University Of Limerick CE4208 Project Report

CE4208 Distributed System Team-Based Programming Assignment

2016 - 2017

Reiner Dojen

Amazon Online Shopping Web Application

Team member

Jiasen Tian	16060245
Qing Lin	16060261
Yifei Zhou	16070461
Zhikang Tian	16060288
Jing Liu	16060296
Meiyu Qi	16060318

Introduction 3

Use Case 3

Data Base - Mysql 4

Function Implement 8

logging facility (Message driven beans)........................... 18

OWASP Top 10 Implementation 20

Test Application 28

How to deploy application...39

Introduction

Amazon Shopping system is a distributed application. It is a fully functional online shopping system, mainly to provide online sales and online shopping services. Its functions include the management of goods, order management, membership management, online payment, etc.

Amazon Shopping system provides the necessary information exchange, payment settlement and physical delivery of these basic services

In terms of site security, we encrypt the user's password to ensure that the user's account is secure and reduces the probability of the site being attacked.

Use Case Implementation

1. Customer

Username: joe Password: 1D10T?

o Browse through all your items.
o Search products by ID number and browse through the search results.
o Search products by name and browse through the search results. o Add displayed items to their shopping cart.
o Remove items from their shopping cart.
o Edit their profile – must contain at least name, Customer ID and a message to other users. Name and ID are taken from Customer table, message can be any text – allow at least for 500 characters.

o View profiles from other users - provide search by name and search by ID.
o Check out or cancel current order
o Choose a satisfied shipping address
o Add new shipping address you want to choose.

2. Administrator

Username: toor Password: 4uIdo0!

o Add new products to the database.
o Remove products from the database.
o Increase/decrease the available amount (quantity_on_hand) of any product

Data Base - Mysql Implementation

MYSQL is the world's most popular open source database. MySQL can cost-effectively help us deliver high performance, scalable database applications. So we decided to use Mysql as our application database.

We wrote sql scripts to create our database.

In our database, 'amazonsystemdb', has eight tables. They are:

Customer	(Customers' informations)
Administer	(Administer's information)
ProductType	(The types of different products)
Product	(Products' details),
Address	(User's shipping address options)
Cart	(Some products which customer added but didn't pay)
Order_Record	(Customers' history of ordering)
Ordering	(It will be created when customer choose products

from cart which want to pay)

Here is the sql scripts for creating tables and adding datas.

(From p - p)

```
drop      database if exists `amazonsystemdb`;
create database if not exists `amazonsystemdb`; /*This is not signal qoutation ''', but the '`' on your the most left & up key on the
keyboard;*/
use amazonsystemdb;

CREATE TABLE Customer(
        Id                  INT                 NOT NULL UNIQUE,
        Username        VARCHAR (20)        NOT NULL UNIQUE,
        Password        VARCHAR(32)         NOT NULL,
        Age                 INT                 NOT NULL,
        phoneNumber    VARCHAR(19)         NOT NULL,
        UserMSG          TEXT(501)                 ,
        Email             VARCHAR(50)               ,
        PRIMARY KEY (Id)
);
INSERT INTO Customer (Id, Username,    Password, Age, phoneNumber, UserMSG, Email)
                    VALUES ( 2, 'joe'      , '1D10T?', 32,    '0838643219','Ahmedabad', 'sdutbruce@outlook.com');


CREATE TABLE Administer(
        Id                              INT                         NOT NULL UNIQUE,
        Username                   VARCHAR (20)         NOT NULL UNIQUE,
        Password                   VARCHAR (32)          NOT NULL,
        PRIMARY KEY (Id)
);
INSERT INTO Administer (Id, Username , Password )
                    VALUES (1 , 'toor'      , '4uIdo0!' );


CREATE TABLE ProductType(
        TypeID            INT                      NOT NULL UNIQUE,
        TypeText         VARCHAR (50)          NOT NULL,
        PRIMARY KEY (TypeID)
);
 INSERT INTO ProductType(TypeID, TypeText)
```

```
                     VALUES (1, 'Meat' );

 INSERT INTO ProductType(TypeID, TypeText)
                     VALUES (2, 'Vagetable' );

 INSERT INTO ProductType(TypeID, TypeText)
                     VALUES (3, 'Snack' );

 INSERT INTO ProductType(TypeID, TypeText)
                     VALUES (4, 'Seasoning' );

 INSERT INTO ProductType(TypeID, TypeText)
                     VALUES (5, 'Drink Powder' );

CREATE TABLE Product(
         ProductID          INT                  NOT NULL UNIQUE,
         ProductName         VARCHAR (50)        NOT NULL,
         TypeID              INT                  NOT NULL,
         Quantity           INT                  NOT NULL,
         Price              FLOAT                NOT NULL,
         Size               INT                 NOT NULL,
         ProductDescribe    TEXT                              ,
         PRIMARY KEY (ProductID)                        ,
         FOREIGN KEY (TypeID) REFERENCES ProductType(TypeID),
         CHECK ( Quantity >= 0 &&    Price >= 0 && Size >= 0)
);
INSERT INTO Product (ProductID, ProductName, TypeID, Quantity, Price, Size, ProductDescribe)
                VALUES (1          , 'Irish Potato', 2    ,    223       , 1.99 , 2000, 'Fresh f-level potatos'     );

INSERT INTO Product (ProductID, ProductName, TypeID, Quantity, Price,    Size, ProductDescribe )
                VALUES (2          , 'Chinese Potato', 2    ,    43      , 5.5    , 10000, 'AAA-level potatos' );

INSERT INTO Product (ProductID, ProductName, TypeID, Quantity, Price,    Size, ProductDescribe )
                VALUES (3          , 'Irish Beef' , 1    ,    140        , 4       ,    500, 'Fresh Beefs');

INSERT INTO Product (ProductID, ProductName,              TypeID, Quantity, Price,    Size, ProductDescribe )
                VALUES (4          , 'Chinese normal tea' , 5    ,    1010        , 2       ,    220, 'Tea from the remote China');

INSERT INTO Product (ProductID, ProductName, TypeID, Quantity, Price,    Size, ProductDescribe )
                VALUES (5          , 'Chinese Sprcial tea' , 5    ,    1010        , 2.4       ,    220, 'Tea from the remote China');

INSERT INTO Product (ProductID, ProductName, TypeID, Quantity, Price,    Size, ProductDescribe )
                VALUES (6          , 'Red Soy' ,        4    ,    1140        , 1       ,    500, 'Speciy Soy');
```

```
INSERT INTO Product (ProductID, ProductName, TypeID, Quantity, Price,    Size, ProductDescribe )
                    VALUES (7            , 'Chiken Powder' , 4    ,    1110        , 0.4       ,    110, 'Chiken Powders');


CREATE TABLE Address(
         Username              VARCHAR(20)       NOT NULL,
         Address_ID            INT                       NOT NULL,
         Address               VARCHAR (50)       NOT NULL,
         FOREIGN KEY (Username) REFERENCES Customer(Username),
         PRIMARY KEY (Username, Address_ID)
);

INSERT INTO Address(Username, Address_ID, Address)
                  VALUES ('joe', 000001, 'limerick..' );

CREATE TABLE Cart(
          Username              VARCHAR(20)       NOT NULL,
          Product_ID            INT                       NOT NULL,
          Product_Quantity INT                          NOT NULL,
          Add_Date              TIMESTAMP          NOT NULL default CURRENT_TIMESTAMP,
          PRIMARY KEY (Username, Product_ID),
          FOREIGN KEY (Username) REFERENCES Customer(Username),
          FOREIGN KEY (Product_ID) REFERENCES Product(ProductID),
          CHECK ( Product_Quantity >= 0 )
);

INSERT INTO Cart(Username,    Product_ID, Product_Quantity)
                          VALUES ('joe', 00043, 3);

CREATE TABLE Order_Record(
          Username            VARCHAR(20)      NOT NULL,
          Order_ID             INT                      NOT NULL,
          Total_Price        DOUBLE               NOT NULL,
          Order_Date         TIMESTAMP         NOT NULL default CURRENT_TIMESTAMP,
          Status                BOOLEAN            NOT NULL,
          Address              VARCHAR (50)      NOT NULL,
          FOREIGN KEY (Username) REFERENCES Customer(Username),
          PRIMARY KEY (Order_ID)
);

INSERT INTO Order_Record(Username, Order_ID, Total_Price, Status, Address )
                           VALUES ('joe', 00043, 567.89, TRUE, 'limerrick');
```

```
/* Refactor:UserID: VARCHAR(20) && Change name UserID into USERNAME */

CREATE TABLE Ordering(
        Order_ID            INT                 NOT NULL,
        ProductID           INT                 NOT NULL,
        Product_Quantity INT                    NOT NULL,
        PRIMARY KEY (Order_ID, ProductID),
        FOREIGN KEY (Order_ID)    REFERENCES Order_Record(Order_ID),
        FOREIGN KEY (ProductID) REFERENCES Product(ProductID),
        CHECK (Product_Quantity >= 0 )
);

INSERT INTO Ordering(Order_ID, ProductID, /*Username,*/ Product_Quantity)
                VALUES (00043, 00043, /*'joe',*/ 2);
```

Function Implementation （With images）

a. Log in: When user want to review this web, the log in is the first page. You must type your correct user name and password to log into amazon online shopping application. (As Fig 1 shows)

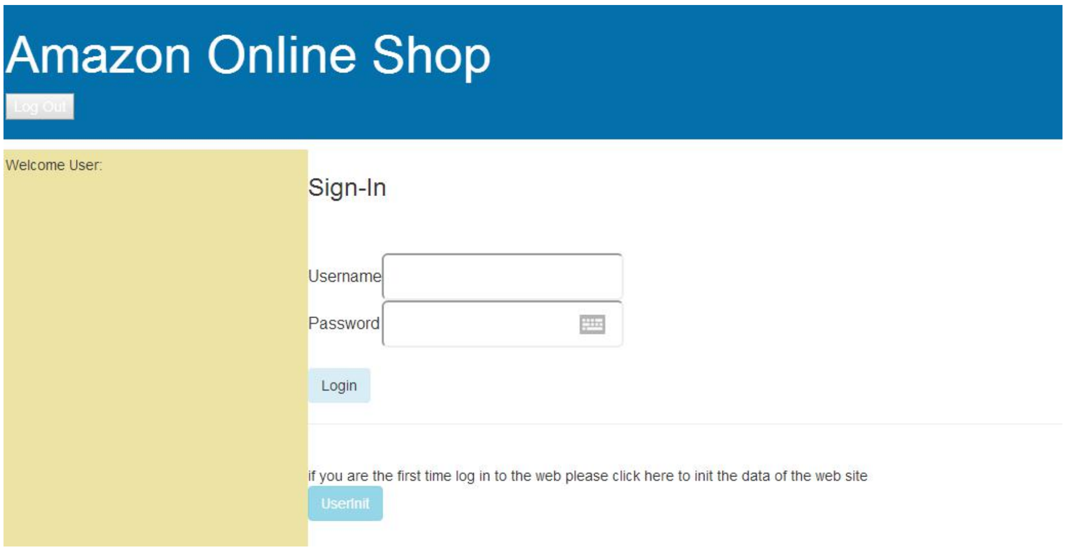

Amazon Online Shop
Log Out
Welcome User:
Sign-In
Username
Password
Login
if you are the first time log in to the web please click here to init the data of the web site
UserInit

Fig 1

b. Log into the Customer or Administrator page (Fig 3)

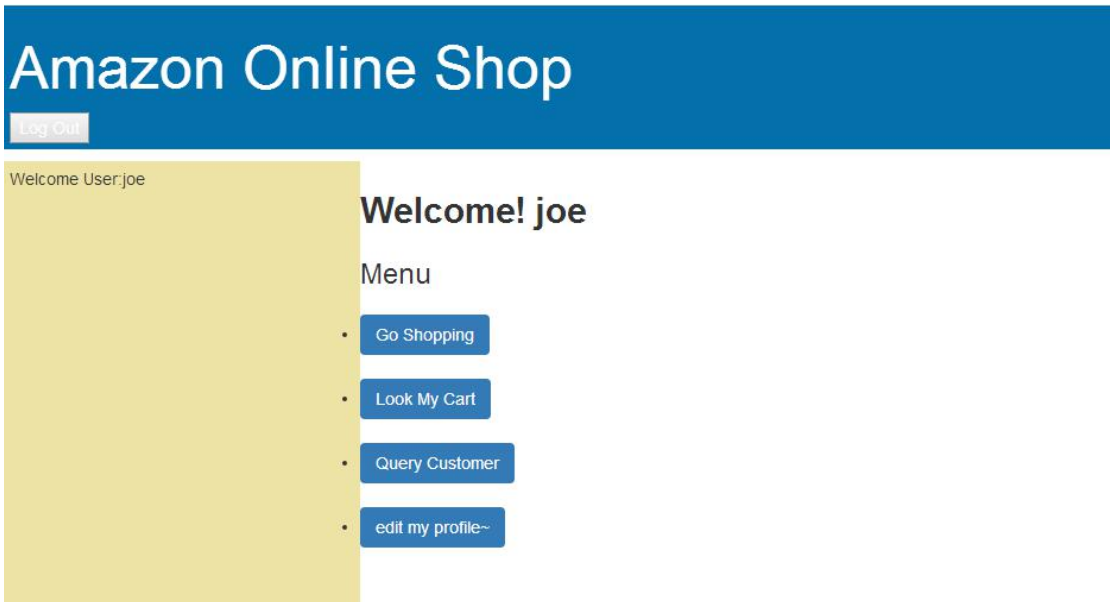


Fig 2

When you log in as a customer , you can find five buttons with five different options, they are Log out, Go shopping, Check my cart, Query Customer and Edit my profile. (Fig 2)

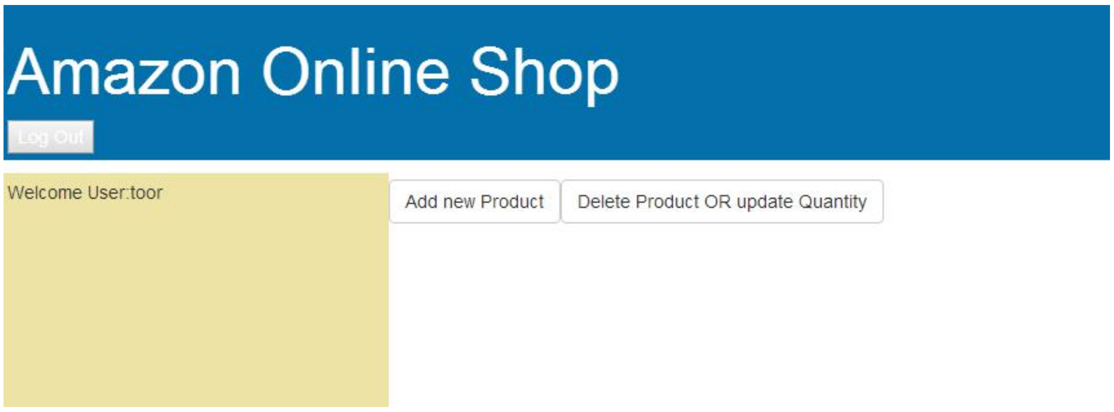


Fig 3

When you log in as an administrator, you will find a very simple page. Administrator can click the buttons to go to two different pages(Add new

Product and Delete Product OR Update Quantity).

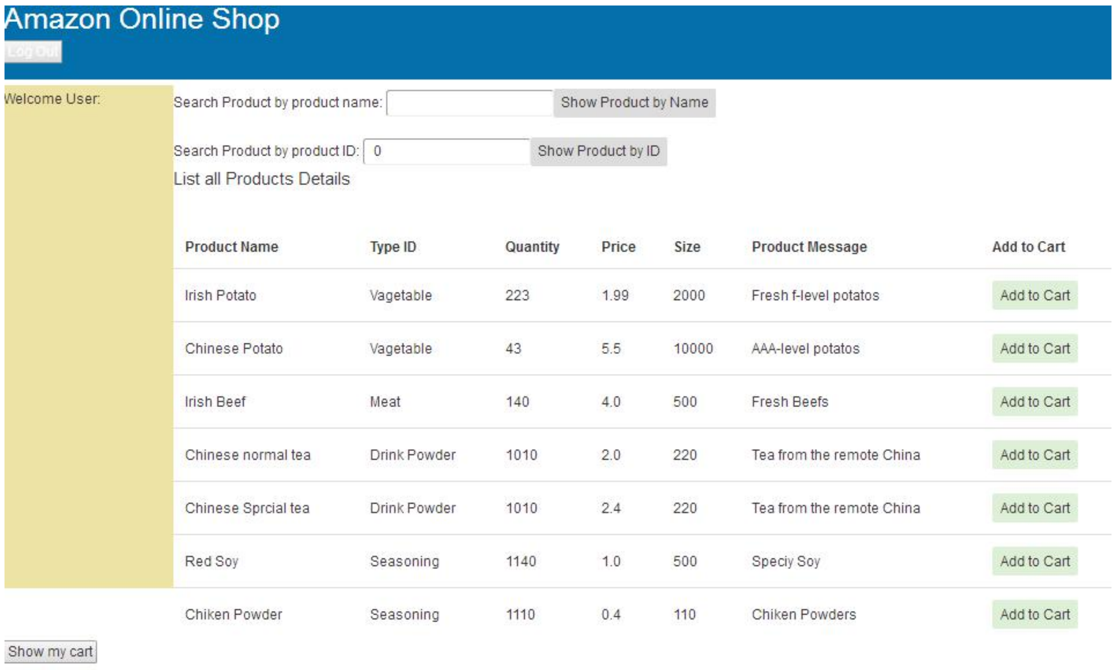
Next we will shows you some customer's operation details.

1. Review the product add cart

As we can see(In Fig 4), when customer go into products browsing page, there is a table which contains all the information of products and each line has an "Add To Cart" buttons which you can click it to add this product to your own cart.

You can also search products by its name or ID.

Finally, you can click "Show my cart" to check your cart.

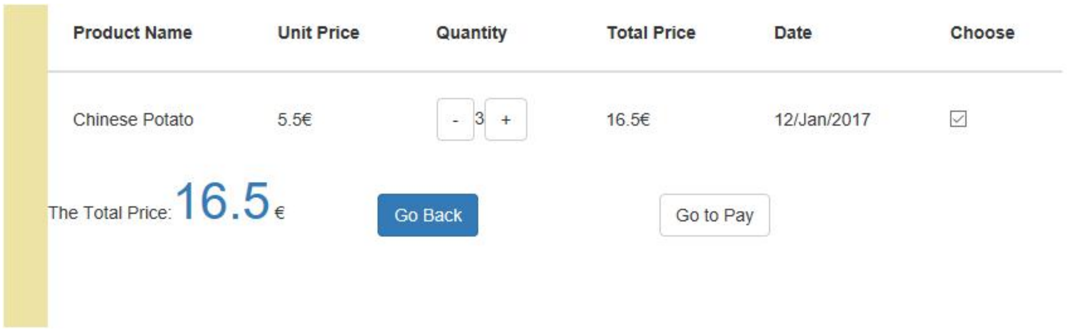


List all Products Details

Product Name	Type ID	Quantity	Price	Size	Product Message	Add to Cart
Irish Potato	Vagetable	223	1.99	2000	Fresh f-level potatos	Add to Cart
Chinese Potato	Vagetable	43	5.5	10000	AAA-level potatos	Add to Cart
Irish Beef	Meat	140	4.0	500	Fresh Beefs	Add to Cart
Chinese normal tea	Drink Powder	1010	2.0	220	Tea from the remote China	Add to Cart
Chinese Sprcial tea	Drink Powder	1010	2.4	220	Tea from the remote China	Add to Cart
Red Soy	Seasoning	1140	1.0	500	Speciy Soy	Add to Cart
Chiken Powder	Seasoning	1110	0.4	110	Chiken Powders	Add to Cart

Fig 4

2. User viewing the cart



Product Name	Unit Price	Quantity	Total Price	Date	Choose
Chinese Potato	5.5€	- 3 +	16.5€	12/Jan/2017	✓

The Total Price: 16.5€

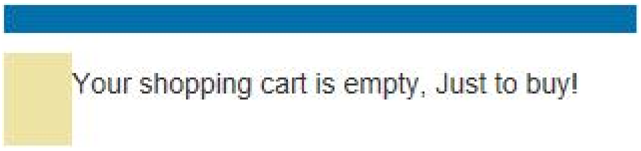
Fig 5

When the user have logged in, they can view their cart, and click the choose button to choose which product they want to buy. Besides, user also can change the amounts of products, and user click

this button, we will turn to payment page. If user doesn't choose anything, the

button would be disabled.

Notes: If there are not items in shopping cart, it will display a warning. (As Fig 6 shows)

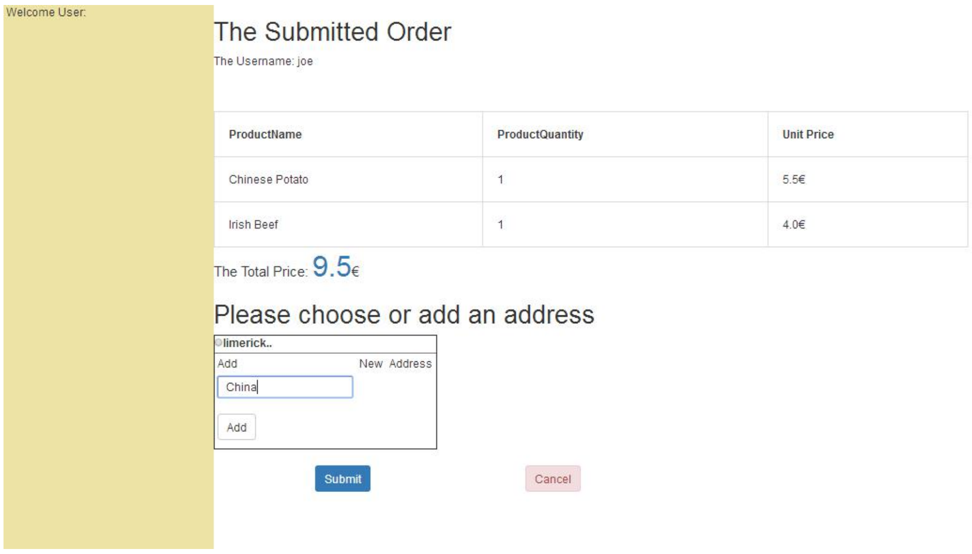


Your shopping cart is empty, Just to buy!

Fig 6

If everything is smoothly going and the web will generate a Order.

When user choose to payment, it will show a order. It mainly list the details of these products and user. (Fig 7.1)


Welcome User:

The Submitted Order

The Username: joe

ProductName	ProductQuantity	Unit Price
Chinese Potato	1	5.5€
Irish Beef	1	4.0€

The Total Price: 9.5€

Please choose or add an address

○ limerick..

Add New Address

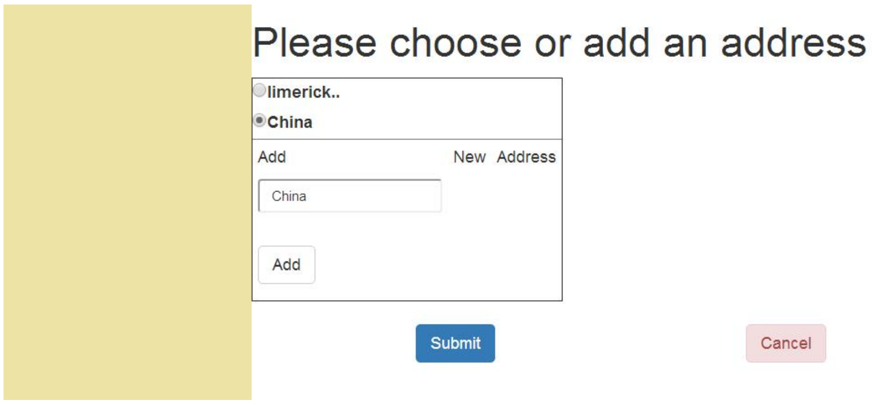
China

Fig 7.1

Choose the address Or add a new Address

If user do not satisfy their current address, they can add more one.

If user want to add more address. E.g. They want to add a new one, 'ADDRESS'. User can input their new address in this box, and then click add, what will happen? (Fig 7.2)


Please choose or add an address

○ limerick..

● China

Add New Address

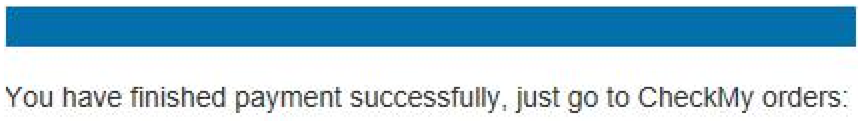
China

Fig 7.2

As Fig 7.2 shows, you have already became successful to add new address in your database and you can choose it to submit the order. Of cause you also can click cancel buttons to turn to the homepage.

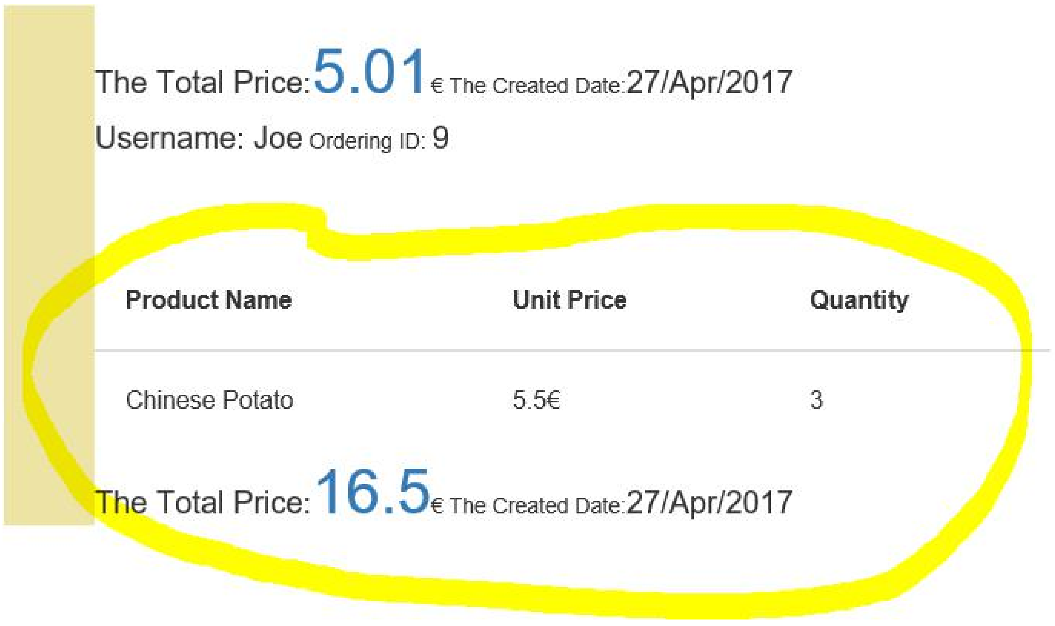
Submitted Payment

At the bottom of this page, it list of the available address which user want to be sent. User can choose randomly one. And Then click submit button. If every information is correct you will get a tip like Fig 7



You have finished payment successfully, just go to CheckMy orders:

Check my history orders



The Total Price:5.01€ The Created Date:27/Apr/2017
Username: Joe Ordering ID: 9

Product Name	Unit Price	Quantity
Chinese Potato	5.5€	3

The Total Price:16.5€ The Created Date:27/Apr/2017

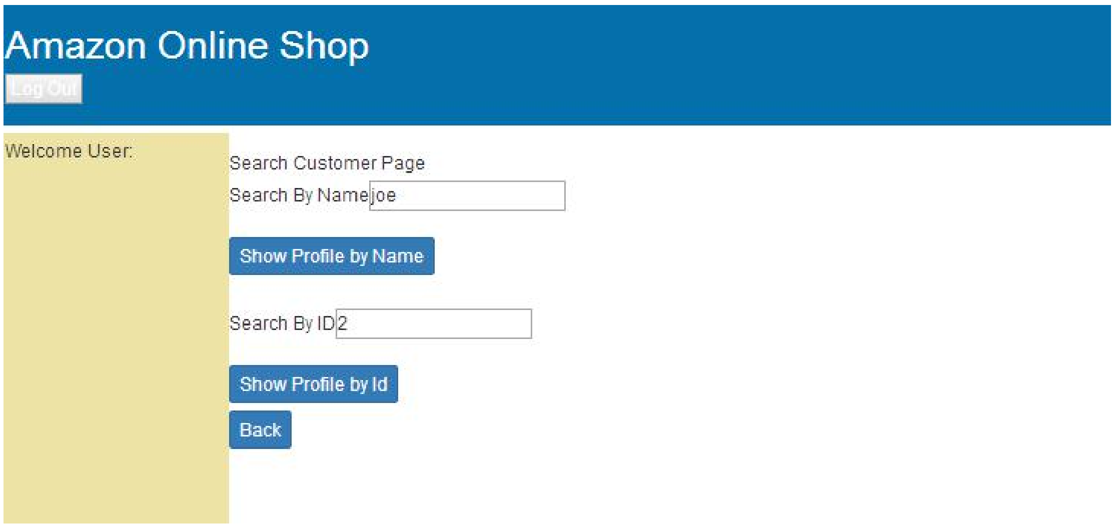
Fig 8

After payment we can check our historical orders list, and we can find that it had add a new record which we just added.(Fig 8)

Check and search customer information

You can search the customer information by User Name or User ID.

(Fig 9)



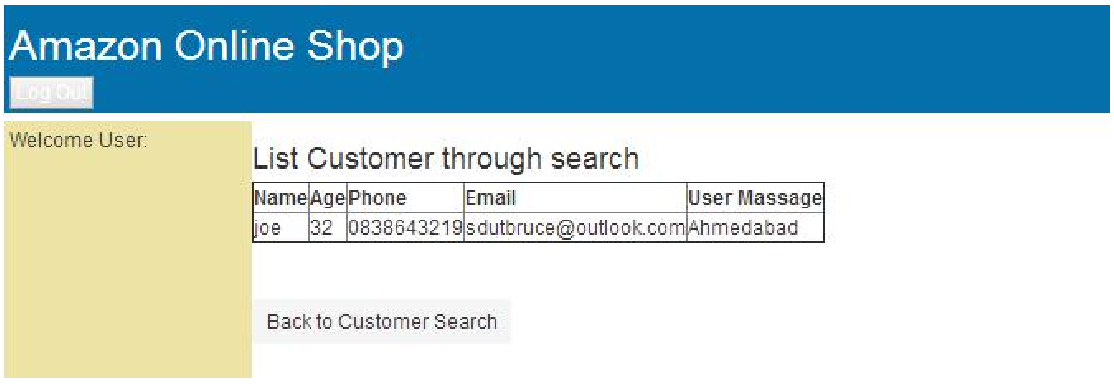
Search Customer Page

Search By Name joe

Search By ID 2

Fig 9

And you will get some information of this customer. (Fig 10)



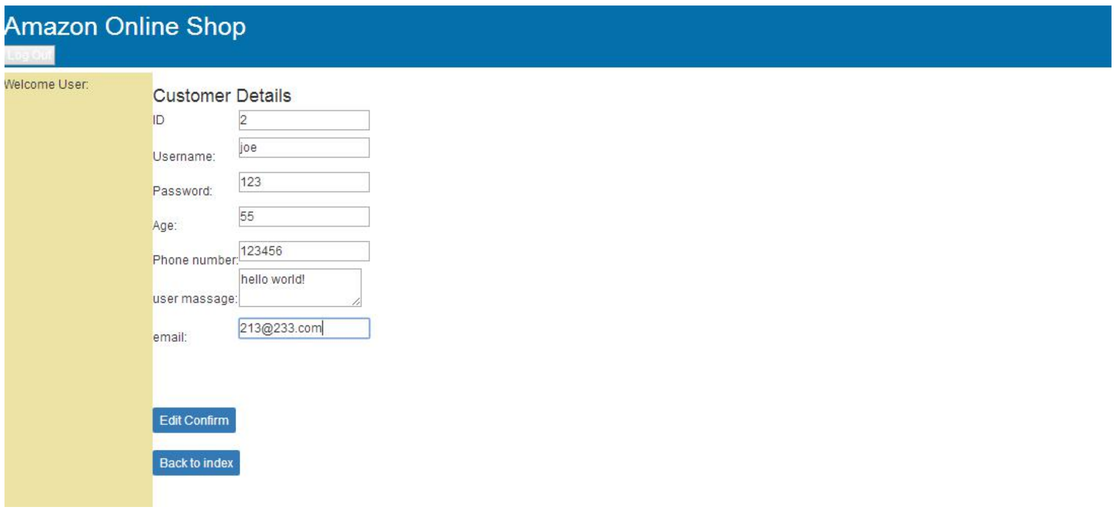
List Customer through search

Name	Age	Phone	Email	User Massage
joe	32	0838643219	sdutbruce@outlook.com	Ahmedabad

Fig 10

Change your Profile

As a customer, you can also change our personal information. (Fig 11)


Amazon Online Shop

Log Out

Welcome User:

Customer Details

ID	2
Username:	joe
Password:	123
Age:	55
Phone number:	123456
user massage:	hello world!
email:	213@233.com

Edit Confirm

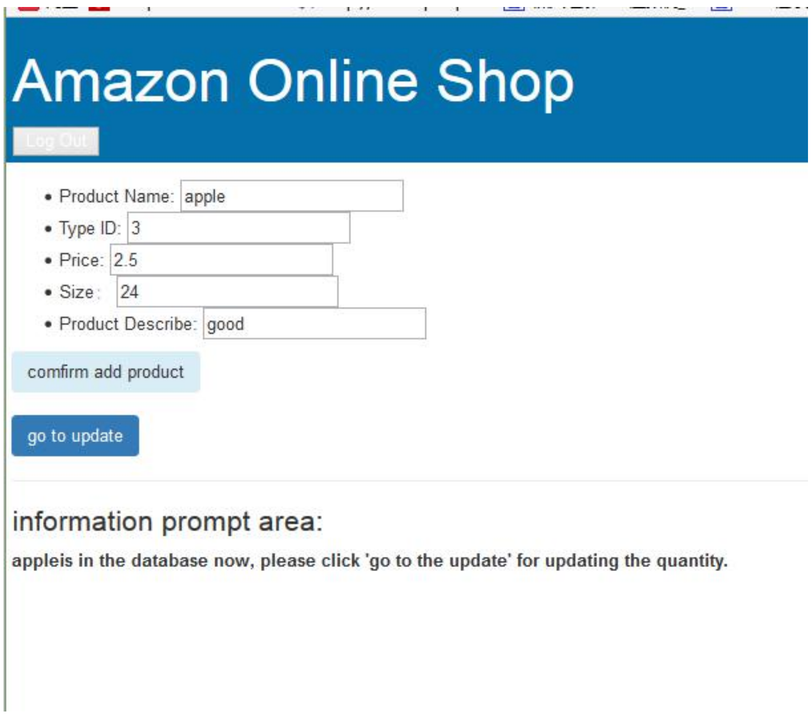
Back to index

Fig 11

Next we will show some details of **Administer Function**

Add a New Product

Administer can enter the details of new product and click "Confirm the product" to add a new product to the stock database. (As Fig 12 shows)



- Product Name: apple
- Type ID: 3
- Price: 2.5
- Size: 24
- Product Describe: good

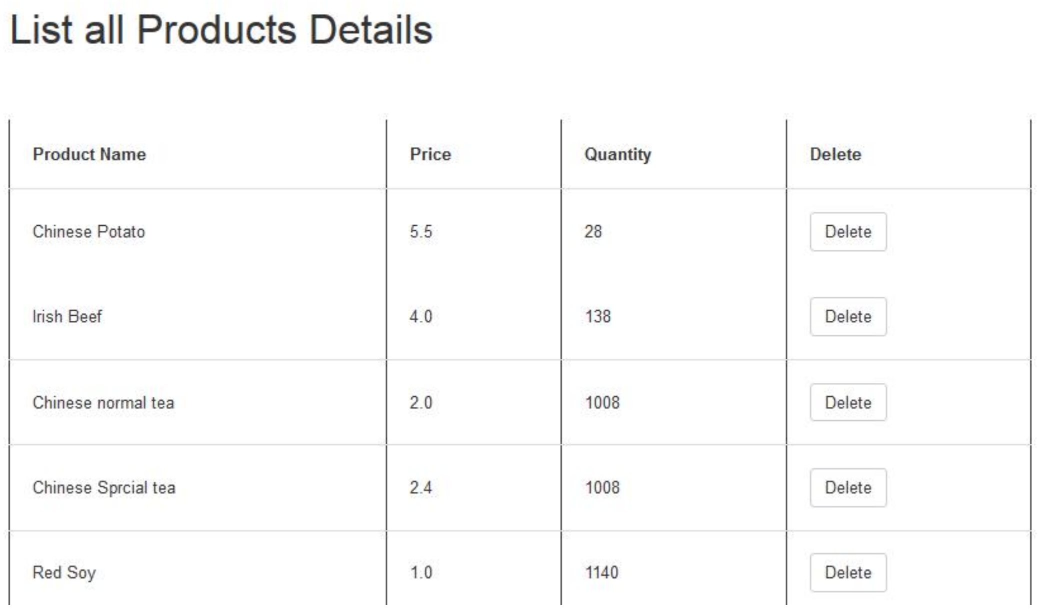
appleis in the database now, please click 'go to the update' for updating the quantity.

Fig 12

Delete a product

Administer could also delete a product from database. In this page, each line contains a product details with a Delete button. Click the button and the information of Item will be removed from database.

(As Fig 13 shows)



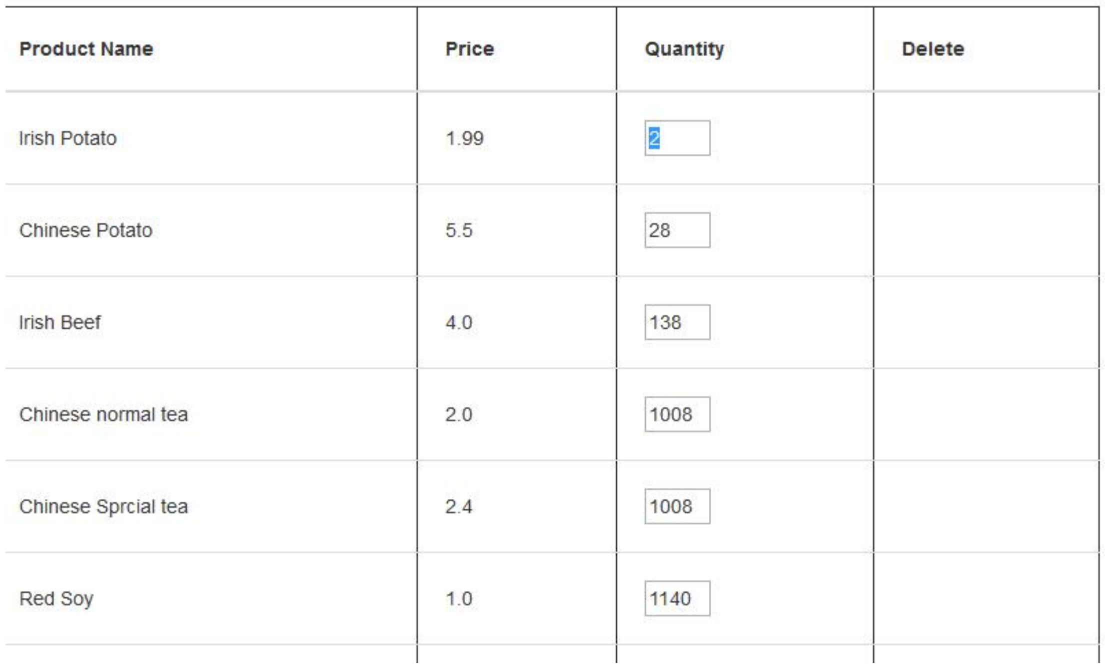
List all Products Details

Product Name	Price	Quantity	Delete
Chinese Potato	5.5	28	Delete
Irish Beef	4.0	138	Delete
Chinese normal tea	2.0	1008	Delete
Chinese Sprcial tea	2.4	1008	Delete
Red Soy	1.0	1140	Delete

Fig 13

Edit the Product Quantity

Administer can also edit the products' details. (Fig 14)

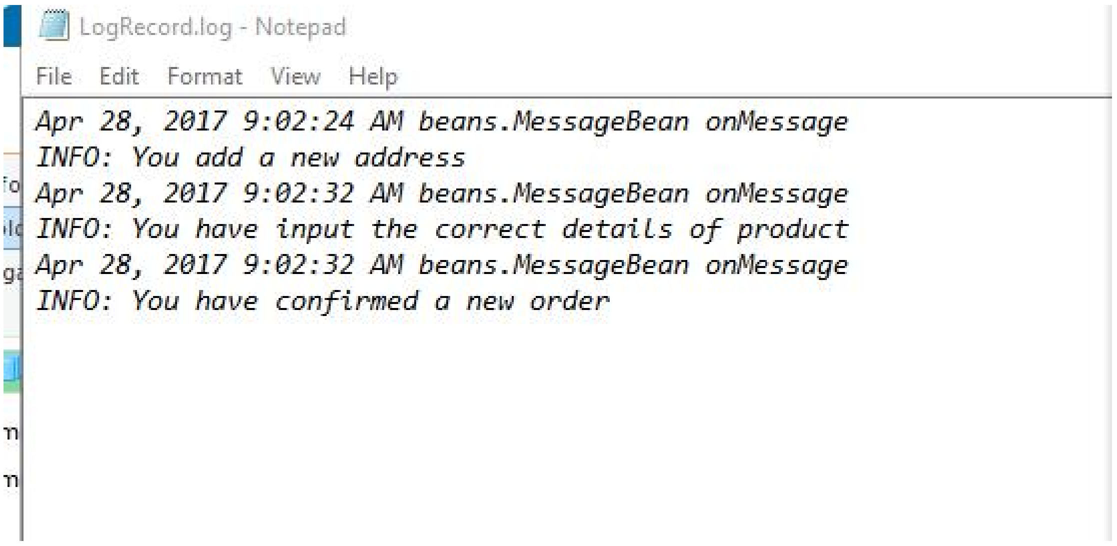


Product Name	Price	Quantity	Delete
Irish Potato	1.99	2	
Chinese Potato	5.5	28	
Irish Beef	4.0	138	
Chinese normal tea	2.0	1008	
Chinese Sprcial tea	2.4	1008	
Red Soy	1.0	1140	

Fig 14

logging facility (Message driven beans)

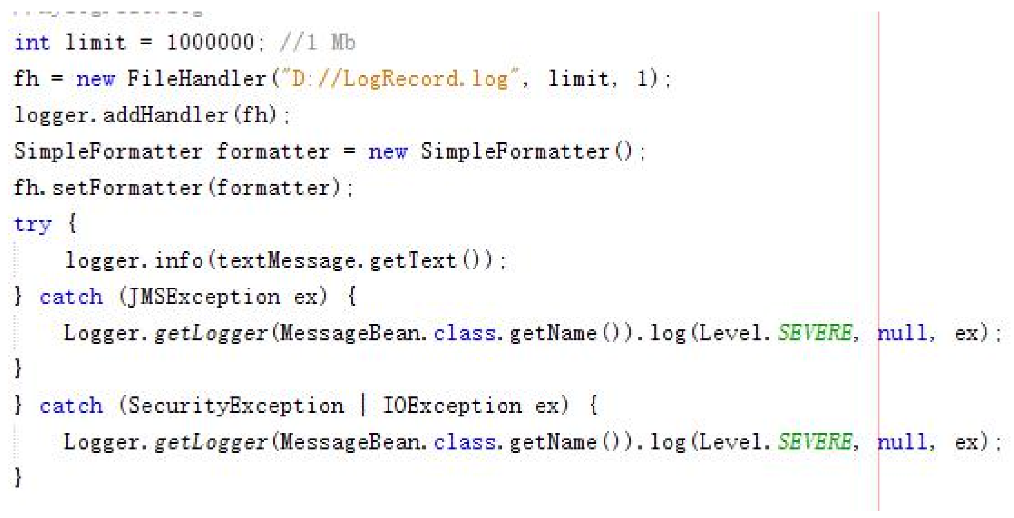
Every time a customer confirms an order or cancels an order and add a new address, a log record is added to the log file. And it is same to administrator, when he/she adds/removes a product an entry is also added to the log. (Fig 15)



```
Apr 28, 2017 9:02:24 AM beans.MessageBean onMessage
INFO: You add a new address
Apr 28, 2017 9:02:32 AM beans.MessageBean onMessage
INFO: You have input the correct details of product
Apr 28, 2017 9:02:32 AM beans.MessageBean onMessage
INFO: You have confirmed a new order
```

Fig 15

As Code shows(Fig 16), every time when Message driven beans ‘s method: onMessage( ) received a new message, it will write a new log record into a file named LogRecord.log. This log file was set a log file whose capacity is 1 Mb. Every time when it fulls with records, it will delete by itself.



```
int limit = 1000000; //1 Mb
fh = new FileHandler("D://LogRecord.log", limit, 1);
logger.addHandler(fh);
SimpleFormatter formatter = new SimpleFormatter();
fh.setFormatter(formatter);
try {
    logger.info(textMessage.getText());
} catch (JMSException ex) {
    Logger.getLogger(MessageBean.class.getName()).log(Level.SEVERE, null, ex);
}
} catch (SecurityException | IOException ex) {
    Logger.getLogger(MessageBean.class.getName()).log(Level.SEVERE, null, ex);
}
```

Fig 16

OWASP Top 10 Implementation

Techniques：

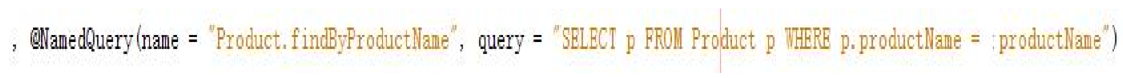
Query parameterization

For prevent web site from SQL Injection we use query parameterization. SQL Injection refers to an injection attack wherein an attacker can execute malicious SQL statements that control a web application ’ s database server. In order to run malicious SQL queries against a database server, an attacker must first find an input within the web application that is included inside of an SQL query. In order for an SQL Injection attack to take place, the vulnerable website needs to directly include user input within an SQL statement. An attacker can then insert a payload that will be included as part of the SQL query and run against the database server. Why we have to prevent the web site from SQL Injection? Because SQL can be used to access, modify and delete data. Furthermore, in specific cases, an RDBMS(Relational Database

Management System) could also run commands on the operating system from an SQL statement. As the result, an attacker can use SQL Injection to bypass authentication or even impersonate specific users, etc.

What is parameterized query? A parameterized query is a means of pre-compiling a SQL statement so that all you need to supply are the "parameters" that need to be inserted into the statement for it to be executed.

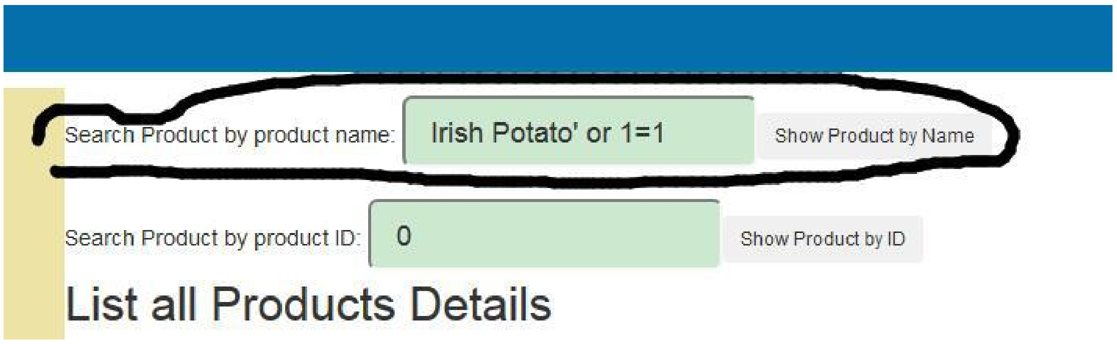
For example, this a parameterized query for select product by product name below:



```
, @NamedQuery(name = "Product.findByProductName", query = "SELECT p FROM Product p WHERE p.productName = :productName")
```

How does the parameterized query prevent the web site? Parameterized queries do proper substitution of arguments prior to running the SQL query. It completely removes the possibility of "dirty" input changing the meaning of your query. That is, if the input contains SQL, it can't become part of what is executed because the SQL is never injected into the resulting statement.

For example, let’s see if the attacker want to know all the information of product by product name, the attacker may enter a malicious input like ‘Irish Potato or 1 = 1’ (Fig 17):


Search Product by product name: Irish Potato' or 1=1 Show Product by Name

Search Product by product ID: 0 Show Product by ID

List all Products Details

Fig 17

If we have not use parameterized query, this malicious input will change the meaning of the relative SQL query, because ‘1 = 1’ will be another condition and it always be true. After we use parameterized query, the attacker will not get all the information, because ‘1 = 1’ will be a part of the product name and pass. This is the page source for this input:(Fig 18)

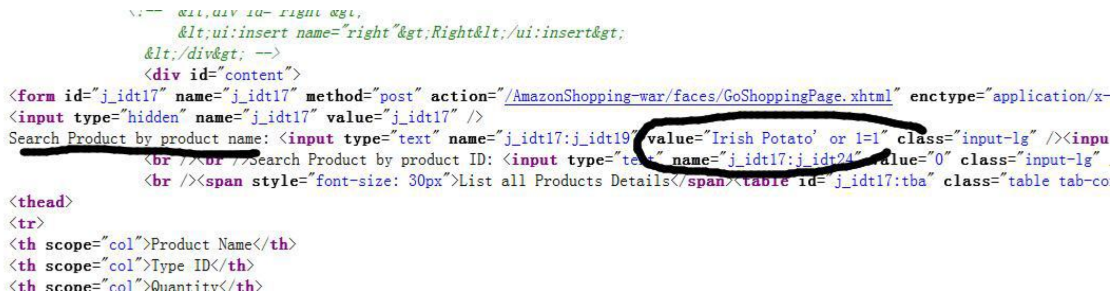

```
            <!-- &lt;div id="right"&gt;
                &lt;ui:insert name="right"&gt;Right&lt;/ui:insert&gt;
              &lt;/div&gt; -->
            <div id="content">
<form id="j_idt17" name="j_idt17" method="post" action="/AmazonShopping-war/faces/GoShoppingPage.xhtml" enctype="application/x-
<input type="hidden" name="j_idt17" value="j_idt17" />
Search Product by product name: <input type="text" name="j_idt17:j_idt19" value="Irish Potato' or 1=1" class="input-lg" /><inpu
            <br /><br />Search Product by product ID: <input type="text" name="j_idt17:j_idt24" value="0" class="input-lg"
            <br /><span style="font-size: 30px">List all Products Details</span><table id="j_idt17:tba" class="table tab-co
<thead>
<tr>
<th scope="col">Product Name</th>
<th scope="col">Type ID</th>
<th scope="col">Quantity</th>
```

Fig 18

The attacker will finally get the result below:

(Fig 19)

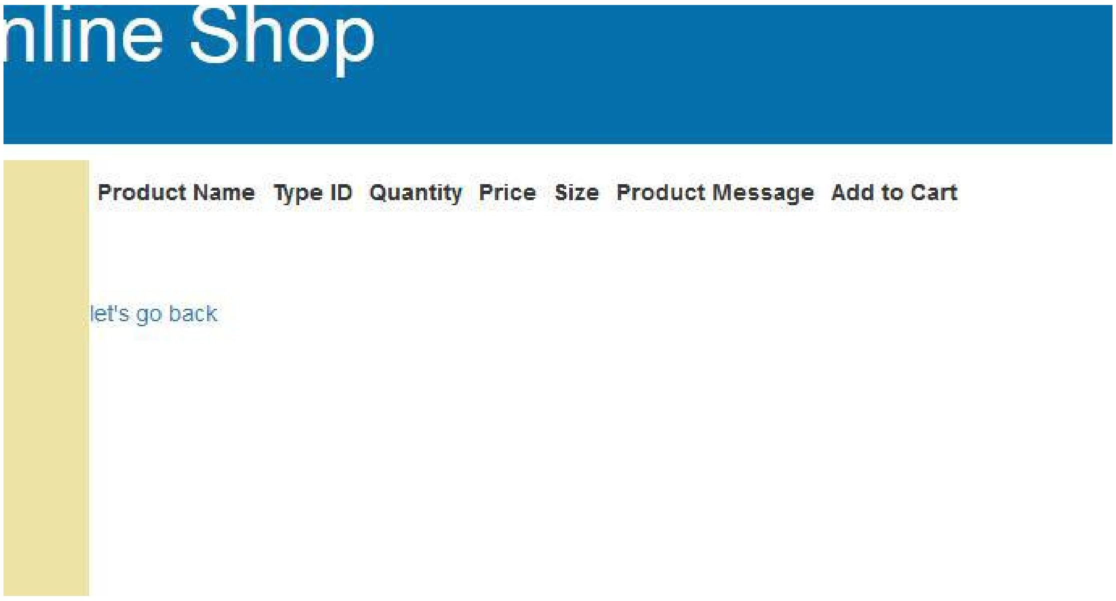
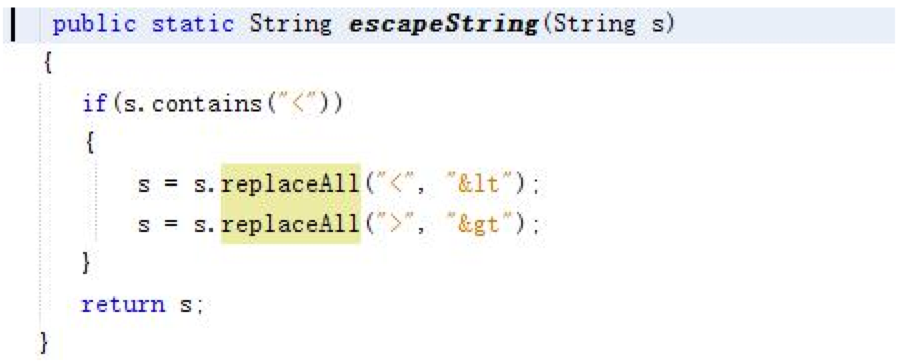


Fig 19

Filter

We use filter in java to prevent the website from Cross Site Script(XSS). Cross-site scripting attack is a kind of attack on web applications in which attackers try to inject malicious scripts to perform malicious actions on trusted websites. In order to run malicious JavaScript code in a victim’s browser, an attacker must first find a way to inject a payload into a web page that the victim visits. Of course, an attacker could use social engineering techniques to convince a user to visit a vulnerable page with an injected JavaScript payload. In order for an XSS attack to take place the vulnerable website needs to directly include user input in its pages. An attacker can then insert a string that will be used within the web page and treated as code by the victim’s browser.

For avoid XSS, we firstly encode every data that is given by a user and use filter to check if the request from user is valid. If data is not given by a user but supplied via the GET parameter, encode these data too. Even a POST form can contain XSS vectors. We encode all “<”, “>” which enter by user to "&lt" or "&gt". (Fig 20)

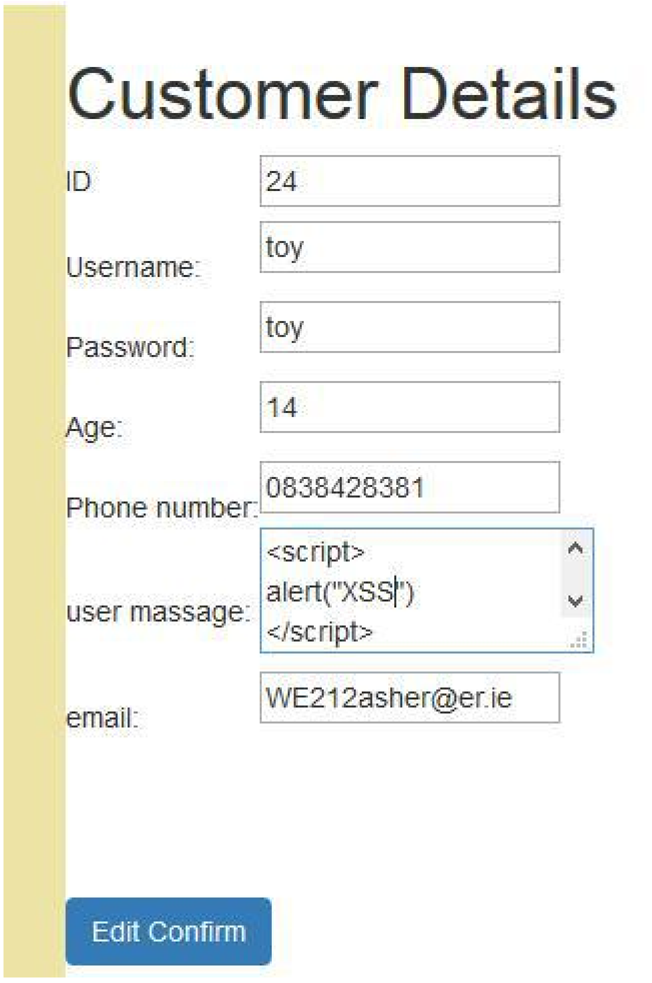


```
public static String escapeString(String s)
{
    if(s.contains("<"))
    {
        s = s.replaceAll("<", "&lt");
        s = s.replaceAll(">", "&gt");
    }
    return s;
}
```

Fig 20

The purpose of encoding is to convert untrusted input into a safe form where the input is displayed as data to the user without executing as code in the browser. Therefore, when the attacker want to add a script language into our website, the script command will be encoding to a normal input:

For example, let’s see if the attacker enter a script language like “<script>alert(“XSS”)</script>”: (Fig 21)



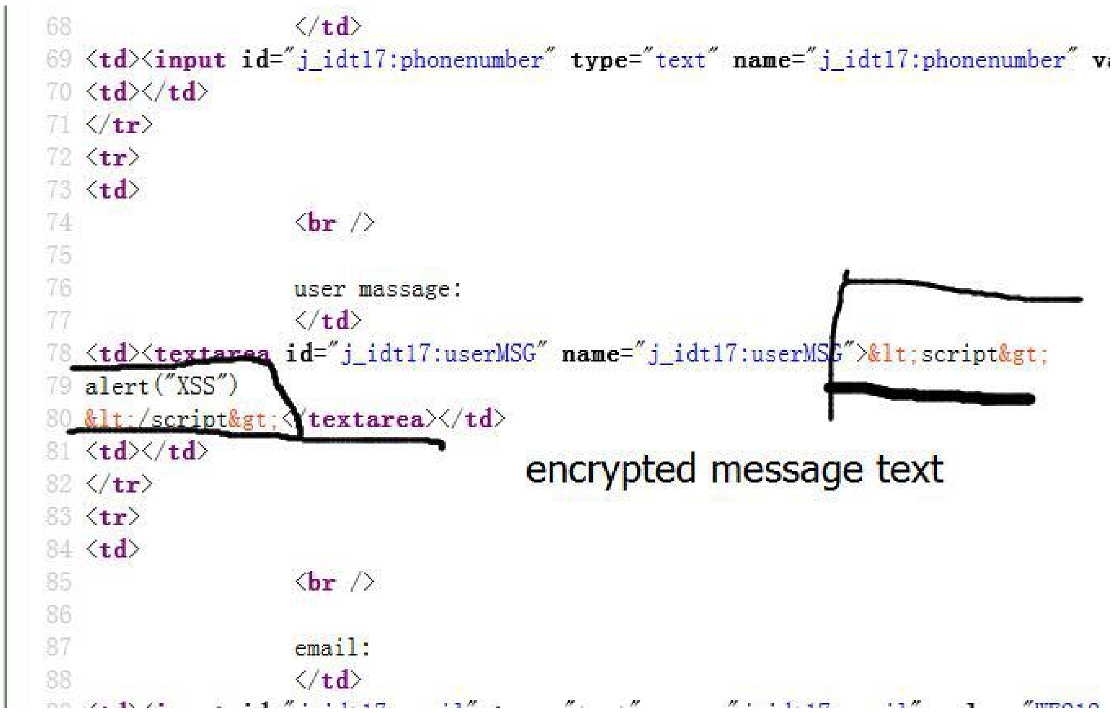
Customer Details

ID	24
Username:	toy
Password:	toy
Age:	14
Phone number:	0838428381
user massage:	<script> alert("XSS") </script>
email:	WE212asher@er.ie

Edit Confirm

Fig 21

The value passed will be the value in the picture below actually:(Fig 22)



```
                </td>
<td><input id="j_idt17:phonenumber" type="text" name="j_idt17:phonenumber" v
<td></td>
</tr>
<tr>
<td>
                <br />

                user massage:
                </td>
<td><textarea id="j_idt17:userMSG" name="j_idt17:userMSG">&lt;script&gt;
alert("XSS")
&lt;/script&gt;</textarea></td>
<td></td>
</tr>
<tr>
<td>
                <br />

                email:
                </td>
```

Fig 22

Token

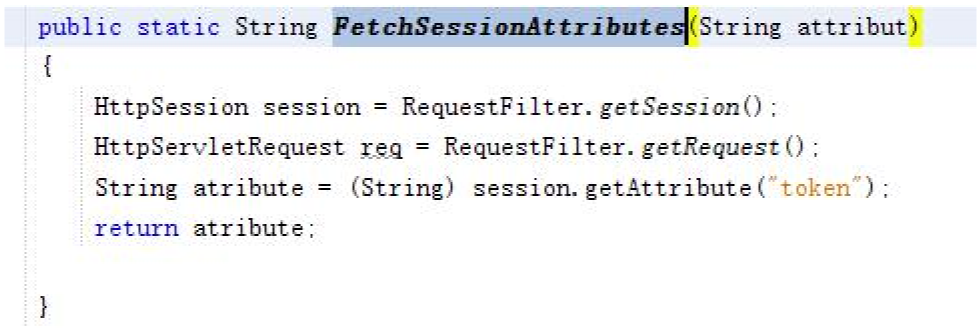
We use token to avoid Cross Site Request Forgery(CSRF). CSRF attacks include a malicious exploit of a website in which a user will transmit malicious requests that the target website trusts without the user’s consent. In Cross-Site Scripting (XSS), the attacker exploits the trust a user has for a website, with CSRF on the other hand, the attacker exploits the trust a website has against a user’s browser. Basically, an attacker will use CSRF to trick a victim into accessing a website or clicking a URL link that contains malicious or unauthorized requests. It is called ‘malicious’ since the CSRF attack will use the identity and privileges of the victim and impersonate them in order to perform any actions desired by the attacker

A prevention measure could be the implementation and inclusion of tokens in a user’s (current) session. Tokens are long cryptographic values that are difficult to guess. These will be generated when a user’s session begins and will be associated with this particular user’ s session. This challenge token will be included in each request, which will be used by the server side to verify the legitimacy of the end-user’ s request.

In order for an attacker to forge a HTTP request, they would have to know the particular challenge value (token) of the victim’ s session. The disclosure of the challenge token in the URL (GET requests) should be done wisely and with awareness of the CSRF attack.

Moreover, tokens can be used in the submission of double cookies. The server-side will generate a strong random value which will be included in the submitted cookie on the user’ s machine. This will act as the session ID. On sending a POST request, the website will require the particular session ID to be included as a hidden value in the submission form and be included in the cookie as well. If the two values are the same, the POST request will be considered as valid and submitted successfully. Therefore, even if the attacker is able to include any value in the form, based on the same-origin policy, the attacker will not be able to retrieve or modify the token value in the cookie and launch a CSRF attack unless they manage to guess the session ID value.

In our website, on sending a request, the website will get the token by executing the FetchSessionAttributes(): (Fig 23)



```
public static String FetchSessionAttributes(String attribut)
{
    HttpSession session = RequestFilter.getSession();
    HttpServletRequest req = RequestFilter.getRequest();
    String atribute = (String) session.getAttribute("token");
    return atribute;

}
```

Fig 23

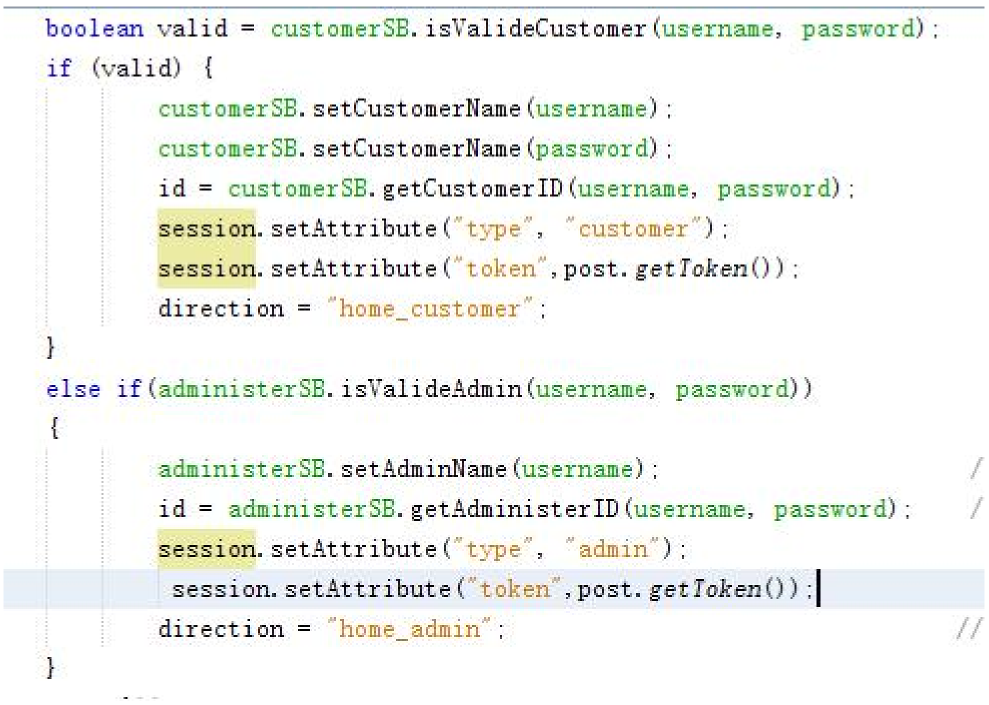
Then the filter will check if the attribute which be returned is null or is relative number. If the token is null or is not match the relative number, that means this request is from the attacker. Then the server will not dispose this request.

Token and Filter:

We also use filter and token to avoid Missing Function Level Access Control. Virtually all web applications verify function level access rights before making that functionality visible in the UI. However, applications need to perform the same access control checks on the server when each function is accessed. If requests are not verified, attackers will be able to forge requests in order to access unauthorized functionality.

We firstly give token to user randomly when they login to the website.

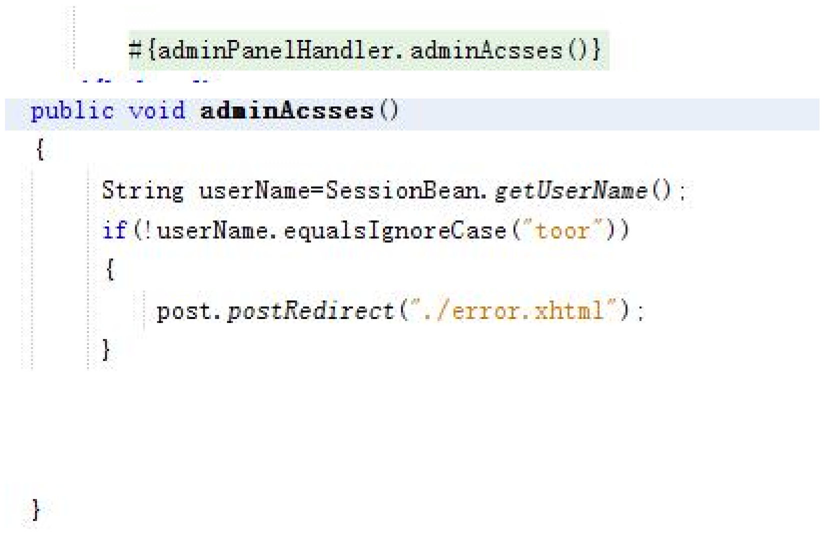
(Fig 24)



```
boolean valid = customerSB.isValideCustomer(username, password);
if (valid) {
        customerSB.setCustomerName(username);
        customerSB.setCustomerName(password);
        id = customerSB.getCustomerID(username, password);
        session.setAttribute("type", "customer");
        session.setAttribute("token",post.getToken());
        direction = "home_customer";
}
else if(administerSB.isValideAdmin(username, password))
{
        administerSB.setAdminName(username);                    /
        id = administerSB.getAdministerID(username, password);    /
        session.setAttribute("type", "admin");
         session.setAttribute("token",post.getToken());
        direction = "home_admin";                                //
}
```

Fig 24

When the user want to access the page directly, we get the user name from session:(Fig 25)

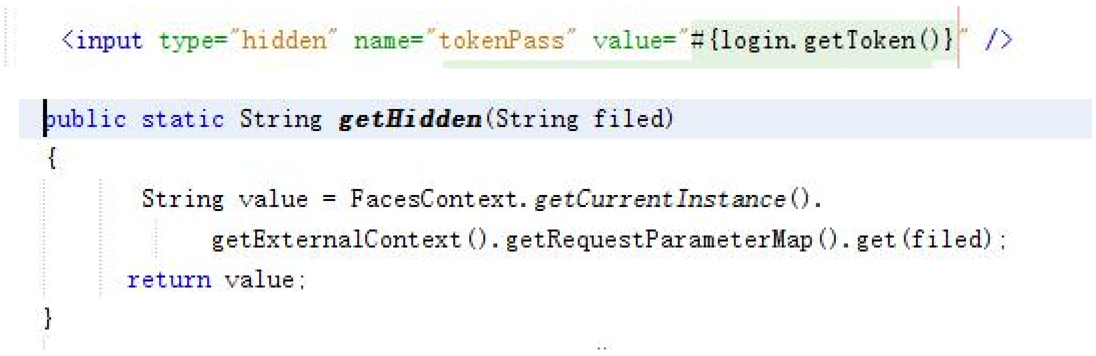


```
        #{adminPanelHandler.adminAcsses()}
public void adminAcsses()
{
    String userName=SessionBean.getUserName();
    if(!userName.equalsIgnoreCase("toor"))
    {
        post.postRedirect("./error.xhtml");
    }



}
```

Fig 25

To check if it has authority to access the page.

When user want to do some action which has authority, the filter will get the token from the hidden form in html. (Fig 26).

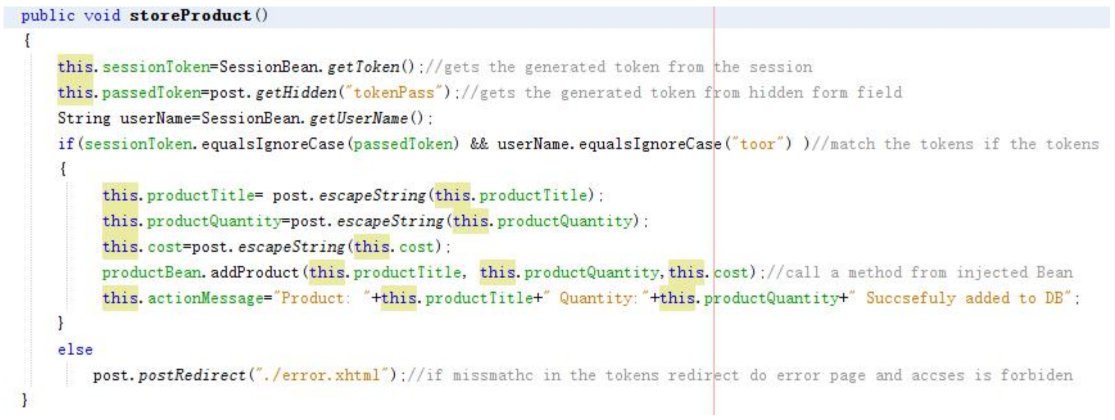


```
<input type="hidden" name="tokenPass" value="#{login.getToken()}" />
```

```
public static String getHidden(String filed)
{
      String value = FacesContext.getCurrentInstance().
          getExternalContext().getRequestParameterMap().get(filed);
    return value;
}
```

Fig 26

Every time the user want to do the action which can only do by administer, the filter will check the token if it is same with the token of administer. For example, when the customer want to add a product: (Fig 27)



```
public void storeProduct()
{
    this.sessionToken=SessionBean.getToken();//gets the generated token from the session
    this.passedToken=post.getHidden("tokenPass");//gets the generated token from hidden form field
    String userName=SessionBean.getUserName();
    if(sessionToken.equalsIgnoreCase(passedToken) && userName.equalsIgnoreCase("toor") )//match the tokens if the tokens
    {
        this.productTitle= post.escapeString(this.productTitle);
        this.productQuantity=post.escapeString(this.productQuantity);
        this.cost=post.escapeString(this.cost);
        productBean.addProduct(this.productTitle, this.productQuantity,this.cost);//call a method from injected Bean
        this.actionMessage="Product: "+this.productTitle+" Quantity:"+this.productQuantity+" Succsefuly added to DB";
    }
    else
       post.postRedirect("./error.xhtml");//if missmathc in the tokens redirect do error page and accses is forbiden
}
```

Fig 27

The administer Panel Handler will check if the token and user name is match with administer. If it is match, then do add product, else go to the error page.

Encoding and Decoding

This web application use Session for storaging the data from the client, generally it will store the id , username , etc.

It is a not just very clever action than store the password into session directly, it needs to be encrypted anyway.

The encoding algorithm used in this project for encrypt password is that, generate the byte type variable of the passsword string and make the exclusive or operation for the binary data, and they, reverse it into string as the encrypted data for storage in session.

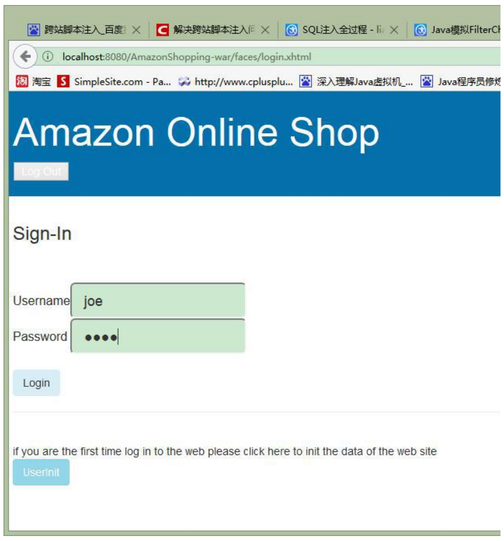
```
//Encoding :
        public static String encode(String enc){
                byte[] b = enc.getBytes(charset);
                for(int i=0,size=b.length;i<size;i++){
                        for(byte keyBytes0:keyBytes){
                                b[i] = (byte) (b[i]^keyBytes0);
                        }
                }
                return new String(b);
        }

//decoding
  public static String decode(String dec){
                byte[] e = dec.getBytes(charset);
                byte[] dee = e;
                for(int i=0,size=e.length;i<size;i++){
                        for(byte keyBytes0:keyBytes){
                                e[i] = (byte) (dee[i]^keyBytes0);
                        }
                }
                return new String(e);
        }
```

Test Application

Details about error hint and defence

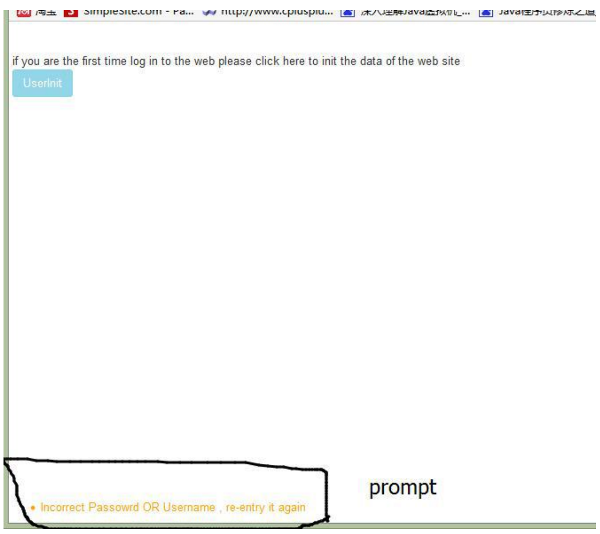
1. Log in

When users or administer log in their home page, a hint will be appeared if their password is incorrect. (Fig 28.1)



if you are the first time log in to the web please click here to init the data of the web site

Fig 28.1



if you are the first time log in to the web please click here to init the data of the web site

• Incorrect Passowrd OR Username , re-entry it again

Fig 28.2

If the user name and user password is matched, user will access their home page successfully.

2. Add New Product

When administer register a new product, a hint will be appeared if the necessary product information (product Name, type ID, product price, product size)is null and the format of some product information(product Name, product price) is incorrect. (Fig 29)

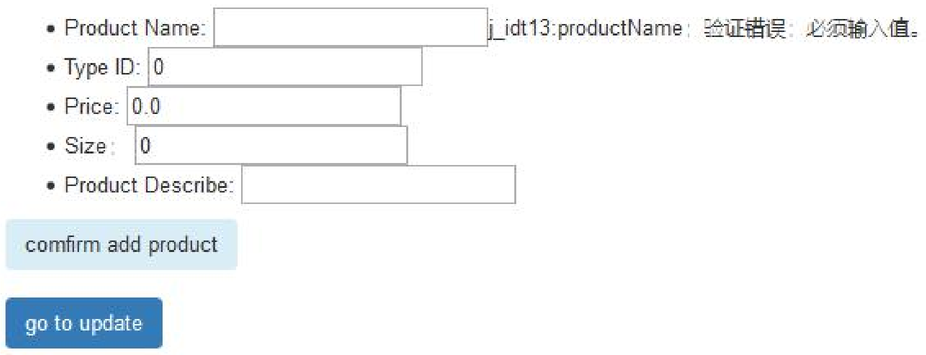
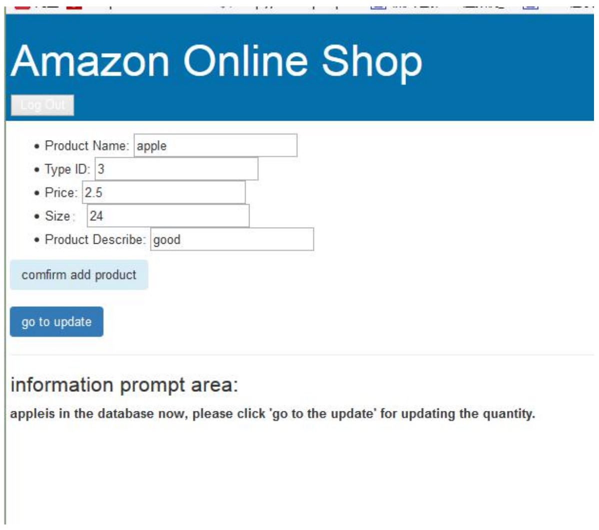

- Product Name: j_idt13:productName：验证错误：必须输入值。
- Type ID: 0
- Price: 0.0
- Size: 0
- Product Describe:

Fig 29

If all information of the product is correct, this product will be added into database successfully. (Fig 30)



- Product Name: apple
- Type ID: 3
- Price: 2.5
- Size: 24
- Product Describe: good

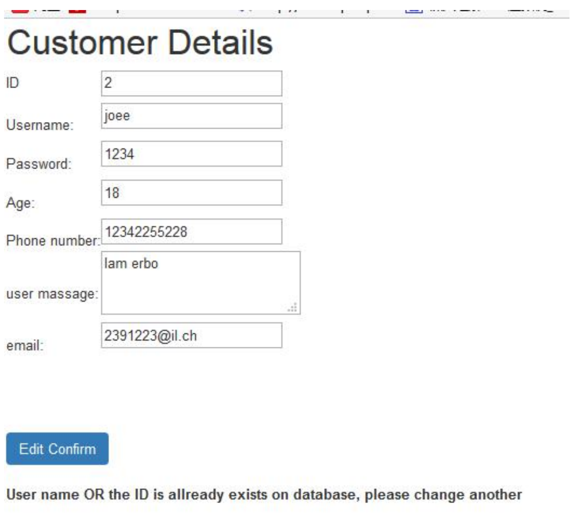
information prompt area:

appleis in the database now, please click 'go to the update' for updating the quantity.

Fig 30

3. Edit A Customer Profile

When user edit their profile, a hint will be appeared if the ID number and the User Name that user want to changed is existing in database.(Fig 31)

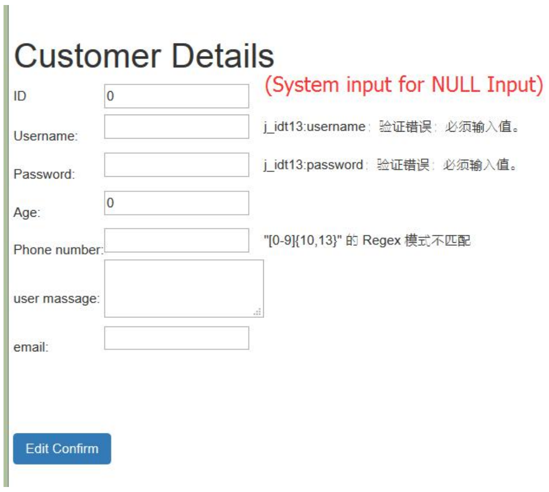

Customer Details

ID	2
Username:	joee
Password:	1234
Age:	18
Phone number:	12342255228
user massage:	lam erbo
email:	2391223@il.ch

User name OR the ID is allready exists on database, please change another

Fig 31

When user edit their profile, a hint will be appeared if the necessary user information (user ID, user Name, password, user Age, phone number) is null and the format of some user information(Age,phone number) is incorrect.(Fig 32)

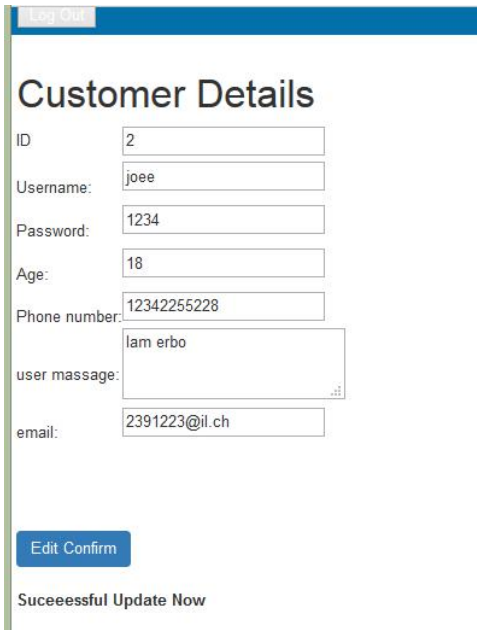

Customer Details

(System input for NULL Input)

ID	0	
Username:		j_idt13:username：验证错误：必须输入值。
Password:		j_idt13:password：验证错误：必须输入值。
Age:	0	
Phone number:		"[0-9]{10,13}" 的 Regex 模式不匹配
user massage:		
email:		

Fig 32

If the user information they changed is correct, their information will be modified successfully. (Fig 33)



Log Out

Customer Details

ID	2
Username:	joee
Password:	1234
Age:	18
Phone number:	12342255228
user massage:	lam erbo
email:	2391223@il.ch

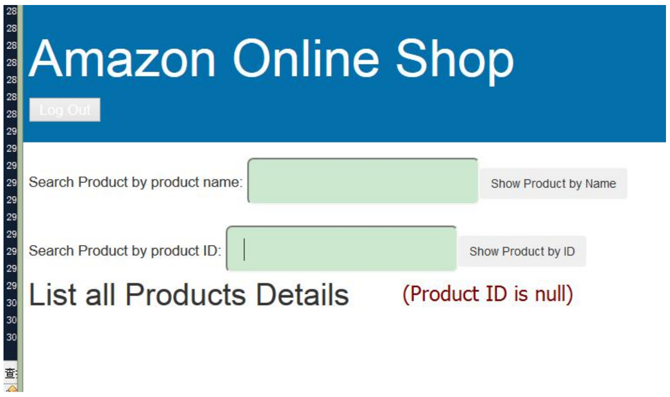
Edit Confirm

Suceeessful Update Now

Fig 33

4. Search Product by ID and Name

When the user search a specific product, a hint will be appeared if the product ID that the user want to search is null or is incorrect. (Fig 34)



Amazon Online Shop

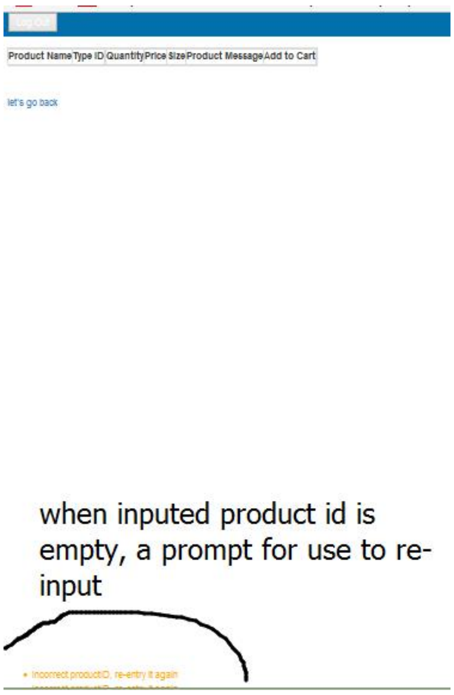
Log Out

Search Product by product name: Show Product by Name

Search Product by product ID: Show Product by ID

List all Products Details (Product ID is null)

Fig 34



Product Name Type ID Quantity Price Size Product Message Add to Cart

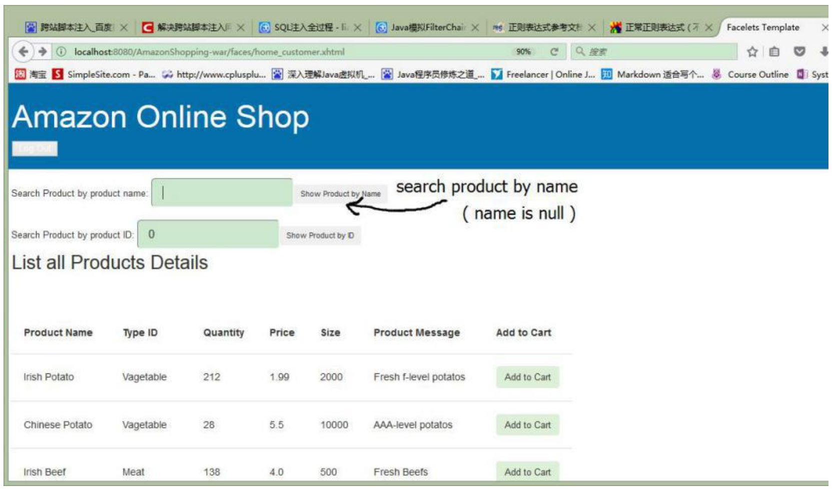
let's go back

when inputed product id is empty, a prompt for use to re-input

- Incorrect productID, re-entry it again

Fig 35

When the user search a specific product, a hint will be appeared if the product name that the user want to search is null or is incorrect.(Fig 36.1, Fig 36.2)



localhost:8080/AmazonShopping-war/faces/home_customer.xhtml

Amazon Online Shop

Search Product by product name: [Show Product by Name]

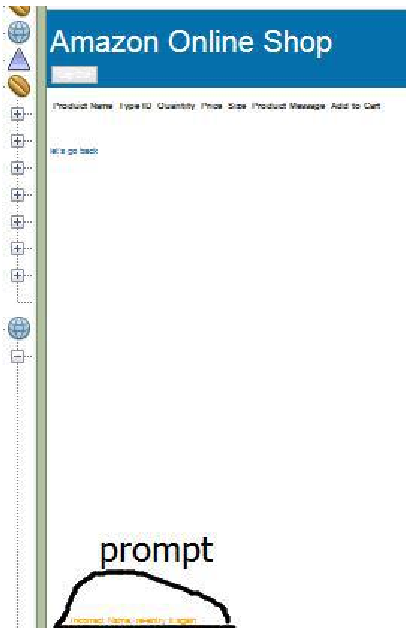
search product by name (name is null)

Search Product by product ID: 0 [Show Product by ID]

List all Products Details

Product Name	Type ID	Quantity	Price	Size	Product Message	Add to Cart
Irish Potato	Vagetable	212	1.99	2000	Fresh f-level potatos	Add to Cart
Chinese Potato	Vagetable	28	5.5	10000	AAA-level potatos	Add to Cart
Irish Beef	Meat	138	4.0	500	Fresh Beefs	Add to Cart

Fig 36.1



Amazon Online Shop

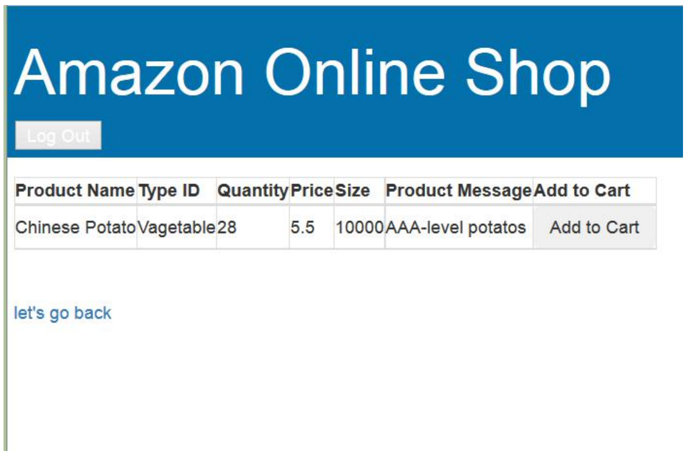
Product Name Type ID Quantity Price Size Product Message Add to Cart

let's go back

Incorrect Name, re-entry it again

Fig 36.2

If the product ID and product name is correct that user searched, the user will access the product information page successfully. (Fig 37)



Amazon Online Shop

Product Name	Type ID	Quantity	Price	Size	Product Message	Add to Cart
Chinese Potato	Vagetable	28	5.5	10000	AAA-level potatos	Add to Cart

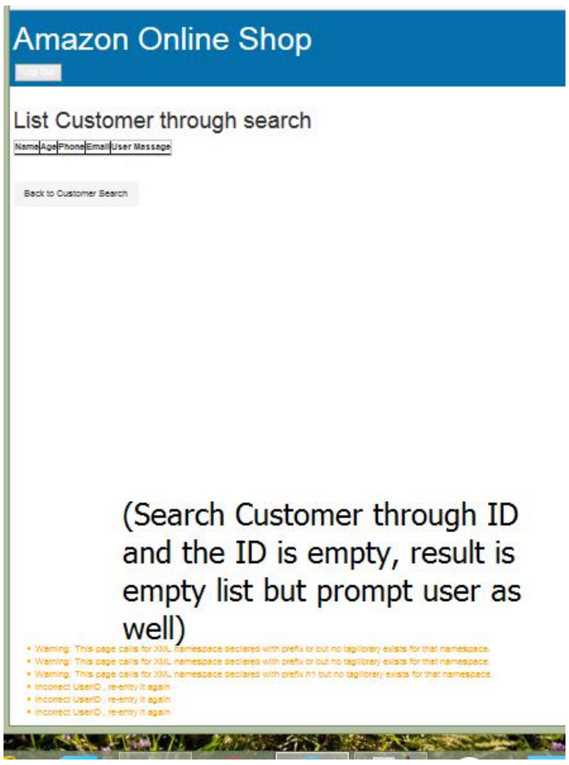
let's go back

Fig 37

Search User by ID and Name

When the user search a specific user , a hint will be appeared if the user

ID that the user want to search is null or is incorrect.(Fig 38)


Amazon Online Shop

List Customer through search

Name	Age	Phone	Email	User Massage

Back to Customer Search

(Search Customer through ID and the ID is empty, result is empty list but prompt user as well)

- Warning: This page calls for XML namespace declared with prefix br but no taglibrary exists for that namespace.
- Warning: This page calls for XML namespace declared with prefix br but no taglibrary exists for that namespace.
- Warning: This page calls for XML namespace declared with prefix h1 but no taglibrary exists for that namespace.
- Incorrect UserID , re-entry it again
- Incorrect UserID , re-entry it again
- Incorrect UserID , re-entry it again

Fig 38

When the user search a specific user, a hint will be appeared if the user name that the user want to search is null or is incorrect.(Fig 39.1, Fig 39.2)

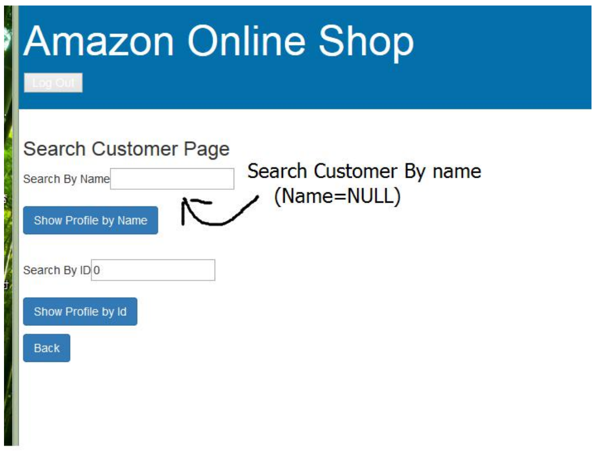
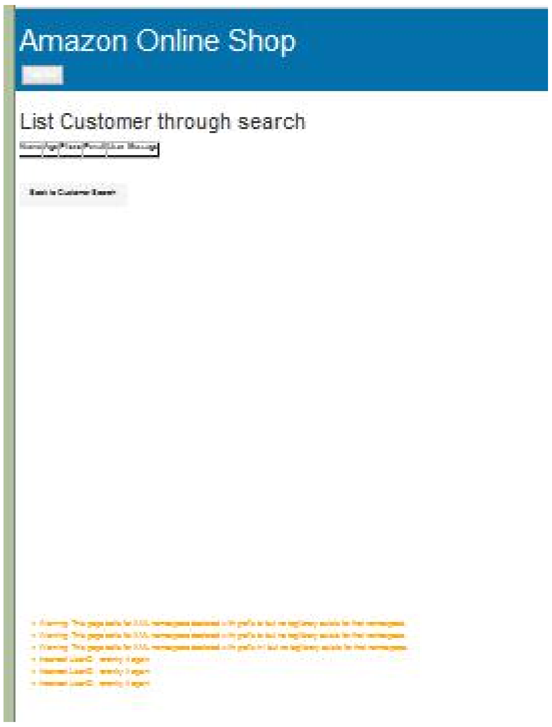


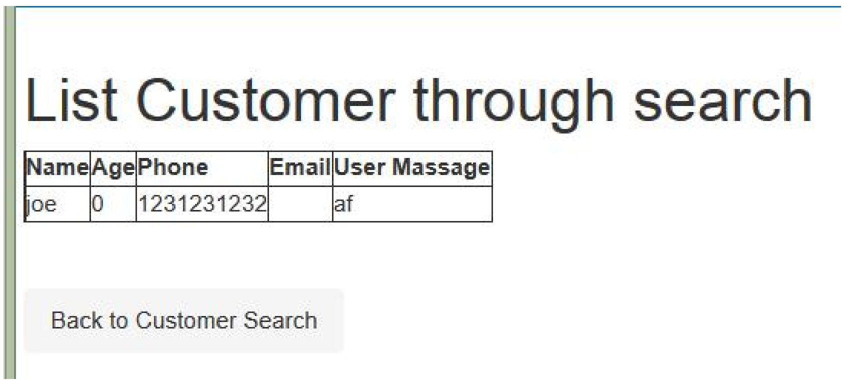
Fig 39.1



- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]

Fig 39.2

If the user ID and user name is correct that user searched, the user will access the ***use***r information page successfully. (Fig 40)



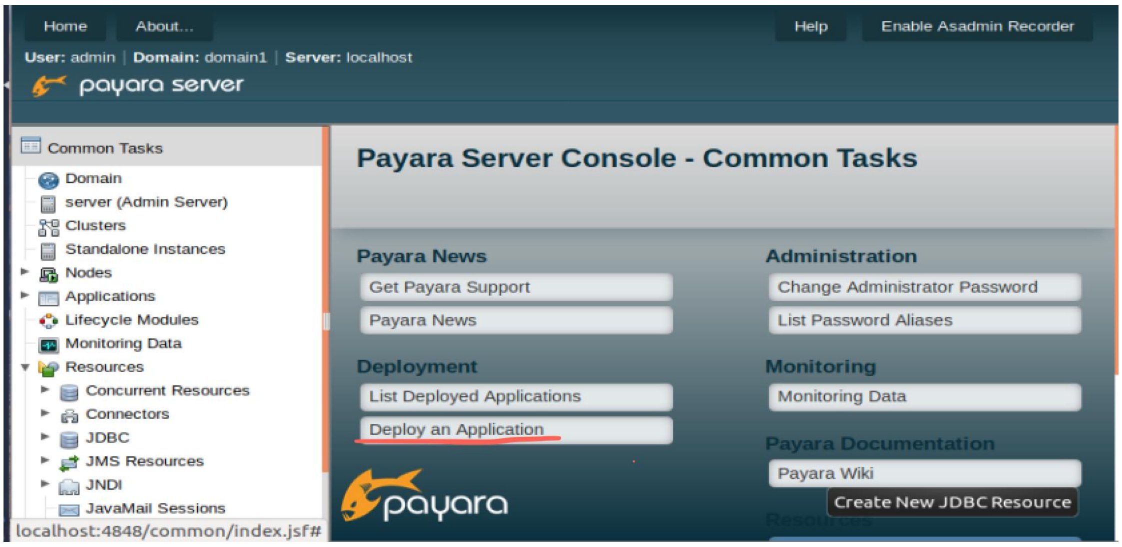
List Customer through search

Name	Age	Phone	Email	User Massage
joe	0	1231231232		af

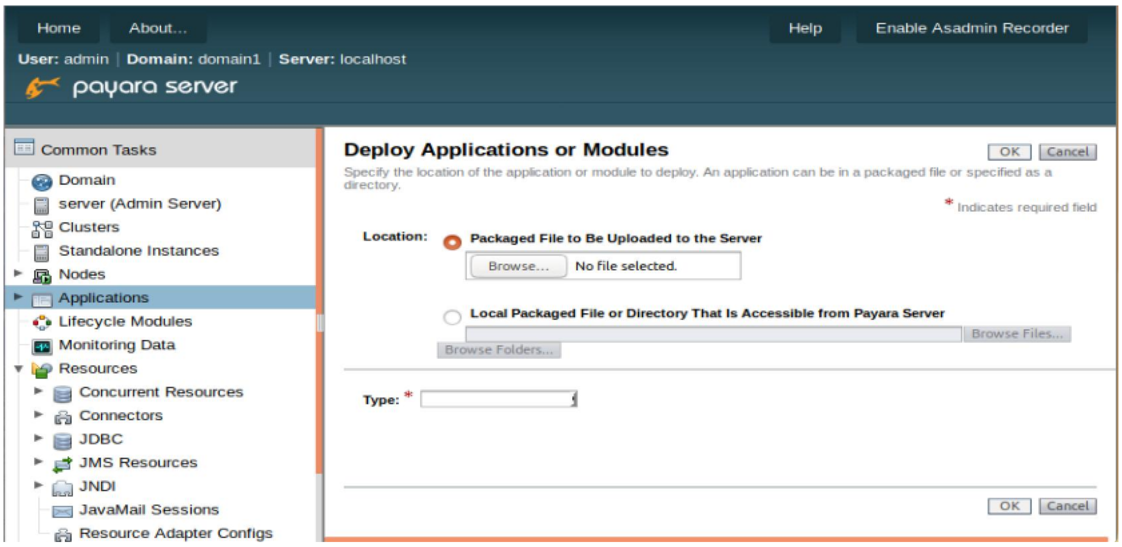
Fig 40

How to deploy application

Step 1： Open Admin Console and Clink the “Deploy an Application ”



Step 2: Select “Applications ”



Deploy Applications or Modules

Specify the location of the application or module to deploy. An application can be in a packaged file or specified as a directory.

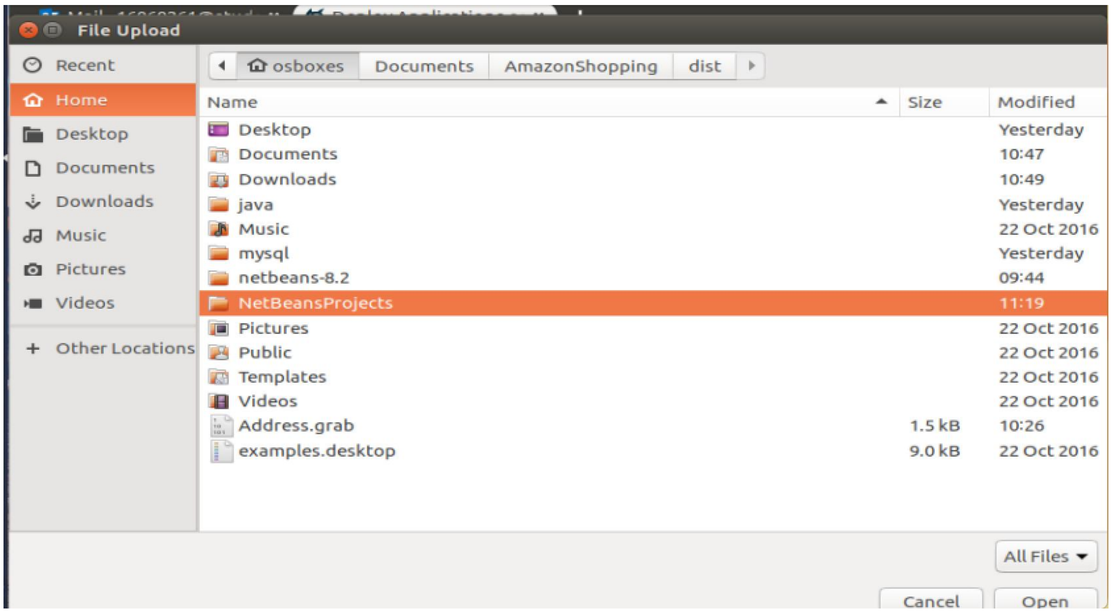
* Indicates required field

Location: [✓ Packaged File to Be Uploaded to the Server] No file selected.

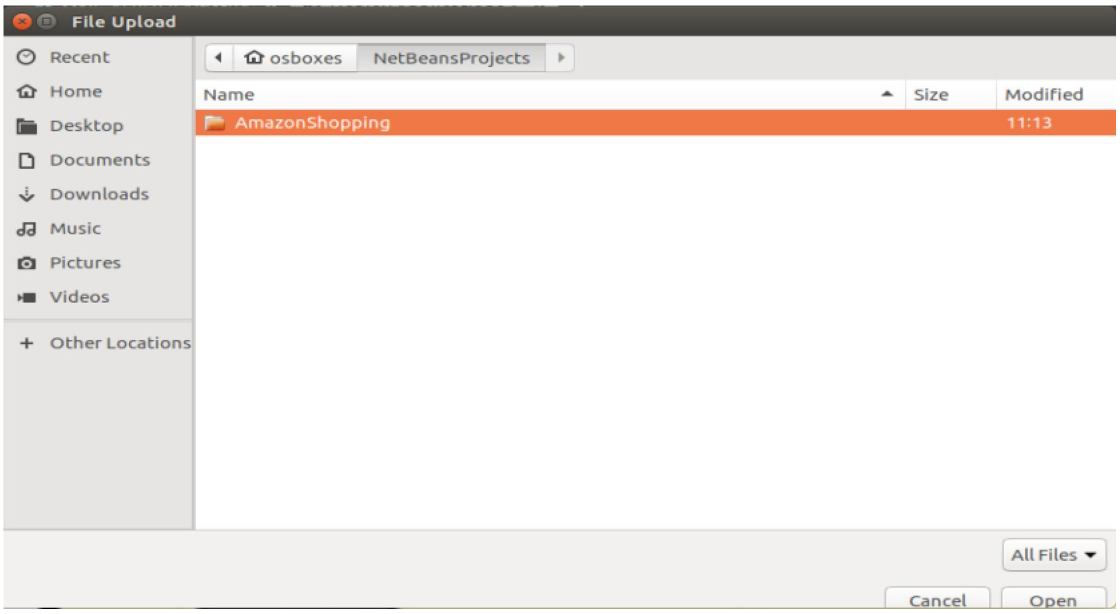
Local Packaged File or Directory That Is Accessible from Payara Server

Type: *

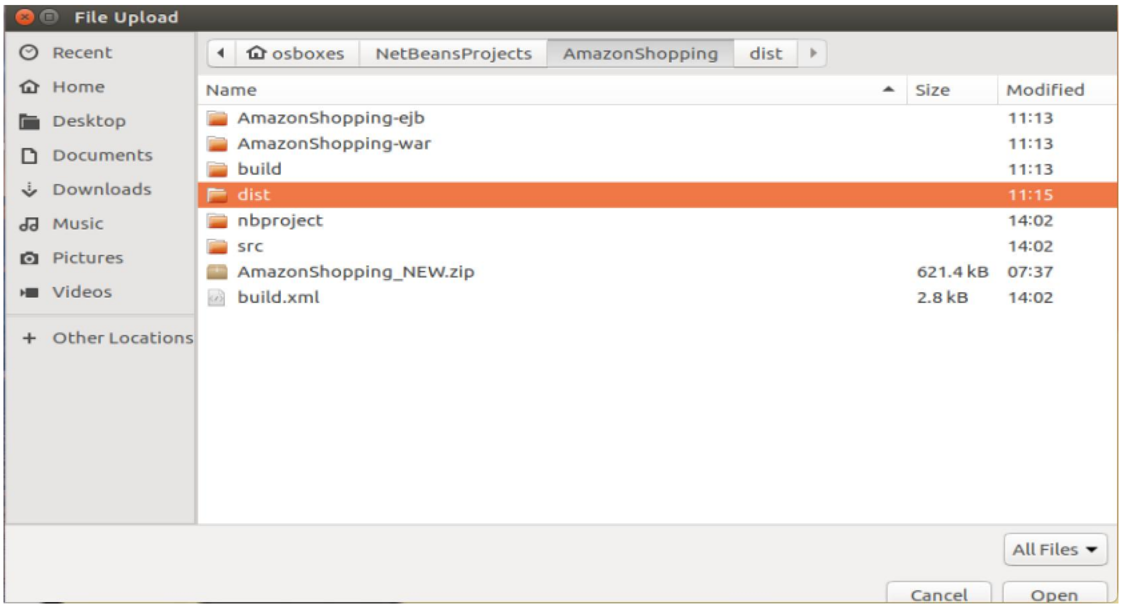
Step 3: find and open the .ear file



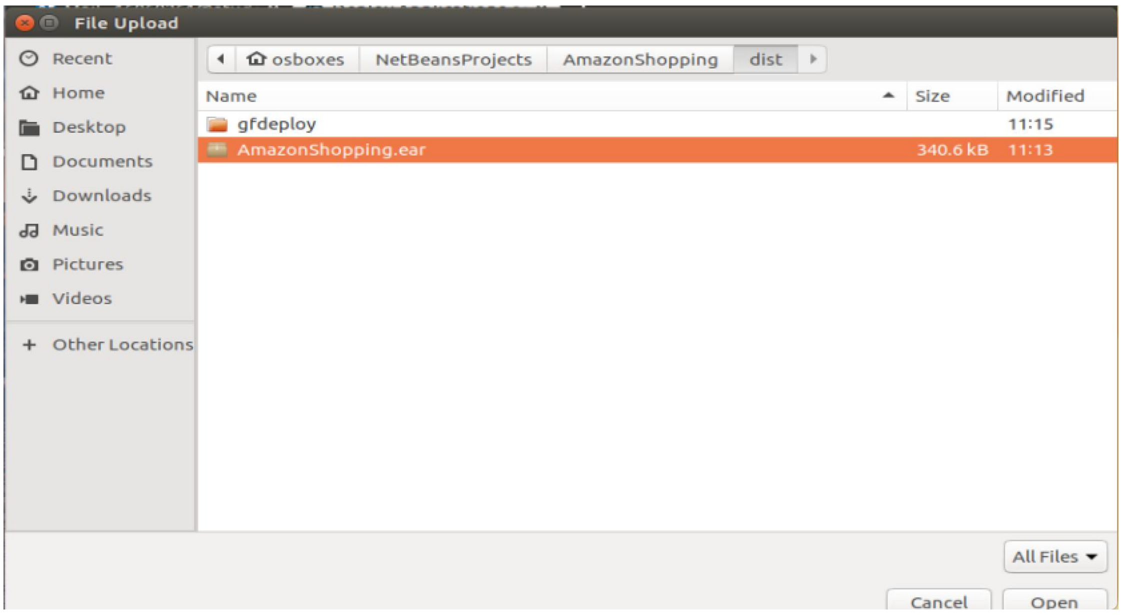
Name	Size	Modified
Desktop		Yesterday
Documents		10:47
Downloads		10:49
java		Yesterday
Music		22 Oct 2016
mysql		Yesterday
netbeans-8.2		09:44
NetBeansProjects		11:19
Pictures		22 Oct 2016
Public		22 Oct 2016
Templates		22 Oct 2016
Videos		22 Oct 2016
Address.grab	1.5 kB	10:26
examples.desktop	9.0 kB	22 Oct 2016



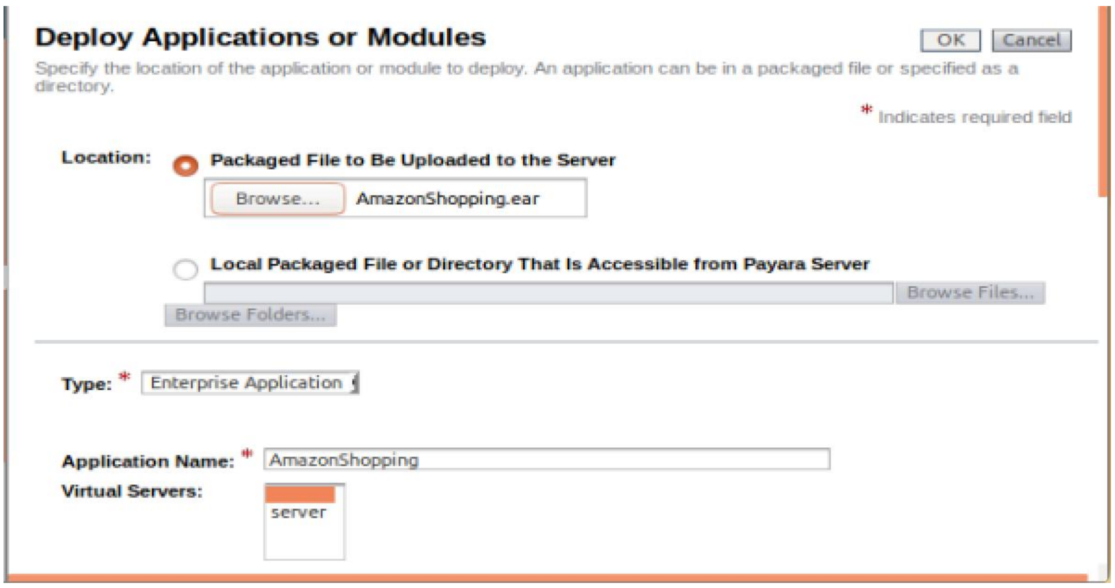
Name	Size	Modified
AmazonShopping		11:13



Name	Size	Modified
AmazonShopping-ejb		11:13
AmazonShopping-war		11:13
build		11:13
dist		11:15
nbproject		14:02
src		14:02
AmazonShopping_NEW.zip	621.4 kB	07:37
build.xml	2.8 kB	14:02



Name	Size	Modified
gfdeploy		11:15
AmazonShopping.ear	340.6 kB	11:13



Deploy Applications or Modules

Specify the location of the application or module to deploy. An application can be in a packaged file or specified as a directory.

* Indicates required field

Location: [✓ Packaged File to Be Uploaded to the Server] AmazonShopping.ear

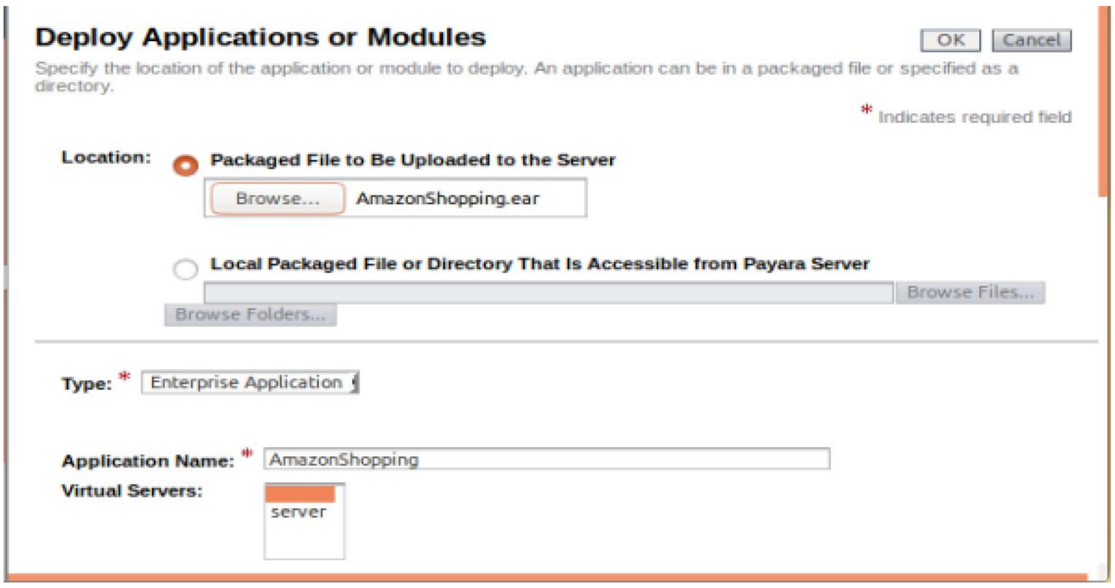
Local Packaged File or Directory That Is Accessible from Payara Server

Type: * Enterprise Application

Application Name: * AmazonShopping

Virtual Servers: server

Step 4: Clink the “OK ” button



Deploy Applications or Modules

Specify the location of the application or module to deploy. An application can be in a packaged file or specified as a directory.

* Indicates required field

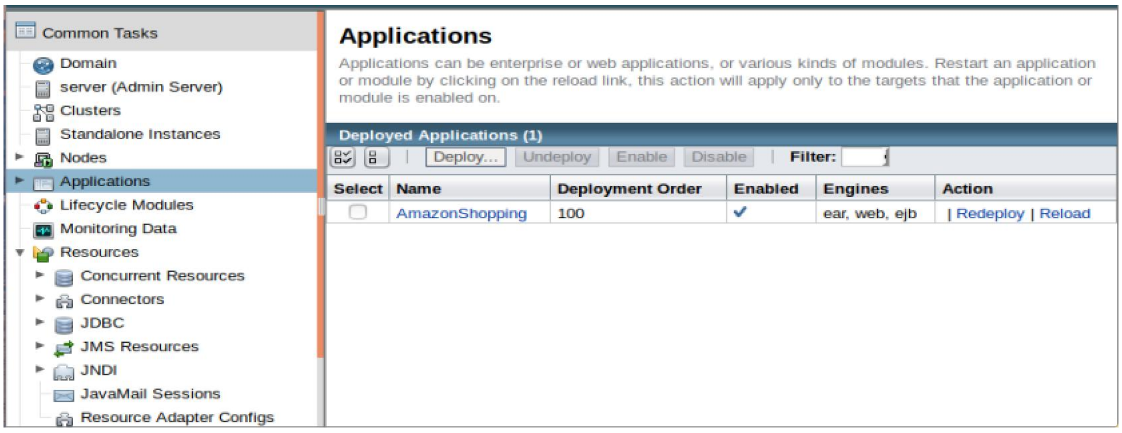
Location: [● Packaged File to Be Uploaded to the Server] AmazonShopping.ear

○ Local Packaged File or Directory That Is Accessible from Payara Server

Type: * Enterprise Application

Application Name: * AmazonShopping

Virtual Servers: server



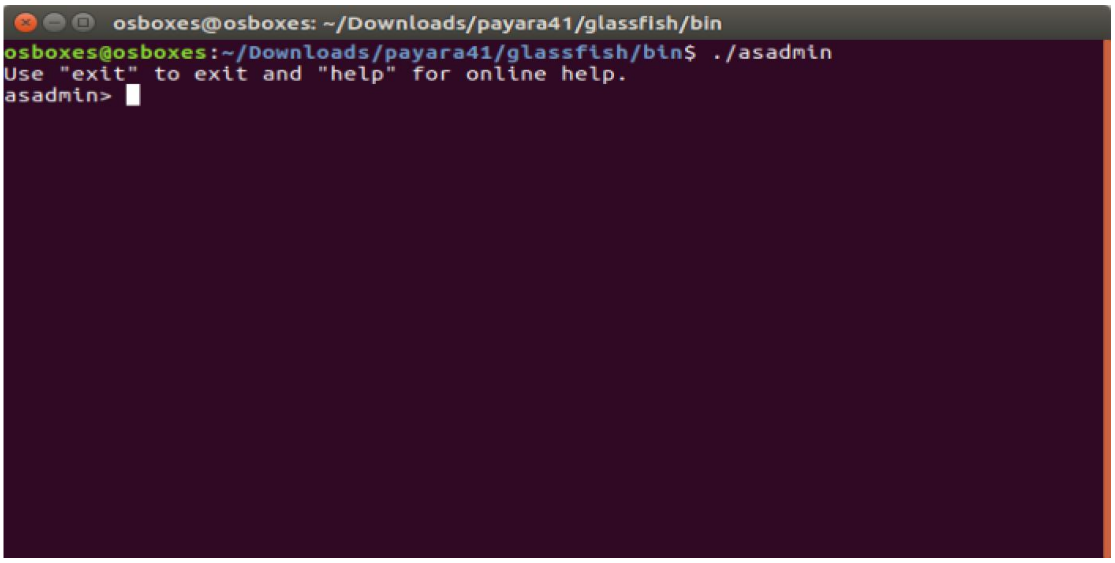
Applications

Applications can be enterprise or web applications, or various kinds of modules. Restart an application or module by clicking on the reload link, this action will apply only to the targets that the application or module is enabled on.

Deployed Applications (1)

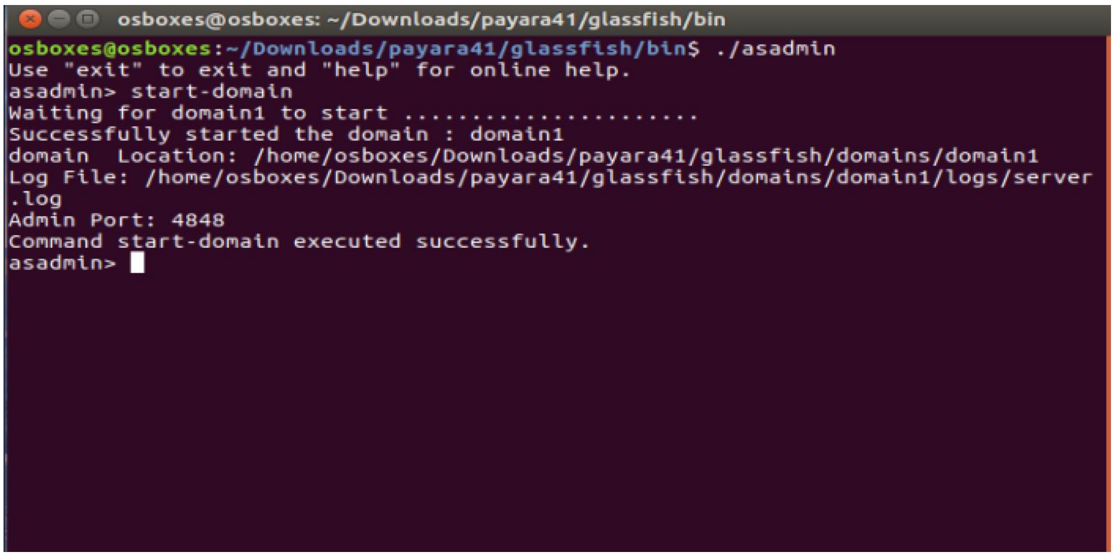
Select	Name	Deployment Order	Enabled	Engines	Action
☐	AmazonShopping	100	✓	ear, web, ejb	| Redeploy | Reload

Step 5: Find the "/glassfish/bin/asadmin" and open it in terminal



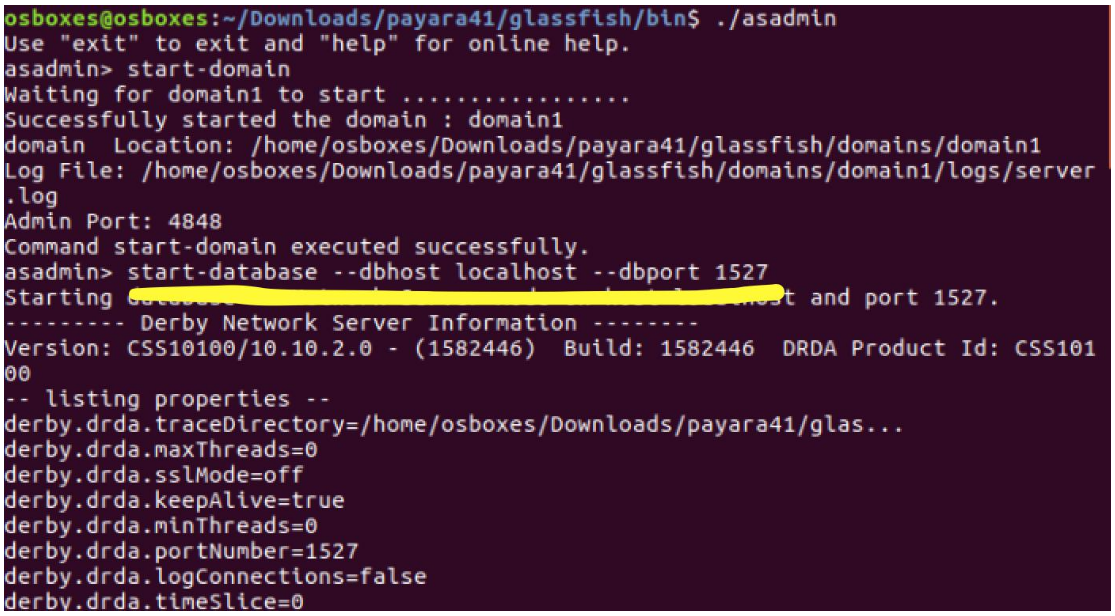
```
osboxes@osboxes: ~/Downloads/payara41/glassfish/bin
osboxes@osboxes:~/Downloads/payara41/glassfish/bin$ ./asadmin
Use "exit" to exit and "help" for online help.
asadmin> 
```

Step 6: start-domain

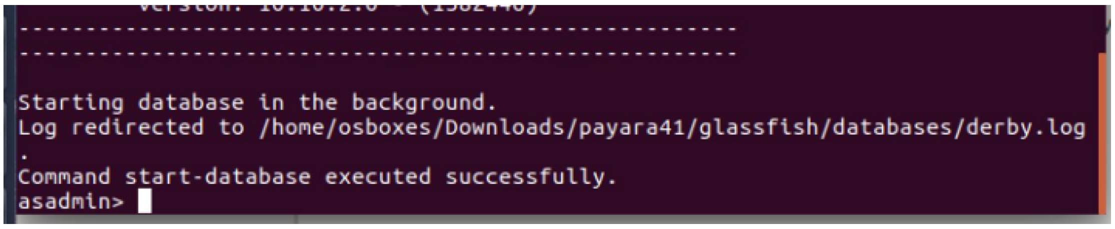


```
osboxes@osboxes: ~/Downloads/payara41/glassfish/bin
osboxes@osboxes:~/Downloads/payara41/glassfish/bin$ ./asadmin
Use "exit" to exit and "help" for online help.
asadmin> start-domain
Waiting for domain1 to start ......................
Successfully started the domain : domain1
domain  Location: /home/osboxes/Downloads/payara41/glassfish/domains/domain1
Log File: /home/osboxes/Downloads/payara41/glassfish/domains/domain1/logs/server
.log
Admin Port: 4848
Command start-domain executed successfully.
asadmin>
```

Step 7: start database

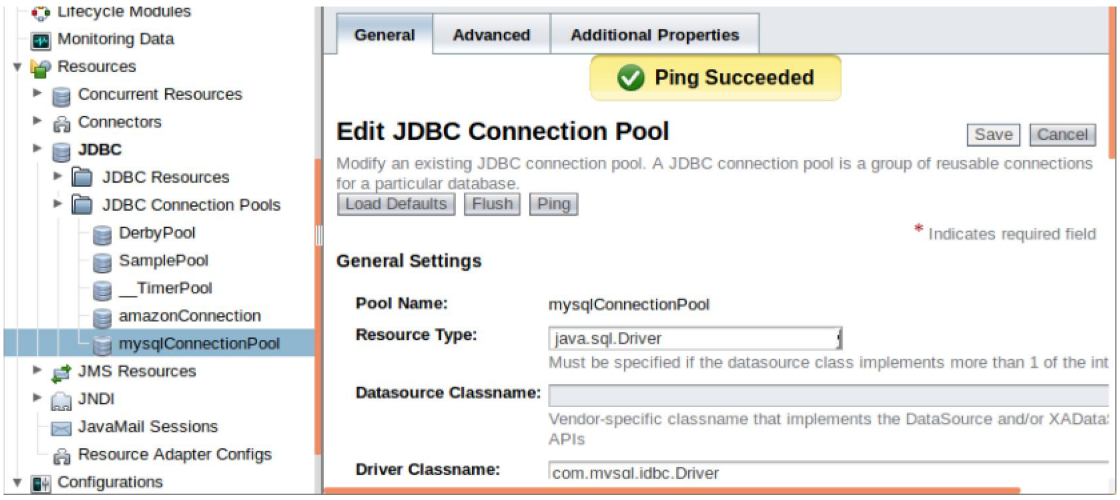


```
osboxes@osboxes:~/Downloads/payara41/glassfish/bin$ ./asadmin
Use "exit" to exit and "help" for online help.
asadmin> start-domain
Waiting for domain1 to start .................
Successfully started the domain : domain1
domain  Location: /home/osboxes/Downloads/payara41/glassfish/domains/domain1
Log File: /home/osboxes/Downloads/payara41/glassfish/domains/domain1/logs/server
.log
Admin Port: 4848
Command start-domain executed successfully.
asadmin> start-database --dbhost localhost --dbport 1527
Starting [illegible]t and port 1527.
--------- Derby Network Server Information --------
Version: CSS10100/10.10.2.0 - (1582446)  Build: 1582446  DRDA Product Id: CSS101
00
-- listing properties --
derby.drda.traceDirectory=/home/osboxes/Downloads/payara41/glas...
derby.drda.maxThreads=0
derby.drda.sslMode=off
derby.drda.keepAlive=true
derby.drda.minThreads=0
derby.drda.portNumber=1527
derby.drda.logConnections=false
derby.drda.timeSlice=0
```



```
         Version: 10.10.2.0 - (1582446)
------------------------------------------------------
------------------------------------------------------

Starting database in the background.
Log redirected to /home/osboxes/Downloads/payara41/glassfish/databases/derby.log
.
Command start-database executed successfully.
asadmin>
```

Step 7: refresh Admin Console and test it



Ping Succeeded

Edit JDBC Connection Pool

Modify an existing JDBC connection pool. A JDBC connection pool is a group of reusable connections for a particular database.

* Indicates required field

General Settings

Pool Name:	mysqlConnectionPool
Resource Type:	java.sql.Driver Must be specified if the datasource class implements more than 1 of the int
Datasource Classname:	 Vendor-specific classname that implements the DataSource and/or XAData APIs
Driver Classname:	com.mysql.jdbc.Driver