Internet and Technology Project documentation

Parsian Jewelry Company

Document is about an online shopping system for Parsian Jewelry Company which used three tier architecture of developing systems.

2009

Seyed Amirsaleh Salehzadeh

ICTEC

4/20/2009

Contents

[Introduction 4](#_Toc227949318)

[Scenario 4](#_Toc227949319)

[System Use case Diagram 4](#_Toc227949320)

[Site Map 7](#_Toc227949321)

[Website 7](#_Toc227949322)

[Home 7](#_Toc227949323)

[Shop 8](#_Toc227949324)

[Contact Us 10](#_Toc227949325)

[Register 10](#_Toc227949326)

[Search 11](#_Toc227949327)

[Confirm order 11](#_Toc227949328)

[Administrator 12](#_Toc227949329)

[Admin user control 13](#_Toc227949330)

[Order submit 14](#_Toc227949331)

[Project outlines 15](#_Toc227949332)

[Database 15](#_Toc227949333)

[System layers 17](#_Toc227949334)

[Web layer (interfaces) 17](#_Toc227949335)

[Java Server Pages (JSP) 17](#_Toc227949336)

[Servlets 34](#_Toc227949337)

[Common layer 48](#_Toc227949338)

[Data access layer 56](#_Toc227949339)

[Packages information 72](#_Toc227949340)

[Installation 72](#_Toc227949341)

[Database 72](#_Toc227949342)

[System 74](#_Toc227949343)

# Introduction

Parsian jewelry is one of the jewelry producers in Iran. The company decides to start selling and presenting their products in the internet through a web site. To reduce the expenditures of shopping centers such as rent of shops, manpower, electricity power, etc. The offered system was a system which allowed the admin to control the products, orders and customers and users can access to a basket to purchase products.

One of the advantages of the system is user can access to all information about products such as their name, price, description, product name and their image. Customers can register in the site to be a user of the system.

# Scenario

1. Amirsaleh is the owner of the web site with a username and password which has admin access.

Amirsaleh can add category such as “Rings”, “pierces”, etc.

Amirsaleh can add products with a selected category; define their prices, description of products and their image address.

Amirsaleh can select a user to be admin, and then the user can access the administrator page.

Amirsaleh can delete a user.

Amirsaleh can confirm purchases or allocating the idle statues to orders.

1. Homa is a girl who wants to be married but she has not enough time to go to market and choose her favorite jewelry for marriage ceremonies. She can enter the web site, register there and visit the products and choose her favorite.
2. Reza wants to take a gift for his girl friend in valentine day. He can go to the web site and choose the product which he wants and purchase it.

# System Use case Diagram







# Site Map

User control

Search

Shop

Register

Contact us

Order Control

Administrator

Home

# Website

## Home

The logo of the company and a graphical banner designed and placed in the top of the web page.

In the left, menu of the site and buttons which lead users to the pages exist.

In the right of the web site login page exist which users should enter their username and pass word to use the services of the site. (Figure 1)



Figure (main.jsp)

## Shop

Guests can access this page but before registration and logging in to the system they can’t access to basket or prepare a purchase.

They can also search and find their products based on their names. (Figure 2)

After logging in the system user can access to a virtual basket to add the products in it till confirming the purchase.

A welcome screen in the shop presented to the user. (Figure 3)

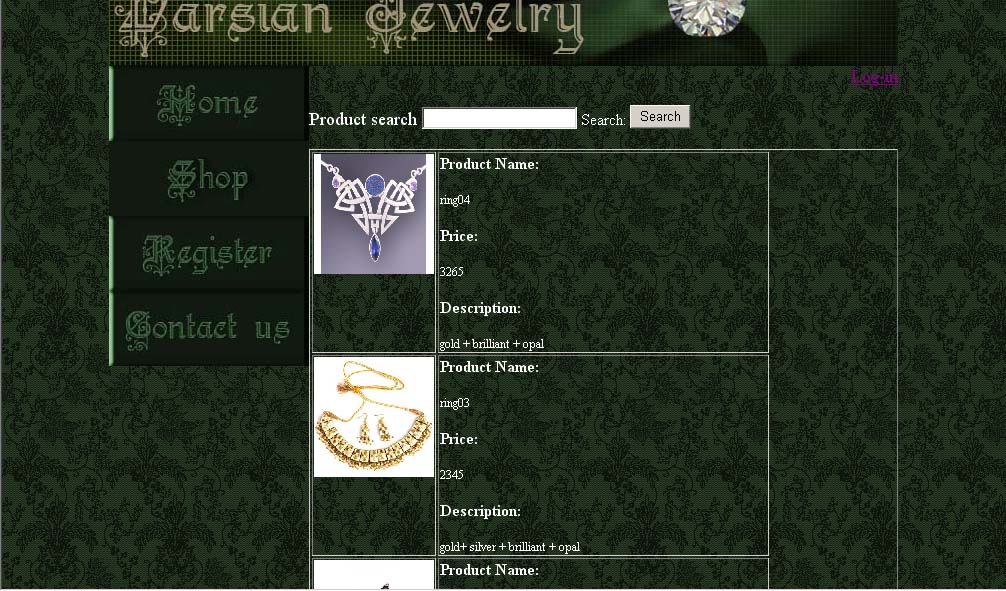
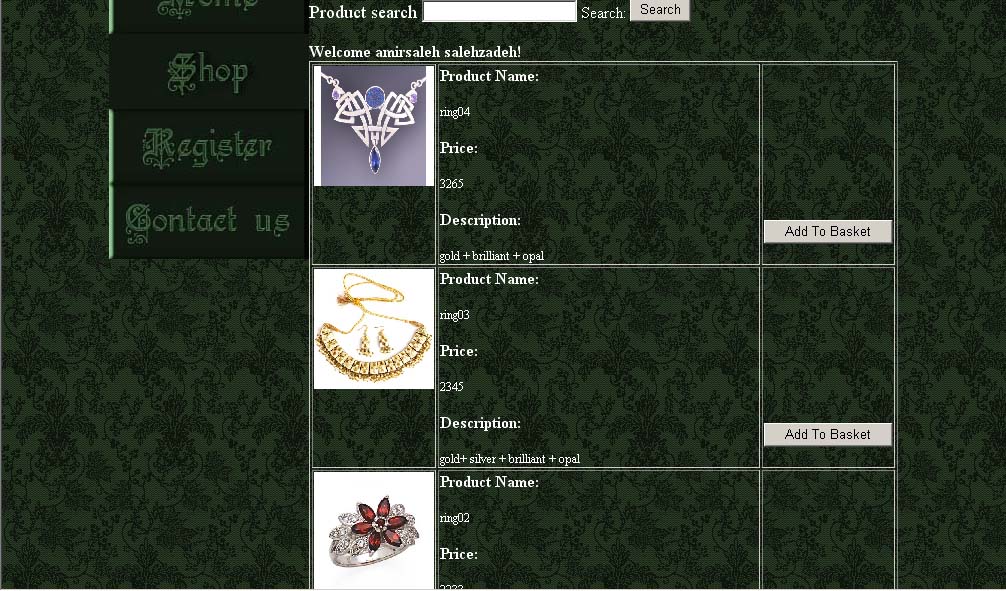


Figure (shop.jsp)

Figure (shop.jsp)

## Contact Us

The page contains contact information of the company.

## Register

This page contains a form which guest should fill it to be a user of the system. In formation in the form send to the database and stored there. (Figure 4)

Figure (Register.jsp)

## Search

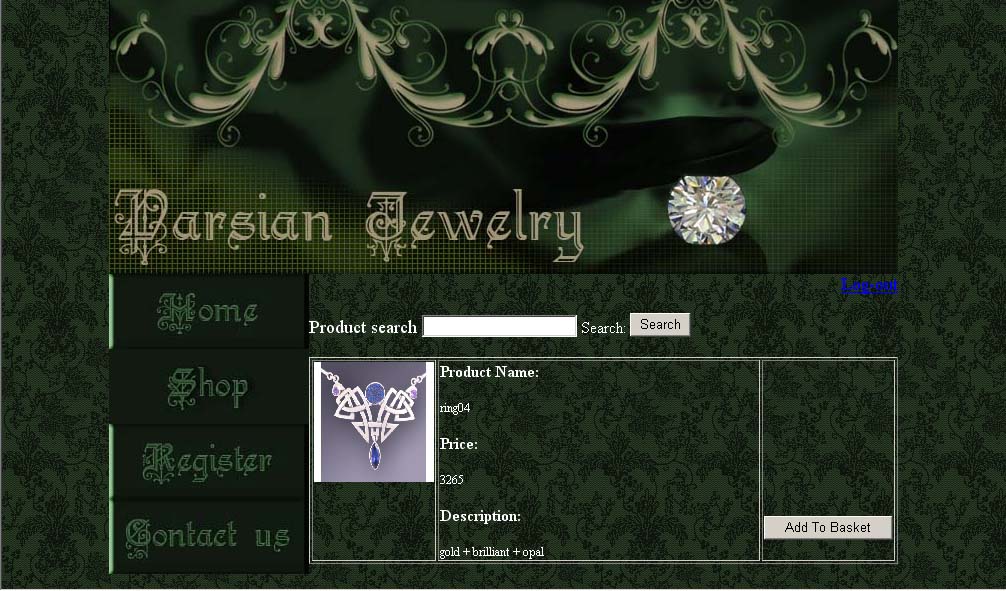
Search page is the result of what user searched. 

Figure (search.jsp)

## Confirm order

In this page user can view all in his/her basket and confirm the purchase.

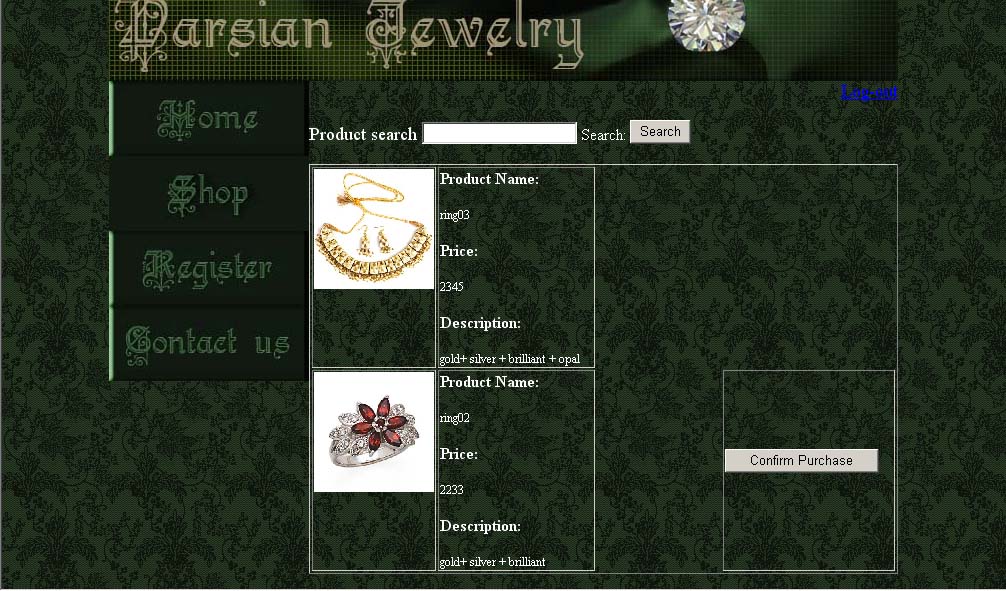


Figure (confirmorder.jsp)

## Administrator

In this page only users with the “admin” status for the access field in database can achieve the information.

In this page administrator of site can add category, add products through the forms in the page and access to the user control page and order control page. (Figure 7)

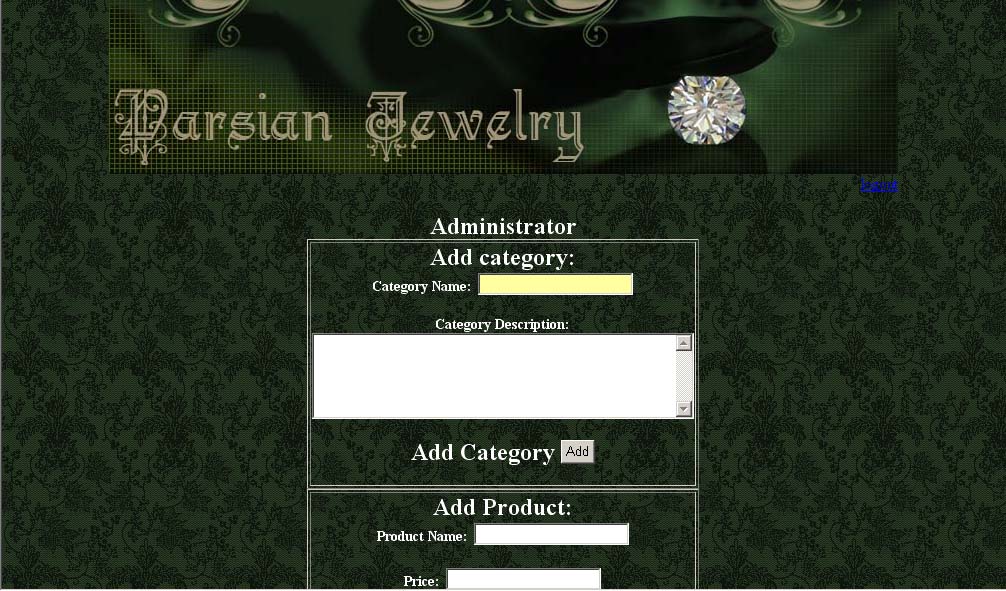


Figure (admin.jsp)

### Admin user control

Administrator of the site can allow users to be admin or not and delete the users through the table in the page.



Figure (adminuserlist.jsp)

### Order submit

Administrator of the site can allow orders to be confirmed or not through the table in the page.



Figure (Adminorderlist.jsp)

# Project outlines

System allows users to purchase product of Parsian Jewelry Company online. The server side program works with Java Server Page (JSP) by using Servlet technology to transfer the information between database and JSP pages.

The system works based on 3tier layer architecture of web based systems.

1-Web layer (interface)

2-Common layer (middle layer)

3-Data access layer (database transactions)

# Database

System is working based on the information which stored in database of it. System database is managed by “MySQL”. Table and rows structure is shown below.

|  |  |
| --- | --- |
| **Parsian - Category** | |
| **Field** | **Description** |
| Categn | Category name.  Primary key.  Type: VARCHAR |
| Description | Description of the category  Type: TEXT |

|  |  |
| --- | --- |
| **Parsian - Product** | |
| **Field** | **Description** |
| pname | Product name.  Primary key  Type: VARCHAR |
| Price | Product price.  Type: BIGINT |
| description | Description of the product.  Type: TEXT |
| categn | Category name.  Type: VARCHAR |
| image | Image source address of an image file.  Type: VARCHAR |

|  |  |
| --- | --- |
| **Parsian - User** | |
| **Field** | **Description** |
| name | Name of user.  Type: VARCHAR |
| fname | Family name of the user.  Type: VARCHAR |
| email | Email address of the user.  Type: VARCHAR |
| tel | Telephone number of the user.  Type: VARCHAR |
| usern | Username of the user.  Type: VARCHAR |
| pass | Password of the user.  Type: VARCHAR |
| id | An auto increment number  Type: INT  Primary key |
| access | In this case users can be a simple user or administrator.  Default value is user.  Type: VARCHAR |

|  |  |
| --- | --- |
| **Parsian - Order** | |
| **Field** | **Description** |
| usern | User name of the purchaser  Type: VARCHAR |
| productn | Name of the products which user wants to purchase  Type: VARCHAR |
| ID | An auto increment number  Type: VARCHAR  Primary key |
| status | Status of a order confirmed or not  Type: VARCHAR |

INT = integer

VARCHAR = characters

* Username in Parsian-user is unique.
* Products are unique and produced in limited number so categorized based on their product name.

# System layers

## Web layer (interfaces)

This layer connects the users to the common layer. Through the Java Server Pages (JSP) user interface designed and information between the common layer and interface transfers done by servlets which developed by Java.

### Java Server Pages (JSP)

This pages transfer the information between users and servlets through forms and tables.

All pages which exist in the system will discuss and introduced below.

#### main.jsp

This page is the main page of the system. Users can access to all pages from this page. A login form exists in this page too which allows users to log in to the system and use the services of the site.

There are some messages in the page which have given values from servlets.

<%**//”mess” will appeared when an order done successfully. It valued in ConfirmPurchase.java**

if(request.getParameter("mess")!=null)

out.println(request.getParameter("mess"));

%>

<form id="form1" name="form1" method="post" action="LoginServlet">**//this form check the username and password of users to be correct through its action LoginServlet**

<div align="center">

<span class="style3">Log-in to system </span><span class="style2"><br />

<span class="style2">

</span>

<p align="left">

<label>

<span class="style2">Username: </span>

<input type="text" name="usern" />

</label></p>

<p align="left">

<label><span class="style2">Password:</span>

<br>

<input type="password" name="pass" />

</label>

</p>

<p align="right">

<label>

<div align="center"><span class="style2">Submit</span>

<input type="submit" name="Submit" value="Submit" />

</label>

<%

if(request.getParameter("messl")!=null){**//”messl” when a username and password was invalid this message will print.**

out.println("<div class=\"style5\">");

out.println(request.getParameter("messl"));

out.println("</div>");}%>

<% if(request.getParameter("msg2")!=null){**//”msg2” will print when a user register to the system**

out.println("<div class=\"style5\">");

out.println(request.getParameter("msg2"));

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

}

%>

<p class="style2">Click <a href="Register.jsp">here</a> to register as new user </p>

</form>

#### Admin.jsp

This page includes all processes which admin can do. When a user who has the access layer of admin, logged in, system redirect the user to this page automatically.

<%**//when a category added successfully system print this message**

if(request.getParameter("msg3")!=null)

out.println(request.getParameter("msg3"));

out.println("<br>");

%>

<form id="form1" name="form1" method="post" action="**// system check the user to be admin then allows the form to add a category**

<%**//checking the user to be admin, users all information stored in “login” session**

ArrayList<User> t=(ArrayList<User>)session.getAttribute("login");

for(int j=0;j<t.size();j++){

if(t.get(j).getAccess().equals("admin")){

out.println("CategServlet");

}

}

%>

">

<label>Add category:<br />

<span class="style6">Category Name:</span>

<input type="text" name="name" />**//get the category name**

</label>

<p>

<label><span class="style6">Category Description:

</span><br>

<textarea name="description" id="textarea" cols="45" rows="5"></textarea>**//get the category description**

</label>

</p>

<p>

<label></label><label>Add Category

<input type="submit" name="Submit" value="Add" />

</label>

</p>

</form></td></tr></table>

<table border="1"><tr><td>

<form id="form1" name="form1" method="post" action="="**// system check the user to be admin then allows the form to add a product**

<%

for(int j=0;j<t.size();j++){

if(t.get(j).getAccess().equals("admin")){

out.println("ProdServlet");

}

}

%> ">

<label>Add Product:<br />

<span class="style6">Product Name:</span>

<input type="text" name="pname" />

</label>

<p>

<label><span class="style6">Price:</span>

<input type="text" name="price" />

</label>

</p>

<p>

<label><span class="style6">Product Description: </span><br>

<textarea name="description" id="textarea" cols="45" rows="5"></textarea>

</label>

</p>

<p>

<label><span class="style6">Image Source:</span>

<input type="text" name="img" />

</label>

</p>

<p>

<label><span class="style6">Category Name:</span>

<select name="select" id="select">

<%**//options of category menu read from “category” session and presents in a menu**

ArrayList<Category> categ=(ArrayList<Category>)session.getAttribute("category");

for(int i=0;i<categ.size();i++){

out.println("<option>");

out.println(categ.get(i).getCategn());

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

}

%>

</select>

</label>

</p>

<p>

<label>Add Product

<input type="submit" name="Submit" value="Add" />

</label>

</p>

</form></td></tr></table>

<%

for(int j=0;j<t.size();j++){ ="**// system check the user to be admin then allows to access these pages**

if(t.get(j).getAccess().equals("admin")){

out.println("<a href=\"UserControl\"><span class=\"style7\">User Control</span></a>");

out.println("<br><a href=\"OrderControl\"><span class=\"style7\">Orders Submit</span></a>");

}

}

%>

#### AdminUserlist.jsp

This page through a form allows admin to delete a user or select it as admin of system

<form action="AdminSelect" method="post"><table width="459" border="1">

<tr>

<td width="39"><span class="style4">Name</span></td>

<td width="87"><span class="style4">Family Name</span></td>

<td width="35"><span class="style4">Email</span></td>

<td width="67"><span class="style4">Telephone</span></td>

<td width="75"><span class="style4">User Name</span></td>

<td width="64"><span class="style4">Password</span></td>

<td width="46"><span class="style4">Administrator</span></td>

<td width="46"><span class="style4">Delete</span></td>

</tr>

<span class="style6">

<%**//information of all users stored in “usercontrol” session**

ArrayList<User> t=(ArrayList<User>)session.getAttribute("usercontrol");

for(int j=0;j<t.size();j++){**//reading the information of users from session and putting them in a table**

out.println("<tr><td><span class=\"style6\">");

out.println(t.get(j).getName());

out.println("</span></td><td><span class=\"style6\">");

out.println(t.get(j).getFname());

out.println("</span></td><td><span class=\"style6\">");

out.println(t.get(j).getEmail());

out.println("</span></td><td><span class=\"style6\">");

out.println(t.get(j).getTel());

out.println("</span></td><td><span class=\"style6\">");

out.println(t.get(j).getUsern());

out.println("</span></td><td><span class=\"style6\">");

out.println(t.get(j).getPass());

out.println("</span></td> ");

if(t.get(j).getAccess().equals("admin")){**//checking that user is admin or a simple user**

out.println("<td align=\"center\"><span class=\"style6\"><label>Admin</label></span>");

}

else{

out.print("<td align=\"center\"><input type=\"checkbox\" name=\"checkbox\" value=\"");

out.print(t.get(j).getUsern());

out.print("\"/></td>");

}

out.print("<td align=\"center\"><input type=\"checkbox\" name=\"delete\" value=\"");**//check boxes of delete option of users**

out.print(t.get(j).getUsern());

out.print("\"/></td></tr>");

}

%>

</table>

<p><label>

<input type="submit" name="button" id="button" value="Change" />

</label>

</p>

</form>

#### Adminorderlist.jsp

This page allows the admin of system to confirm an order.

<form action="ConfirmOrder" method="post"><table width="459" border="1">

<tr>

<td width="39"><span class="style4">User Name</span></td>

<td width="87"><span class="style4">Product Name</span></td>

<td width="35"><span class="style4">Submit</span></td>

</tr>

<span class="style6">

<%**//generating a list of all orders which not confirmed by the admin**

ArrayList<Order> t=(ArrayList<Order>)session.getAttribute("ordercontrol");

for(int j=0;j<t.size();j++){

if(t.get(j).getStatus().equals("false")){

out.println("<tr><td><span class=\"style6\">");

out.println(t.get(j).getUsern());

out.println("</span></td><td><span class=\"style6\">");

out.println(t.get(j).getProductn());

out.println("</span></td>");

out.print("<td align=\"center\"><input type=\"checkbox\" name=\"checkbox\" value=\"");

out.print(t.get(j).getId()); **//allocating the ID of order to checkboxes values**

out.print("\"/></td>");

}

}

%>

</table>

<p><label>

<input type="submit" name="button" id="button" value="Confirm" />

</label>

</p>

</form>

#### Register.jsp

This page allows a guest to register in the site as a user.

function validatePwd() {**//JavaScript function for checking the entered password be correct**

var pw1 = document.form1.pass.value;

var pw2 = document.form1.textfield7.value;

// check for a value in both fields.

if (pw1 == '' || pw2 == '') {

alert('Please enter your password twice.');

return false;

}

if (pw1 != pw2) {

alert ("You did not enter the same new password twice. Please re-enter your password.");

return false;

}

}

<form id="form1" name="form1" method="post" onSubmit="return validatePwd()" action="RegServlet">**//this form after validating the password send information to RegServlet.java**

<label><span class="style5">

<%**//this message shows that registration wasn’t successful**

if(request.getParameter("msg2")!=null)

out.println(request.getParameter("msg2"));

%>

Please fill the following form to confirm your registration, and wait to recieve the email by the administrator for your registration confirm. </span><br />

<br />

Name

<input type="text" name="name" />

</label>

<p>

<label>Family name

<input type="text" name="fname" />

</label>

</p>

<p>

<label>Email

<input type="text" name="email" />

</label>

</p>

<p>Tel number

<label>

<input type="text" name="tel" />

</label>

</p>

<p>

<label>UserName

<input type="text" name="usern" />

</label>

</p>

<p>

<label>Password

<input type="password" name="pass" />

</label>

</p>

<p>Re-password

<label>

<input type="password" name="textfield7" />

</label>

</p>

<p>

<label>Submit

<input type="submit" name="Submit" value="Submit" />

</label>

</p>

</form>

#### Shop.jsp

This page shows all products which exist in the database to guest. Logged in users can access a virtual basket and add their products in it till confirming the purchase.

<% if(session.getAttribute("login")!=null){**//if user logged in print logout button**

out.println("<a href=\"Logout\">");

out.println("Log-out");}

else{ out.println("<a href=\"main.jsp\">");**//offer guests to log in the system**

out.println("Log-in");}

%>

</a>

<form id="form1" name="form1" method="post" action="SearchServlet">**//form which search based on the product name**

<span class="style5">

<label>Product search</label>

</span>

<span class="style7">

<label></label>

</span>

<label>

<input type="text" name="product" />

</label>

<label><span class="style8">Search:</span>

<input type="submit" name="Submit" value="Search" />

</label>

</form>

<%**//print a welcome screen in the shop based on the name and family name the user**

ArrayList<User> u=(ArrayList<User>)session.getAttribute("login");

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

for(int j=0;j<u.size();j++){

out.println("<span class=\"style9\">Welcome "+u.get(j).getName()+" "+ u.get(j).getFname()+"!</span>");

}}

%>

<table width="100%" border="1" >

<%

ArrayList<Product> t=(ArrayList<Product>)session.getAttribute("productlist");

for(int i=0;i<t.size();i++)

{ **//creating a list of all products which stored in “productlist” session and put it in a table**

out.println("<tr><td valign=\"top\" width=\"120\" ><img src=\"images/shoppics/");

out.println(t.get(i).getImg());

out.println(".jpg\" width=\"120\" height=\"120\"></td><td><p class=\"style9\">Product Name:</p><p class=\"style11\">");

out.println(t.get(i).getPname());

out.println("</p><p class=\"style11\"><span class=\"style9\">Price:</span> </p><p class=\"style11\">" );

out.println(t.get(i).getPrice());

out.println("</p><p class=\"style11\"><span class=\"style9\">Description:</span></p><p class=\"style11\">");

out.println(t.get(i).getDescription());

out.println("</p></td>");

out.println("<td valign=\"bottom\" width=\"120\">");

if(session.getAttribute("login")!=null){**//for users which logged in add to basket form presents**

out.print("<form name=\"\" method=\"post\" action=\"AddToB\"><input type=\"hidden\" name=\"textfield\" id=\"textfield\" value=\"");**//add to a session which will create in AddToB.java**

out.print(t.get(i).getPname());**//allocating product name to a hidden text field**

out.println("\">");

out.println("<input type=\"submit\" name=\"button\" id=\"button\" value=\"Add To Basket\">");

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

}

out.println("</td></tr>");

}

%>

<tr><td><form action="BasketView" method ="post">**//form will presents all products which exist in the basket**

<input type="submit" name="submit" value="Order View">

</form></table>

#### Search.jsp

Values which stored in “search” session based on searched product name presents here.

<table width="100%" border="1" >

<%

ArrayList<Product> t=(ArrayList<Product>)session.getAttribute("search");

for(int i=0;i<t.size();i++){**//organizing the information which exist in “search” session in tables**

out.println("<tr><td valign=\"top\" width=\"120\" ><img src=\"images/shoppics/");

out.println(t.get(i).getImg());

out.println(".jpg\" width=\"120\" height=\"120\"></td><td><p class=\"style9\">Product Name:</p><p class=\"style11\">");

out.println(t.get(i).getPname());

out.println("</p><p class=\"style11\"><span class=\"style9\">Price:</span> </p><p class=\"style11\">" );

out.println(t.get(i).getPrice());

out.println("</p><p class=\"style11\"><span class=\"style9\">Description:</span></p><p class=\"style11\">");

out.println(t.get(i).getDescription());

out.println("</p></td>");

out.println("<td valign=\"bottom\" width=\"120\">");

if(session.getAttribute("login")!=null){**//presenting the add to basket form to logged in users**

out.print("<form name=\"\" method=\"post\" action=\"AddToB\"><input type=\"hidden\" name=\"textfield\" id=\"textfield\" value=\"");

out.print(t.get(i).getPname());

out.println("\">");

out.println("<input type=\"submit\" name=\"button\" id=\"button\" value=\"Add To Basket\">");

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

}

out.println("</td></tr>");

}

%>

</table>

#### Confirmorder.jsp

This page presents the users all products which exist in the basket. This page works similar to search page, search values are all product names which exist in the basket.

<table width="100%" border="1" >

<%

ArrayList<Product> t=(ArrayList<Product>)session.getAttribute("search");

for(int i=0;i<t.size();i++){

out.println("<tr><td valign=\"top\" width=\"120\" ><img src=\"images/shoppics/");

out.println(t.get(i).getImg());

out.println(".jpg\" width=\"120\" height=\"120\"></td><td><p class=\"style9\">Product Name:</p><p class=\"style11\">");

out.println(t.get(i).getPname());

out.println("</p><p class=\"style11\"><span class=\"style9\">Price:</span> </p><p class=\"style11\">" );

out.println(t.get(i).getPrice());

out.println("</p><p class=\"style11\"><span class=\"style9\">Description:</span></p><p class=\"style11\">");

out.println(t.get(i).getDescription());

out.println("</p></td>");

out.println("<td valign=\"bottom\" width=\"120\">");

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

}

%>

<td>

<form method="post" action="ConfirmPurchase" name="confirmpurchase">

<input type="submit" name="Submit" value="Confirm Purchase" />

</form>

</td></tr>

</table>

### Servlets

Servlets transfer the information from the common layer to the JSPs and from the JSPs to common layer.

These methods included in all servlets and created by the Net-Beans automatically.

// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">

/\*\*

\* Handles the HTTP <code>GET</code> method.

\* @param request servlet request

\* @param response servlet response

\*/

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

/\*\*

\* Handles the HTTP <code>POST</code> method.

\* @param request servlet request

\* @param response servlet response

\*/

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

/\*\*

\* Returns a short description of the servlet.

\*/

public String getServletInfo() {

return "Short description";

}

// </editor-fold>

}

#### AddToB.java

This servlet create a session as a basket and add products to it.

public class AddToB extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

try {

Product product1 = new Product();

product1.setPname(request.getParameter("textfield"));**//allocating product name to product1**

ArrayList<Product> a2b = new ArrayList<Product>();**//creating an array with the name a2b**

HttpSession session = request.getSession(true);

if (session.getAttribute("purchasesession") == null){**//adding the first product to the a2b means when session is empty**

a2b.add(product1);

}

else

{

ArrayList<Product> t=(ArrayList<Product>)session.getAttribute("purchasesession");

for(int j=0;j<t.size();j++)

{

Product p = new Product();

p.setPname(t.get(j).getPname());

a2b.add(p);**//adding the previous products which exist in the session into the a2b**

}

a2b.add(product1);**//adding the product1 to a2b**

}

session.setAttribute("purchasesession",a2b);**//adding a2b in session “purchasesession**”

response.sendRedirect("ShopServlet");**//sending the session to ShopServlet.java**

}catch(Exception ex){ex.printStackTrace();}

}

#### AdminSelect.java

This servlet get the information from adminuserlist.jsp and delete or update user status to admin.

public class AdminSelect extends HttpServlet {

private int j;

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException, SQLException {

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

PrintWriter out = response.getWriter();

MyProjBs mybs =new MyProjBs();

try {

User user = new User();

User user1 = new User();

String[] s = request.getParameterValues("checkbox");**//usernames which should be updated**

String[] p = request.getParameterValues("delete");**//usernames which should be deleted**

if (s!=null){

for(j=0;j<s.length;j++){**//changing the status of “user” for every checkbox in the form, to admin**

user.setUsern(s[j]);

mybs.updateUser(user);

}}

if (p!=null){

for(j=0;j<p.length;j++){**//deleting the “user1” which valued from the form**

user1.setUsern(p[j]);

mybs.deleteUser(user1);

}}

response.sendRedirect("UserControl");

}

finally {

out.close();

}

}

#### BasketView.java

This servlet create a session which contain all information about products which exist in the basket

public class BasketView extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

try {

HttpSession session = request.getSession(true);

ArrayList<Product> t=(ArrayList<Product>)session.getAttribute("purchasesession");**//getting all product names from “purchasesession” which contains all product names in the basket**

MyProjBs bs = new MyProjBs();

for(int j=0;j<t.size();j++)

{

bs.listProductSearch2(t.get(j));**//search all products which exist in the basket and putting them in the array ‘t’**

}

session.setAttribute("search",t); **//filling the session “search” with all information of every products**

response.sendRedirect("confirmorder.jsp");

}catch(Exception ex){ex.printStackTrace();}

}

#### CategServlet.java

This servlet add a new category from the filled form in Admin.jsp.

public class CategServlet extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

PrintWriter out = response.getWriter();

try {

Category categ=new Category();

categ.setCategn(request.getParameter("name"));

categ.setDescription(request.getParameter("description"));

MyProjBs bs = new MyProjBs();

if(bs.addCategory(categ))

{

response.sendRedirect("SelectCategory?msg3=Successful category added");

} else {

response.sendRedirect("Admin.jsp?msg3=Unsuccessful!");

}

} finally {

out.close();

}

}

#### ConfirmOrder.java

This servlet get the checked orders in the adminordelrlist.jsp and change their status to confirmed.

public class ConfirmOrder extends HttpServlet {

private int j;

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException, SQLException {

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

PrintWriter out = response.getWriter();

try {

Order order = new Order();

String[] s = request.getParameterValues("checkbox");**//ID of orders which checked in form, stored here**

for(j=0;j<s.length;j++){

MyProjBs mybs =new MyProjBs();

order.setId(Integer.parseInt(s[j]));**//changing type of values to integer and storing in “order”**

mybs.updateOrder(order);

}

response.sendRedirect("OrderControl");

}

finally {

out.close();

}

}

#### ConfirmPurchase.java

This servlet get the username from session “login” and product name from session “purchasesession” store in variables and objects then send them into the bs.MyProjBs.confirmOrder(username , product)

public class ConfirmPurchase extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException, SQLException {

try{

MyProjBs mybs =new MyProjBs();

String s = null;

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

HttpSession session = request.getSession(true);

ArrayList<Product> t=(ArrayList<Product>)session.getAttribute("purchasesession");

ArrayList<User> h=(ArrayList<User>)session.getAttribute("login");

for (int i=0;i<h.size();i++){**//getting username of logged in user**

s = h.get(i).getUsern();

}

for(int j=0;j<t.size();j++)

{**//getting all product names which exist in the basket**

Product p = new Product();

p.setPname(t.get(j).getPname());

mybs.confirmOrder(s,p);**calling bs.MyProjBs.confirmOrder method for the username ‘s’ and product ‘p’**

}

response.sendRedirect("main.jsp?mess=Your Purchase done successful!");

}catch(Exception ex){ex.printStackTrace();}

}

#### LoginServlet.java

This servlet check the existence of entered username and password in log-in form in main.jsp then store these fields in seession “login”.

public class LoginServlet extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

PrintWriter out = response.getWriter();

try {

user=new User();

user.setUsern(request.getParameter("usern"));

user.setPass(request.getParameter("pass"));

HttpSession session = request.getSession(true);

MyProjBs bs = new MyProjBs();

if(bs.login(user)) **//if username and password was correct**

{

ArrayList<User> ul=new ArrayList<User>();

ul=bs.listUserSession(user);

session.setAttribute("login",ul);**//store in session “login”**

if(bs.loginAdmin(user)){**//if user were admin**

response.sendRedirect("SelectCategory");

}

else{

response.sendRedirect("ShopServlet");

}} else {

response.sendRedirect("main.jsp?messl=Invalid username and password");

}

} finally {

out.close();

}

}

#### Logout.java

This servlet close the session “login”.

public class Logout extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

PrintWriter out = response.getWriter();

try {

HttpSession session = request.getSession(true);

session.invalidate();**//close the session**

response.sendRedirect("main.jsp");

} finally {

out.close();

}

}

#### OrderControl.java

This servlet creates a list of all orders which not confirmed.

public class OrderControl extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

try {

MyProjBs bs = new MyProjBs();

HttpSession session = request.getSession(true);

ArrayList<Order> ol=new ArrayList<Order>();

ol=bs.listOrder(); **//an array which contains a list of all orders**

session.setAttribute("ordercontrol",ol);**//store list of orders in session “ordercontrol”**

response.sendRedirect("Adminorderlist.jsp");

}catch(Exception ex){ex.printStackTrace();}

}

#### ProdServlet.java

This servlet add a new Product from the filled form in Admin.jsp.

public class ProdServlet extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException, SQLException {

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

PrintWriter out = response.getWriter();

try {

Product prod=new Product();

prod.setPname(request.getParameter("pname"));

prod.setPrice(Integer.parseInt(request.getParameter("price")));

prod.setImg(request.getParameter("img"));

prod.setDescription(request.getParameter("description"));

prod.setSelect(request.getParameter("select"));

MyProjBs mybs =new MyProjBs();

if(mybs.addProduct(prod)) {

response.sendRedirect("Admin.jsp?msg3=Successful product added");

}

else {

response.sendRedirect("Admin.jsp?msg3=Unsuccessful!");

}

} finally {

out.close();

}

}

#### RegServlet.java

This servlet add a new user from the filled form in Register.jsp.

public class RegServlet extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException, SQLException {

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

PrintWriter out = response.getWriter();

try {

User user=new User();

user.setName(request.getParameter("name"));

user.setFname(request.getParameter("fname"));

user.setUsern(request.getParameter("usern"));

user.setPass(request.getParameter("pass"));

user.setTel(request.getParameter("tel"));

user.setEmail(request.getParameter("email"));

user.setAccess("user");

MyProjBs mybs =new MyProjBs();

if(mybs.register(user)) {

response.sendRedirect("main.jsp?msg2=Successful registration");

}

else {

response.sendRedirect("Register.jsp?msg2=Unsuccessful registration try again");

}

} finally {

out.close();

} }

#### SearchServlet.java

This servlet get a product name, generate an array of product list and all product’s information then store it in session “search”.

public class SearchServlet extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

try {

Product producto = new Product();

producto.setPname(request.getParameter("product"));**//getting what product should search**

MyProjBs bs = new MyProjBs();

HttpSession session = request.getSession(true);

ArrayList<Product> pl=new ArrayList<Product>();

pl=bs.listProductSearch(producto);**//a list of information of searched product**

session.setAttribute("search",pl); **// store in session “search”**

response.sendRedirect("search.jsp");

}catch(Exception ex){ex.printStackTrace();}

}

#### SelectCategory.java

This servlet creates a list of all category names which defined by the admin and store in session “category”.

public class SelectCategory extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

try{

MyProjBs bs = new MyProjBs();

HttpSession session = request.getSession(true);

ArrayList<Category> cl=new ArrayList<Category>();

cl=bs.listCategory();

session.setAttribute("category",cl ); **//store the array of all category names in session “category”**

response.sendRedirect("Admin.jsp");

}catch(Exception ex){ex.printStackTrace();}

}

#### ShopServlet.java

This servlet creates a list of all products which defined by the admin and store in session “productlist”.

public class ShopServlet extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

try{

MyProjBs bs = new MyProjBs();

HttpSession session = request.getSession(true);

ArrayList<Product> pl=new ArrayList<Product>();

pl=bs.listProduct();

session.setAttribute("productlist",pl ); **//store the array of all information about products in session “productlist”**

response.sendRedirect("Shop.jsp");

}catch(Exception ex){ex.printStackTrace();}

}

#### UserControl.java

This servlet creates a list of all users which exist in the system then store their information in session “usercontrol”.

public class UserControl extends HttpServlet {

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

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

try {

MyProjBs bs = new MyProjBs();

HttpSession session = request.getSession(true);

ArrayList<User> ul=new ArrayList<User>();

ul=bs.listUser();

session.setAttribute("usercontrol",ul); **//store the array of all information about users in session “usercontrol”**

response.sendRedirect("AdminUserlist.jsp");

}catch(Exception ex){ex.printStackTrace();}

}

## Common layer

This layer is the middle layer which transfers the information between interface and data access layer by the methods which defines in it.

#### Category.java

This class defines a category, all its attributes and methods.

public class Category {

private String categn; **//category name as an attribute of class**

private String Description; **//category description as an attribute of class**

public String getCategn() {**//get method for category name which returns the category name value**

return categn;

}

public void setCategn(String categn) { **//set method for category name which returns the category name value**

this.categn = categn;

}

public String getDescription() {

return Description;

}

public void setDescription(String Description) {

this.Description = Description;

}

}

#### Order.java

This class defines an order, all its attributes and methods.

public class Order {

private String usern;

private String Productn;

private String status;

private Integer id;

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

public String getUsern() {

return usern;

}

public void setUsern(String usern) {

this.usern = usern;

}

public String getProductn() {

return Productn;

}

public void setProductn(String Productn) {

this.Productn = Productn;

}

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

}

#### Product.java

This class defines a product, all its attributes and methods.

public class Product {

private String pname;

private int price;

private String description;

private String img;

private String select;

public String getSelect() {

return select;

}

public void setSelect(String select) {

this.select = select;

}

public String getPname() {

return pname;

}

public void setPname(String pname) {

this.pname = pname;

}

public int getPrice() {

return price;

}

public void setPrice(int price) {

this.price = price;

}

public String getDescription() {

return description;

}

public void setDescription(String description) {

this.description = description;

}

public String getImg() {

return img;

}

public void setImg(String img) {

this.img = img;

}

}

#### User.java

This class defines a User, all its attributes and methods.

public class User {

private String name;

private String fname;

private String usern;

private String pass;

private String tel;

private String email;

private String access;

public String getAccess() {

return access;

}

public void setAccess(String access) {

this.access = access;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getFname() {

return fname;

}

public void setFname(String fname) {

this.fname = fname;

}

public String getUsern() {

return usern;

}

public void setUsern(String usern) {

this.usern = usern;

}

public String getPass() {

return pass;

}

public void setPass(String pass) {

this.pass = pass;

}

public String getTel() {

return tel;

}

public void setTel(String tel) {

this.tel = tel;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

}

## Data access layer

This layer contains a class which defines all database connections and transactions which system needs to insert, update, delete or retrieve information.

#### MyProjBs.java

package purchase.bs;

import purchase.common.User;

import purchase.common.Category;

import purchase.common.Product;

import java.sql.\*;

import java.util.ArrayList;

import purchase.common.Order;

public class MyProjBs {

public boolean register(User x){ **//register a user. Insert all information of a user into database**.

try{

Class.forName("com.mysql.jdbc.Driver");**//database driver address**

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");**//creating a connection**

String sql="Insert into user (name, fname, email, tel, usern, pass, access) values (?,?,?,?,?,?,?)"; **//sql syntax**

PreparedStatement ps=con.prepareStatement(sql);

ps.setString(1,x.getName());**//the value of user’s name**

ps.setString(2,x.getFname());

ps.setString(3,x.getEmail());

ps.setString(4,x.getTel());

ps.setString(5,x.getUsern());

ps.setString(6,x.getPass());

ps.setString(7,x.getAccess());

ps.execute();

ps.close();

con.close();

return true;

}

catch (Exception EX){

EX.printStackTrace();

return false;

}

}

public boolean login (User x)**//get the information of a user and check the validity of its username and password**

{

boolean result=false;

try{

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql="Select \* from user where usern like ? AND pass like ? ";

PreparedStatement ps=con.prepareStatement(sql);

ps.setString(1,x.getUsern());

ps.setString(2,x.getPass());

ResultSet rs = ps.executeQuery();

if(rs.next())

{

result=true;

}

else

{

result=false;

}

rs.close();

ps.close();

con.close();

return result;

}

catch (Exception EX){

EX.printStackTrace();

}

return result;

}

public boolean loginAdmin (User x)**//select the value of access field of a user which logged in the system**

{

boolean result=false;

try{

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql="Select access from user where usern like ? AND pass like ? ";

PreparedStatement ps=con.prepareStatement(sql);

ps.setString(1,x.getUsern());

ps.setString(2,x.getPass());

ResultSet rs = ps.executeQuery();

if(rs.next())

{

if (rs.getString("access").equals("admin"))

result=true;

else

return false;

}

rs.close();

ps.close();

con.close();

return result;

}

catch (Exception EX){

EX.printStackTrace();

}

return result;

}

public boolean addCategory (Category x){**//insert the category x in table of category in database**

try{

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql="Insert into category (categn, description) values (?,?)";

PreparedStatement ps=con.prepareStatement(sql);

ps.setString(1,x.getCategn());

ps.setString(2,x.getDescription());

ps.execute();

ps.close();

con.close();

return true;

}

catch (Exception EX){

EX.printStackTrace();

return false;

}

}

public boolean addProduct(Product x){ **//insert the product x in table of product in database**

try{

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql="Insert into product (pname, price, description, categn, image) values (?,?,?,?,?)";

PreparedStatement ps=con.prepareStatement(sql);

ps.setString(1,x.getPname());

ps.setInt(2,x.getPrice());

ps.setString(3,x.getDescription());

ps.setString(4,x.getSelect());

ps.setString(5,x.getImg());

ps.execute();

ps.close();

con.close();

return true;

}

catch (Exception EX){

EX.printStackTrace();

return false;

}

}

public ArrayList <Product> listProduct() {**//select all fields of all products in an array and returns it**

ArrayList<Product> plist=new ArrayList<Product>();

try {

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql = "SELECT \* FROM product";

Statement st=con.createStatement();

ResultSet rs;

rs = st.executeQuery(sql);

while(rs.next())

{

Product p = new Product();

p.setPname(rs.getString("pname"));

p.setPrice(rs.getInt("price"));

p.setDescription(rs.getString("description"));

p.setSelect(rs.getString("categn"));

p.setImg(rs.getString("image"));

plist.add(p);

}

return plist;

}

catch (Exception EX){

EX.printStackTrace();

}

return plist;

}

public ArrayList <Category> listCategory() {**//select all fields of all categories in an array and returns it**

ArrayList<Category> clist=new ArrayList<Category>();

try {

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql = "SELECT \* FROM category";

Statement st=con.createStatement();

ResultSet rs;

rs = st.executeQuery(sql);

while(rs.next())

{

Category p = new Category();

p.setCategn(rs.getString("categn"));

p.setDescription(rs.getString("description"));

clist.add(p);

}

return clist;

}

catch (Exception EX){

EX.printStackTrace();

}

return clist;

}

public ArrayList<Product> listProductSearch(Product i) {**//create an array which contains all fields where product name is equal to product name of product ‘i’.**

ArrayList<Product> plist=new ArrayList<Product>();

try {

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql="Select \* from product where pname like (?)";

PreparedStatement ps = con.prepareStatement(sql);

ps.setString(1,i.getPname());

ResultSet rs = ps.executeQuery();

Product o = new Product();

while(rs.next())

{

o.setPname(rs.getString("pname"));

o.setPrice(rs.getInt("price"));

o.setDescription(rs.getString("description"));

o.setSelect(rs.getString("categn"));

o.setImg(rs.getString("image"));

plist.add(o);

}

return plist;

}

catch (Exception EX){

EX.printStackTrace();

}

return plist;

}

public ArrayList <User> listUser() {**//select all fields of all Users in an array and returns it**

ArrayList<User> ulist=new ArrayList<User>();

try {

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql = "SELECT \* FROM user";

Statement st=con.createStatement();

ResultSet rs;

rs = st.executeQuery(sql);

while(rs.next())

{

User u = new User();

u.setName(rs.getString("name"));

u.setUsern(rs.getString("usern"));

u.setFname(rs.getString("fname"));

u.setPass(rs.getString("pass"));

u.setTel(rs.getString("tel"));

u.setEmail(rs.getString("email"));

u.setAccess(rs.getString("access"));

ulist.add(u);

}

return ulist;

}

catch (Exception EX){

EX.printStackTrace();

}

return ulist;

}

public boolean updateUser(User x){**//change the status of user ’x’ to “admin”**

try{

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql="UPDATE user SET access='admin' WHERE usern like ?";

PreparedStatement ps=con.prepareStatement(sql);

ps.setString(1,x.getUsern());

ps.execute();

ps.close();

con.close();

return true;

}

catch (Exception EX){

EX.printStackTrace();

return false;

}

}

public ArrayList <User> listUserSession(User i) {**//store all information about the User ‘i’** **in an array then returns the array**

ArrayList<User> plist=new ArrayList<User>();

try {

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql="Select \* from user where usern like ? and pass like ?";

PreparedStatement ps = con.prepareStatement(sql);

ps.setString(1,i.getUsern());

ps.setString(2,i.getPass());

ResultSet rs = ps.executeQuery();

User o = new User();

while(rs.next())

{

o.setName(rs.getString("name"));

o.setUsern(rs.getString("usern"));

o.setFname(rs.getString("fname"));

o.setPass(rs.getString("pass"));

o.setTel(rs.getString("tel"));

o.setEmail(rs.getString("email"));

o.setAccess(rs.getString("access"));

plist.add(o); }

return plist;

}

catch (Exception EX){

EX.printStackTrace();

}

return plist;

}

public boolean deleteUser(User x){**//delete the user ‘x’**

try{

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql="delete from user where usern like ?";

PreparedStatement ps=con.prepareStatement(sql);

ps.setString(1,x.getUsern());

ps.execute();

ps.close();

con.close();

return true;

}

catch (Exception EX){

EX.printStackTrace();

return false;

}

}

public boolean confirmOrder(String usern,Product o){**//insert username and product name in table order**

try{

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql ="INSERT INTO `order` (`usern` ,`productn`,`status` )VALUES (?,?,?)";

PreparedStatement ps=con.prepareStatement(sql);

ps.setString(1,usern);

ps.setString(2,o.getPname());

ps.setString(3,"false");

ps.execute();

ps.close();

con.close();

return true;

}

catch (Exception EX){

EX.printStackTrace();

return false;

}

}

public Product listProductSearch2(Product i) { {**//create a product which matches with product name of product ‘i’.**

try {

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql="Select \* from product where pname like (?)";

PreparedStatement ps = con.prepareStatement(sql);

ps.setString(1,i.getPname());

ResultSet rs = ps.executeQuery();

while(rs.next())

{

i.setPrice(rs.getInt("price"));

i.setDescription(rs.getString("description"));

i.setSelect(rs.getString("categn"));

i.setImg(rs.getString("image"));

}

}

catch (Exception EX){

EX.printStackTrace();

}

return i;

}

public ArrayList <Order> listOrder() {{**//create an array which contains all fields of all orders**

ArrayList<Order> olist=new ArrayList<Order>();

try {

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql = "SELECT \* FROM `order`";

Statement st=con.createStatement();

ResultSet rs;

rs = st.executeQuery(sql);

while(rs.next())

{

Order o = new Order();

o.setProductn(rs.getString("productn"));

o.setUsern(rs.getString("usern"));

o.setStatus(rs.getString("status"));

o.setId(rs.getInt("ID"));

olist.add(o);

}

return olist;

}

catch (Exception EX){

EX.printStackTrace();

}

return olist;

}

public boolean updateOrder(Order x){ **//update the order ’x’ status to confirmed based on the ID number of the order**

try{

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

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/parsian", "root", "");

String sql="UPDATE `parsian`.`order` SET `status` = 'true' WHERE `order`.`ID` =?";

PreparedStatement ps=con.prepareStatement(sql);

ps.setInt(1,x.getId());

ps.execute();

ps.close();

con.close();

return true;

}

catch (Exception EX){

EX.printStackTrace();

return false;

}

}

}

## Packages information

|  |  |
| --- | --- |
| **Package name** | **Classes** |
| Purchase.bs | MyProjBs.java |
| Purchase.common | User.java  Category.java  Product.java  Order.java |
| Purchase.servlet | AddToB.java  AdminSelect.java  BasketView.java  CategServlet.java  ConfirmOrder.java  ConfirmPurchase.java  LoginServlet.java  Logout.java  OrderControl.java  ProdServlet.java  RegServlet.java  SearchServlet.java  SelectCategory.java  ShopServlet.java  UserControl.java |

# Installation

## Database

Import the code below in Mysql syntax query.

|  |
| --- |
| -- phpMyAdmin SQL Dump  -- version 2.11.9.2  -- http://www.phpmyadmin.net  --  -- Host: localhost  -- Generation Time: Apr 16, 2009 at 09:02 AM  -- Server version: 5.0.67  -- PHP Version: 5.2.6  SET SQL\_MODE="NO\_AUTO\_VALUE\_ON\_ZERO";  --  -- Database: `parsian`  --  -- --------------------------------------------------------  --  -- Table structure for table `category`  --  CREATE TABLE IF NOT EXISTS `category` (  `categn` varchar(15) collate latin1\_general\_ci NOT NULL default '',  `description` text collate latin1\_general\_ci NOT NULL,  PRIMARY KEY (`categn`)  ) ENGINE=MyISAM DEFAULT CHARSET=latin1 COLLATE=latin1\_general\_ci;  --  -- Dumping data for table `category`  --  INSERT INTO `category` (`categn`, `description`) VALUES  ('rings', 'all rings which produced by the Parsian company\r\n');  -- --------------------------------------------------------  --  -- Table structure for table `order`  --  CREATE TABLE IF NOT EXISTS `order` (  `usern` varchar(25) default 'AA',  `productn` varchar(25) default 'A',  `ID` int(11) NOT NULL auto\_increment,  `status` varchar(7) NOT NULL,  PRIMARY KEY (`ID`)  ) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO\_INCREMENT=42 ;  --  -- Dumping data for table `order`  --  -- --------------------------------------------------------  --  -- Table structure for table `product`  --  CREATE TABLE IF NOT EXISTS `product` (  `pname` varchar(20) collate latin1\_general\_ci NOT NULL default '',  `price` bigint(20) NOT NULL default '0',  `description` text collate latin1\_general\_ci NOT NULL,  `categn` varchar(11) collate latin1\_general\_ci NOT NULL default '0',  `image` varchar(25) collate latin1\_general\_ci NOT NULL default '',  PRIMARY KEY (`pname`)  ) ENGINE=MyISAM DEFAULT CHARSET=latin1 COLLATE=latin1\_general\_ci;  --  -- Dumping data for table `product`  --  INSERT INTO `product` (`pname`, `price`, `description`, `categn`, `image`) VALUES  ('ring04', 3270, 'gold + brilliant + opal', 'rings', '4'),  ('ring03', 2300, 'gold+ silver + brilliant + opal', ' rings ', '3'),  ('ring02', 2200, 'gold+ silver + brilliant', ' rings ', '2'),  ('ring01', 2000, 'gold+ silver', ' rings ', '1');  -- --------------------------------------------------------  --  -- Table structure for table `user`  --  CREATE TABLE IF NOT EXISTS `user` (  `name` varchar(20) collate latin1\_general\_ci NOT NULL default '',  `fname` varchar(25) collate latin1\_general\_ci NOT NULL default '',  `email` varchar(50) collate latin1\_general\_ci NOT NULL default '',  `tel` varchar(15) collate latin1\_general\_ci NOT NULL default '',  `usern` varchar(15) collate latin1\_general\_ci NOT NULL default '',  `pass` varchar(15) collate latin1\_general\_ci NOT NULL default '',  `id` int(5) NOT NULL auto\_increment,  `access` char(5) collate latin1\_general\_ci NOT NULL default 'user',  PRIMARY KEY (`id`)  ) ENGINE=MyISAM DEFAULT CHARSET=latin1 COLLATE=latin1\_general\_ci AUTO\_INCREMENT=20 ;  --  -- Dumping data for table `user`  --  INSERT INTO `user` (`name`, `fname`, `email`, `tel`, `usern`, `pass`, `id`, `access`) VALUES  ('amirsaleh', 'salehzadeh', 'amirsa\_salehzadeh@yaoo.com', '2229348', 'admin', 'admin', 16, 'admin'); |

## System

Upload the system to your server; put the main.jsp as default page.

Add the file “mysql-connector-java-5.1.5-bin” into the library of your server side program.

Now you can login with username: Admin and password: Admin