice"*

placeholder=*"Product Choice"* value=*"*<%=DataUtility.getStringData(bean.getProductChoice())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("choice", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"price"*

placeholder=*"Product Price"* value=*"*<%=(bean.getPrice()!=0.0)?bean.getPrice():""%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("price", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"file"* class=*"form-control"* name=*"photo"*

placeholder=*"Upload Image"* value=*"*<%=DataUtility.getStringData(bean.getImage())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("photo", request)%></font>

</div>

<input type=*"submit"* name=*"operation"*

class=*"btn btn-primary pull-right"* value=*"*<%=ProductCtl.OP\_SAVE%>*"*>

<input type=*"submit"* name=*"operation"*

class=*"btn btn-primary pull-right"* value=*"*<%=ProductCtl.OP\_RESET%>*"*>

</form>

</div>

</div>

</div> </div> <!--feedback-->

<br>

<div style="margin-top: *3px*">

<%@ include file=*"Footer.jsp"*%>

</div>

</body>

</html><%@page import=*"com.online.garments.deal.util.ServletUtility"*%>

<%@page import=*"com.online.garments.deal.bean.ProductBean"*%>

<%@page import=*"com.online.garments.deal.model.ProductModel"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Success</title>

</head>

<body>

<%@ include file=*"Header.jsp"* %>

<div class=*"container"*>

<div class=*"card"*>

<div class=*"container-fliud"*>

<div class=*"wrapper row"*>

<div class=*"preview col-md-6"*>

<% Long pid=(Long)session.getAttribute("pid");

ProductModel pModel=**new** ProductModel();

ProductBean pBean=pModel.findByPk(pid);

%>

<div class=*"preview-pic tab-content"*>

<div class=*"tab-pane active"* id=*"pic-1"*><img src=*"*<%=OGDView.APP\_CONTEXT%>*/image/*<%=pBean.getImage()%>*"* /></div>

</div>

</div>

<div class=*"details col-md-6"*>

<h3 class=*"product-title"*><%=pBean.getProductName() %></h3>

<div class=*"rating"*>

<div class=*"stars"*>

<span class=*"fa fa-star checked"*></span>

<span class=*"fa fa-star checked"*></span>

<span class=*"fa fa-star checked"*></span>

<span class=*"fa fa-star"*></span>

<span class=*"fa fa-star"*></span>

</div>

<b><font color=*"red"*> <%=ServletUtility.getErrorMessage(request)%>

</font></b>

<b><font color=*"Green"*> <%=ServletUtility.getSuccessMessage(request)%>

</font></b>

</div>

<h4 class=*"price"*>current price: <span><%=pBean.getPrice()%></span></h4>

</div>

</div>

</div>

</div>

</div>

<div style="margin-top: *100px*">

<%@ include file=*"Footer.jsp"* %>

</div>

</body>

</html><%@page import=*"com.online.garments.deal.controller.UserRegistrationCtl"*%>

<%@page import=*"com.online.garments.deal.util.ServletUtility"*%>

<%@page import=*"com.online.garments.deal.util.DataUtility"*%>

<%@page import=*"com.online.garments.deal.controller.OGDView"*%>

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Registration</title>

</head>

<body>

<%@ include file=*"Header.jsp"* %>

<br>

<nav aria-label=*"breadcrumb"* role=*"navigation"*> <nav

aria-label=*"breadcrumb"* role=*"navigation"*>

<ol class=*"breadcrumb"*>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.WELCOME\_CTL%>*"*>Home</a></li>

<li class=*"breadcrumb-item"*><a href=*"*<%=OGDView.LOGIN\_CTL%>*"*>Login</a></li>

<li class=*"breadcrumb-item active"* aria-current=*"page"*>User Registration</li>

</ol>

</nav>

<br>

<div col-md-5 img-thumbnail">

<div id=*"feedback"*> <div class=*"container"*>

<div class=*"col-md-5"*>

<div class=*"form-area"*>

<form role=*"form"* action=*"*<%=OGDView.USER\_REGISTRATION\_CTL%>*"* method=*"post"* >

<jsp:useBean id=*"bean"* class=*"com.online.garments.deal.bean.LoginBean"*

scope=*"request"*></jsp:useBean>

<input type=*"hidden"* name=*"id"* value=*"*<%=bean.getId()%>*"*>

<input type=*"hidden"* name=*"createdBy"* value=*"*<%=bean.getCreatedBy()%>*"*>

<input type=*"hidden"* name=*"modifiedBy"* value=*"*<%=bean.getModifiedBy()%>*"*>

<input type=*"hidden"* name=*"createdDatetime"* value=*"*<%=DataUtility.getTimestamp(bean.getCreatedDatetime())%>*"*>

<input type=*"hidden"* name=*"modifiedDatetime"* value=*"*<%=DataUtility.getTimestamp(bean.getModifiedDatetime())%>*"*>

<br style="clear:*both*">

<h3 style="margin-bottom: *15px*; text-align: *left*: ;">User Registration</h3>

<b><font color=*"red"*> <%=ServletUtility.getErrorMessage(request)%>

</font></b>

<b><font color=*"Green"*> <%=ServletUtility.getSuccessMessage(request)%>

</font></b>

<div class=*"form-group"*>

<input type=*"text"* class=*"form-control"* name=*"login"*

placeholder=*"Login Id"* value=*"*<%=DataUtility.getStringData(bean.getLogin())%>*"* >

<font color=*"red"*><%=ServletUtility.getErrorMessage("login", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"password"* class=*"form-control"*

name=*"password"* placeholder=*"Password"* value=*"*<%=DataUtility.getStringData(bean.getPassword()) %>*"* >

<font color=*"red"*> <%=ServletUtility.getErrorMessage("password", request)%></font>

</div>

<div class=*"form-group"*>

<input type=*"password"* class=*"form-control"*

name=*"confirmPassword"* placeholder=*"Confirm Password"* value=*"*<%=DataUtility.getStringData(bean.getConfirmPassword()) %>*"* >

<font color=*"red"*> <%=ServletUtility.getErrorMessage("confirmPassword", request)%></font>

</div>

<input type=*"submit"* name=*"operation"*

class=*"btn btn-primary pull-right"* value=*"*<%=UserRegistrationCtl.OP\_SIGN\_UP %>*"*>

</form>

</div>

</div>

</div> </div> <!--feedback-->

<br>

<div style="margin-top: *83px*">

<%@ include file=*"Footer.jsp"*%>

</div>

</body>

</html><%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Welcome</title>

</head>

<body>

<%@ include file=*"Header.jsp"* %>

<br>

<nav aria-label=*"breadcrumb"* role=*"navigation"*> <nav

aria-label=*"breadcrumb"* role=*"navigation"*>

<ol class=*"breadcrumb"*>

<li class=*"breadcrumb-item active"* aria-current=*"page"*>Welcome</li>

</ol>

</nav>

<div class=*"container"*>

<div class=*"row"*>

<div class=*"col-lg-12"*>

<div class=*"jumbotron"*>

<h1 class=*"display-3"*>Welcome To SPORTY SHOES ONLINE </div>

We are a Shoes and sports wear manufacturer form India</div>

We aspire to be a one stop shop for all your needs from shoes to sports wear </h1>

</div>

</div>

</div>

</div>

<div style="margin-top: *159px*">

<%@ include file=*"Footer.jsp"* %>

</div>

</body>

</html>