

Project

1. Summary:

Front-end:

A E-Commercial Website which can buy goods, search items.

The customer can sign up, login in and add items into the cart, the cart can show the price for the selected items and can be added, deleted in the cart. And customers can search items by name, and select what department of items they want to purchase.

Back-end:

The admin who has an account number can log in and manage the items.

For the General, the admin can add, search and delete generals.

For the Book, the admin can add and delete books.

For the Food, the admin can add and delete foods.

All the items can be added with pictures.

For the Food, because it has the production date and shelf life, so it would be added with dates.

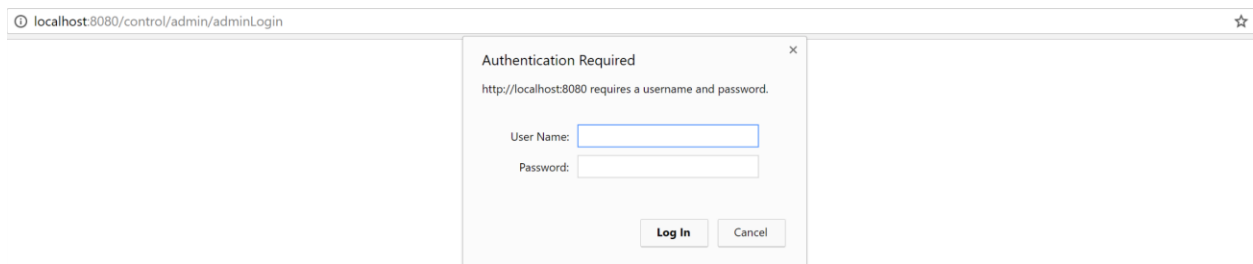
The admin also can make a pdf for all the items.

2. Function

First, the Back-end:

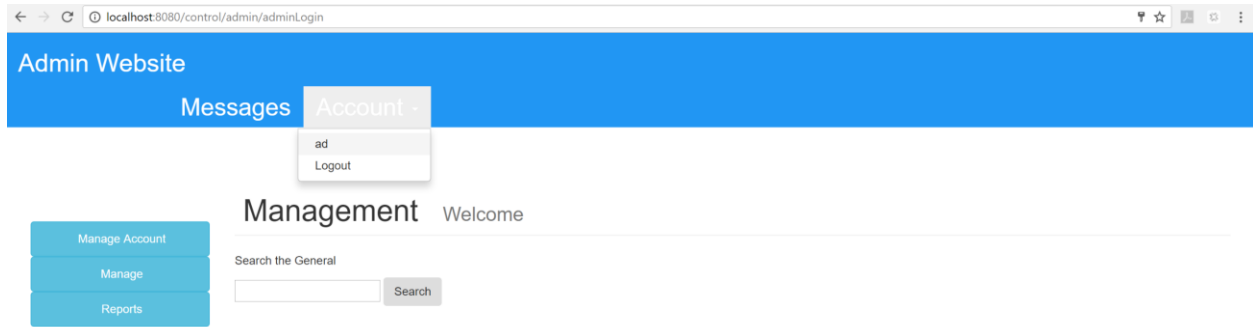
1. Admin log in

The Admin Login part uses the programmatic security.

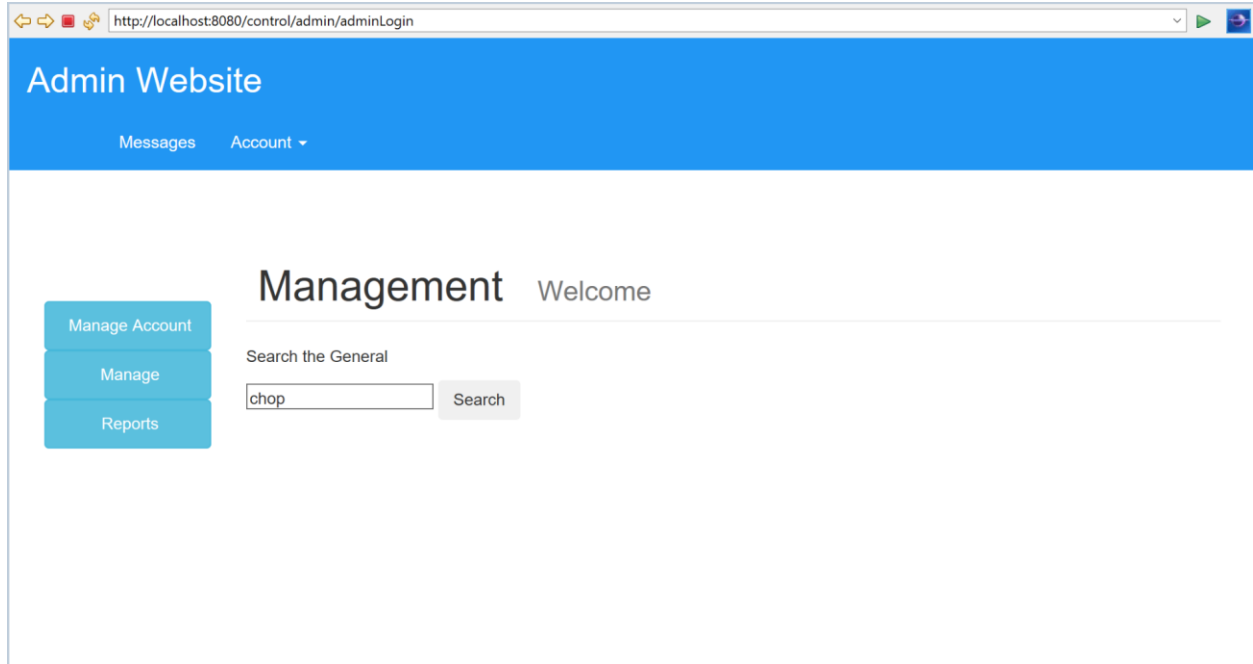


2. When the admin (name: ad, password: ad) log in

The admin can search General items here, and the admin can logout and see its admin name.



3. Search the General Items




The search part supports fuzzy queries using Criteria, like type “chop”,

Admin Website

[Messages](#)[Account ▾](#)[Manage Account](#)[Manage](#)[Reports](#)

Management Welcome

Name	Picture	Price	Description
Chopsticks		3.0	Bamboo

We can see the chopsticks here.

4. Add admin

When click the “Manage Account” button,

Admin Website

[Messages](#)[Account ▾](#)[Manage Account](#)[Manage](#)[Reports](#)

Management Welcome

First Name:	<input type="text"/>
Last Name:	<input type="text"/>
Admin Name:	<input type="text"/>
Password:	<input type="password"/>
<input type="button" value="Add Admin"/>	

If add success,

Admin Website

Messages

Account ▾

Management

Welcome

Manage Account

Manage

Reports

Success

5. Manage Items

Admin Website

Messages

Account ▾

Management

Welcome

Manage Account

Manage

General Item

Book

Food

Manage All

Reports

Search the General

Search

The manage part has 4 sub-parts.

I. Manage General Items

Admin Website

MessagesAccount ▾

Manage Account

Manage

Reports

Management

Welcome

How many Items do you want to add?

Enter:

Submit

When the admin type numbers here, it will show the table to add General items using JavaScript to avoid change the page.

Admin Website

MessagesAccount ▾

Manage Account

Manage

Reports

Management

Welcome

How many Items do you want to add?

Enter:

Name	Picture	Quantity	Price/per	Sale Price	Description
	<div>Browse...</div>				
	<div>Browse...</div>				

Submit

Using JavaScript(onkeypress) to avoid typing non-number string here.

When the admin enters all, it will show the success page.

If the database already has the items the admin input(the same name), when the admin add that item, it will update its information, the quantity will add to the item which saved in the database.

Admin Website

[Messages](#)[Account](#) ▾[Manage Account](#)[Manage](#)[Reports](#)

Management Welcome

How many Items do you want to add?

Enter:

Name	Picture	Quantity	Price/per	Sale Price	Description
Chopsticks	C:\Users\zhaos Browse...	55	1.3	3.9	de By Bamboo

<input type="button" value="Delete"/>	Chopsticks		555	1.3	3.9	Made By Bamboo
---------------------------------------	------------	--	-----	-----	-----	----------------

From 500 changed to 555.

II. Manage Book

The Book Management Part is like the General Item Management part, but need to enter ISBN, author, which are the Books' special attributes.

Management Welcome

How many Books do you want to add?

Enter:

Name	ISBN	Author	Picture	Quantity	Price/per	Sale Price	Description
			<input type="button" value="Choose File"/> No file chosen				
			<input type="button" value="Choose File"/> No file chosen				
			<input type="button" value="Choose File"/> No file chosen				

III. Manage Food

Manage Account
Manage
Reports

Management

Welcome

How many Food do you want to add?

Enter:

Name	Production Date	Self Life	Picture	Quantity	Price/per	Sale Price	Description
			Choose File No file chosen				

PrevNext
April 2017
SuMoTuWeThFrSa
1
2 3 4 5 6 7 8
9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30

Submit

In this part, the admin need to enter Food's production date and self life, and to make sure the self life is later than the production date, I use the jQuery.

Manage Account
Manage
Reports

Management

Welcome

How many Food do you want to add?

Enter:

Name	Production Date	Self Life	Picture	Quantity	Price/per	Sale Price	Description
Milk	04/12/2017		Choose File milk1.jpg	300	1	1.4	Fresh

PrevNext
April 2017
SuMoTuWeThFrSa
1
2 3 4 5 6 7 8
9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30

Submit

The red part is the date before the production date, and the admin cannot choose it.

IV. Manage All

6. Reports

When the admin click the Seller button in the Reports, it will show all the items in database.

Admin Website

Messages Account -

Manage Account

Manage

Reports

Management

Welcome

HTML to PDF

Name	Price	Description	Production Date	Self Life
Snow Crash	20.0	Fiction		
Gone with the wind	35.0	Great		
Milk	2.89	Fresh	2017-04-01 00:00:00.0	2017-04-11 00:00:00.0
Orange	3.0	King	2017-04-12 00:00:00.0	2017-07-20 00:00:00.0
Chopsticks	3.0	Bamboo		
Cup	15.0	Couple		
Rug	22.0	op		

DOWNLOAD PDF

And the admin can click the “DOWNLOAD PDF” hyperlink to download this form.

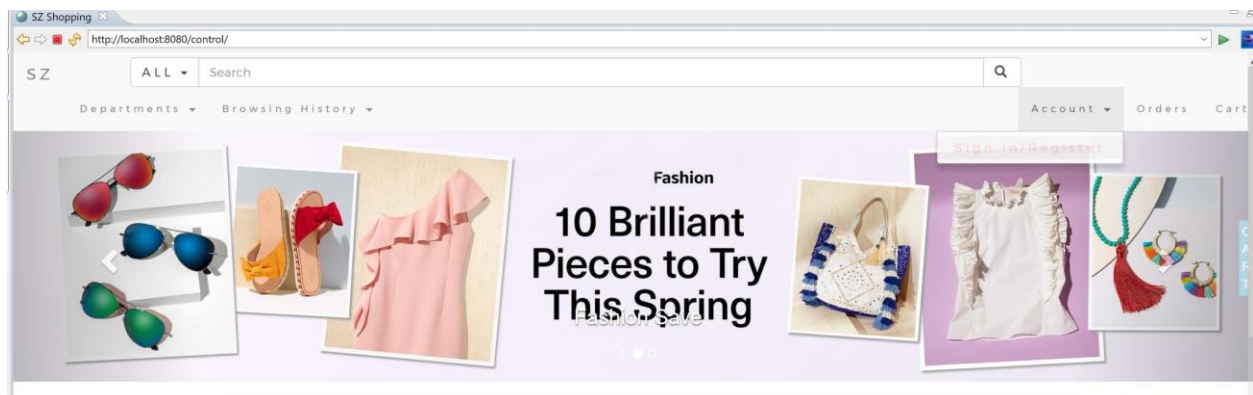
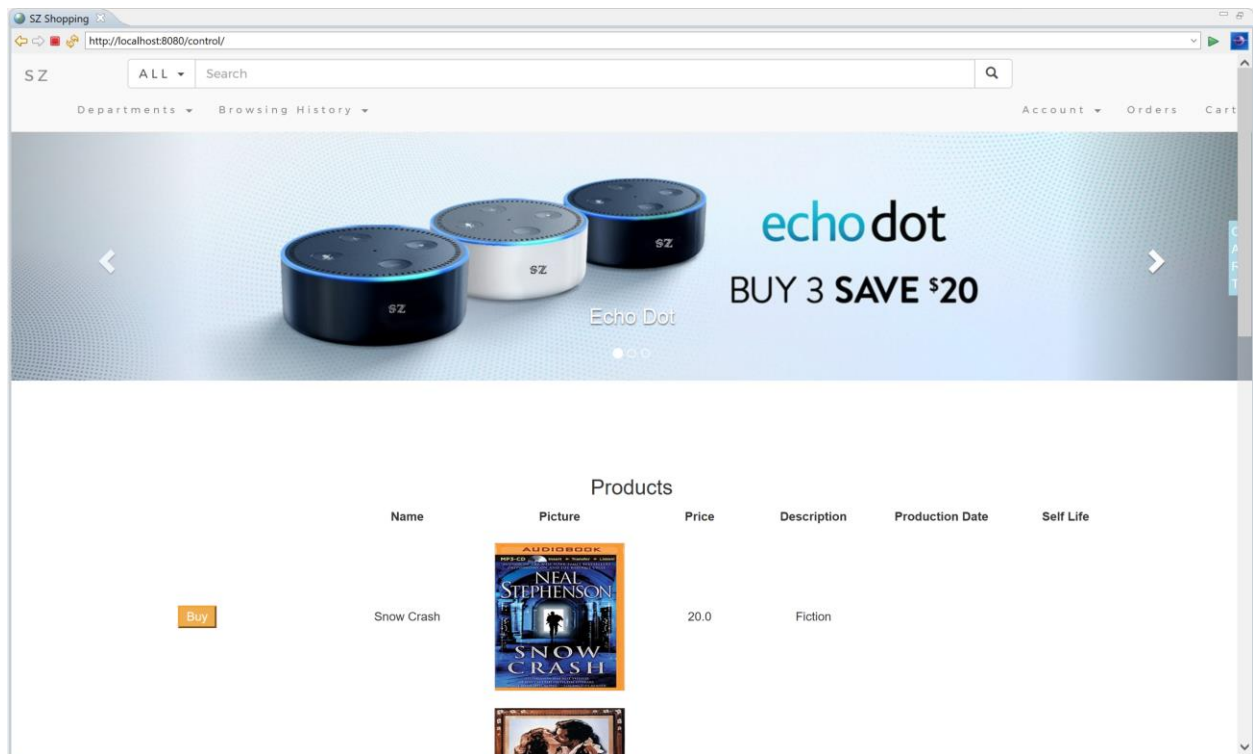
html2pdf.pdf1 / 1

HTML to PDF

Name	Price	Description	Production Date	Self Life
Snow Crash	20.0	Fiction		
Gone with the wind	35.0	Great		
Milk	2.89	Fresh	2017-04-01 00:00:00.0	2017-04-11 00:00:00.0
Orange	3.0	King	2017-04-12 00:00:00.0	2017-07-20 00:00:00.0
Chopsticks	3.0	Bamboo		
Cup	15.0	Couple		
Rug	22.0	op		

Second, the Front-end:

The index page of customer



When customer click the "Sign in/Register" button, it will go to the sign in page.

Account

http://localhost:8080/control/customer/account.htm

New? Register

User Name:

Password:

Login

If the customer never register, he/she can click the “New? Register” hyperlink, and go to the register page.

http://localhost:8080/control/customer/register.htm

First Name:

Last Name:

User Name:

Password:

Email:

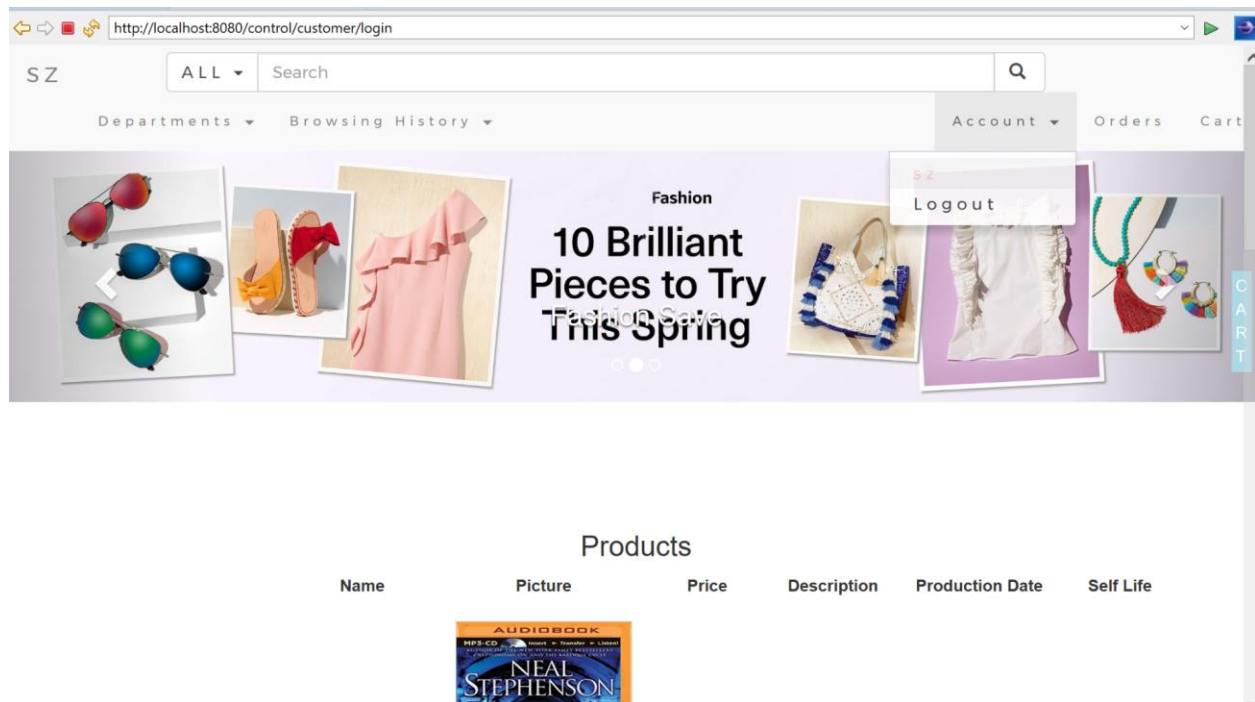
Are you student? ☐ Yes ☒ No

Register User

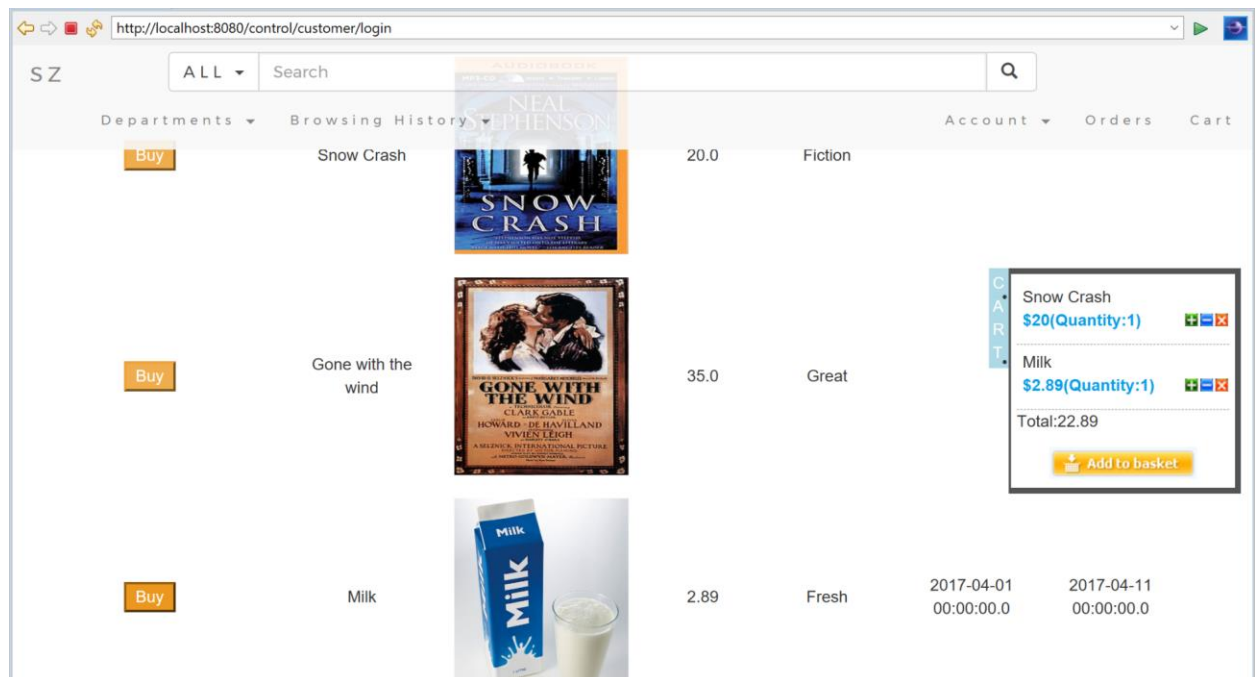
When customer enter the password, the back-end will use BASE64Decoder to encrypt the password and save in the database. And when the customer enter the real Email, he/she will receive an email to confirm.

customerAddr	customerAddress2	customerCard	customerEmail	customerPassword	customerPayExpiration	isStudent	userName	personID
NULL	NULL	0	zhao.shen@husky.neu.edu	xmp	NULL	0	<test>	16

When the customer login,



When the customer click the “Buy” button to make the purchase,



The items’ information and the total price will show in the cart.

When the customer click the “+”, “-”, “x” button, it also reacts. For example, the customer click the “+”, and the Quantity will add, and the Total price will add, too.

<div>Buy</div>	Gone with the wind		35.0	Great		
<div>Buy</div>	Milk		2.89	Fresh	2017-04-01 00:00:00.0	2017-04-11 00:00:00.0

CART

Snow Crash

\$40(Quantity:2)

+

-

x

Milk

\$2.89(Quantity:1)

+

-

x

Total:42.89

+

Add to basket

When click the “Add to basket”, it will show:

<div>Buy</div>	Milk		2.89	Fresh	2017-04-01 00:00:00.0	2017-04-11 00:00:00.0
<div>Buy</div>	Orange		3.0	King	2017-04-12 00:00:00.0	2017-07-20 00:00:00.0



CART

Nothing

Total:42.89

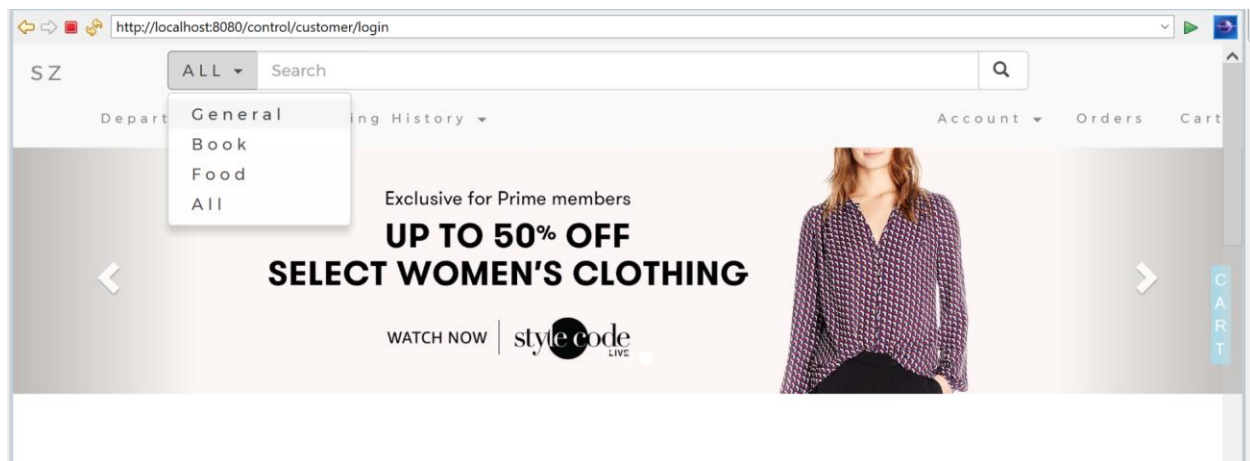
When enter something in the search bar, like “c”,

The screenshot shows a web browser at the URL `http://localhost:8080/control/customer/login`. The search bar contains the letter 'c'. Below the search bar, a table displays the search results. The table has four columns: Name, Picture, Sale, and Description. Two items are listed: 'Chopsticks' and 'Cup'.

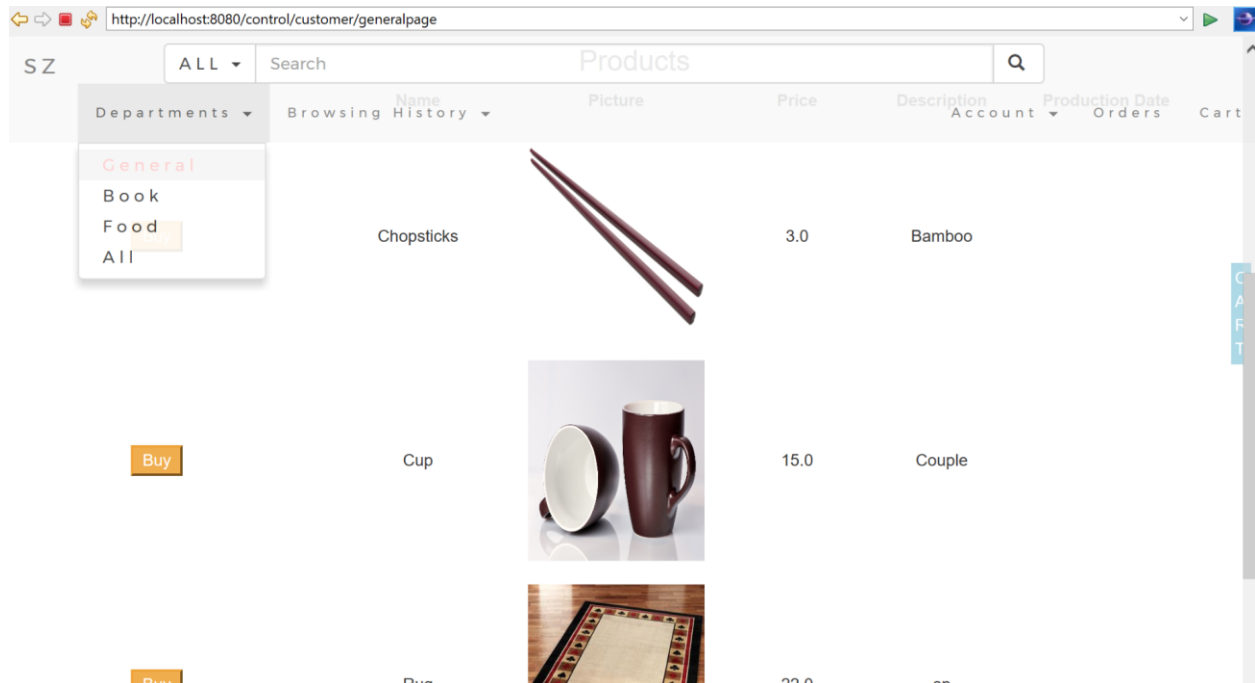
Name	Picture	Sale	Description
Chopsticks		3	Bamboo
Cup		15	Couple

It will show all the items which their name begins with “c”, this is made by Ajax.

The customer can also click the “General” button to only see the items in General Item category.



And the “General” button in “Departments” is the same function.



Third, this website uses the filter to avoid XSS injection and the SSL certificates to protect the site.

3. References of Controller

AdminController

```
package com.shopping.control;
```

```
import java.io.IOException;
```

```
import java.util.List;
```

```
import javax.servlet.http.HttpServletRequest;
```

```
import javax.servlet.http.HttpServletResponse;
```

```
import javax.servlet.http.HttpSession;
```

```
import org.apache.commons.mail.DefaultAuthenticator;
```

```
import org.apache.commons.mail.Email;
import org.apache.commons.mail.SimpleEmail;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.WebDataBinder;
import org.springframework.web.bind.annotation.InitBinder;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;
```

```
import com.shopping.dao.AdminDAO;
import com.shopping.dao.ProductDAO;
import com.shopping.exception.AdminException;
import com.shopping.pojo.Admin;
import com.shopping.pojo.Admin;
import com.shopping.pojo.Product;
import com.shopping.validator.AdminValidator;
```

```
import sun.misc.BASE64Decoder;
```

```
@Controller
```

```
@RequestMapping("/admin/*")
```

```
public class AdminController {
```

```
    @Autowired
```

```
    @Qualifier("adminDao")
```

```
    AdminDAO adminDao;
```



```
@Autowired
@Qualifier("productDao")
ProductDAO productDao;
```

```
@Autowired
@Qualifier("adminValidator")
AdminValidator validator;
```

```
@InitBinder
private void initBinder(WebDataBinder binder) {
    binder.setValidator(validator);
}
```

```
// @RequestMapping(value = "/adminLogin", method = RequestMethod.GET)
// protected ModelAndView adminLogin() throws Exception {
//
//     return new ModelAndView("admin-login");
//
// }
//
// @RequestMapping(value = "/adminLogin", method = RequestMethod.POST)
// protected ModelAndView adminLoginSuccess(HttpServletRequest request) throws
Exception {
//     HttpSession session = (HttpSession) request.getSession();
//     try {
//         System.out.println("loginAdmin");
```

```

//                // check if this admin exist
//                Admin adm =
adminDao.getAdNandP(request.getParameter("adminName"),
request.getParameter("adminPassword"));
//
//                if (adm == null) {
//                System.out.println("UserName/Password does not exist");
//                session.setAttribute("errorMessage", "UserName/Password does
not exist");
//                return new ModelAndView("error");
//                }
//
//                session.setAttribute("adm", adm);
//                return new ModelAndView("admin-home", "admin", adm);
//        } catch (Exception e) {
//                System.out.println("Exception: " + e.getMessage());
//                session.setAttribute("errorMessage", "error while login");
//                return new ModelAndView("error");
//        }
//
//    }

```

```

private void askForPassword(HttpServletResponse response)
{
    response.setStatus(response.SC_UNAUTHORIZED);//I.e.,401
    response.setHeader("WWW-Authenticate", "BASIC realm=\"FinalShop\"");
}

```

```

private boolean areEqual(String s1, String s2)
{

```

```

List<Admin> admins =adminDao.show();
for(int i=0;i<admins.size();i++){
    Admin admin = admins.get(i);
    if(admin.getAdminName().equals(s1) &&
admin.getAdminPassword().equals(s2))
    {
        System.out.println(s2);
        return true;
    }
}
return false;
}

```

```

@RequestMapping(value = "/adminLogin", method = RequestMethod.GET)
public ModelAndView authentication(HttpServletRequest request,HttpServletResponse
response) throws IOException {

```

```

    String authorization = request.getHeader("Authorization");

    if(authorization == null){
        askForPassword(response);
    }else{
        String adminInfo = authorization.substring(6).trim();
        BASE64Decoder decoder = new BASE64Decoder();
        String nameAndPassword = new
String(decoder.decodeBuffer(adminInfo));

        int index = nameAndPassword.indexOf(":");
        String admin = nameAndPassword.substring(0, index);

```

```

        String password = nameAndPassword.substring(index+1);
        Admin adm = adminDao.getAdNandP(admin, password);
        request.getSession().setAttribute("adm", adm);
        if(areEqual(admin,password)){
            return new ModelAndView("admin-home", "admin", adm);
        }else{
            askForPassword(response);
        }
    }

    return null;
}

```

```

// 2.3 Logout the account, and return the main page
@RequestMapping(value = "/logout", method = RequestMethod.GET)
protected ModelAndView logoutAdmin(HttpServletRequest request,
HttpServletResponse response) throws Exception {
    HttpSession session = request.getSession(false);// avoid to create

    // Session
    if (session == null) {
        return new ModelAndView("admin-login");
    }
    session.removeAttribute("admin");
    System.out.print("Logout");
    return new ModelAndView("admin-login", "admin", new Admin());
}

```

```

// admin-home
@RequestMapping(value = "/adminHome", method = RequestMethod.GET)
protected ModelAndView adminHome() throws Exception {

    return new ModelAndView("admin-home");
}

// addTable
@RequestMapping(value = "/addTable", method = RequestMethod.GET)
protected ModelAndView addTable() throws Exception {

    return new ModelAndView("addTable");

}

//Manage All
@RequestMapping(value="/manage", method=RequestMethod.GET)
protected ModelAndView goManageTable(HttpServletRequest request) throws
Exception {

    List<Product>products = productDao.showAll();
    request.getSession().setAttribute("products",products);
    System.out.println("manage");
    return new ModelAndView("admin-manage");

}

@RequestMapping(value="/manage", method=RequestMethod.POST)
protected ModelAndView ManageTable(HttpServletRequest request) throws Exception {

    List<Product>products = productDao.showAll();
    request.getSession().setAttribute("products",products);
    return new ModelAndView("admin-manage");
}

```

```

    }

    //pdf
    @RequestMapping(value="/pdf", method=RequestMethod.GET)
    protected String showPdf() throws Exception {
        return "admin-pdf";
    }

    //add admin
    @RequestMapping(value="/addAdmin", method=RequestMethod.GET)
    protected ModelAndView addAdmin() throws Exception {
        return new ModelAndView("admin-add", "admin", new Admin());
    }

    @RequestMapping(value = "/admin/addAdmin", method = RequestMethod.POST)
    protected ModelAndView addSuccess(HttpServletRequest request,
    @ModelAttribute("admin") Admin admin, BindingResult result) throws Exception {
        BASE64Decoder decoder = new BASE64Decoder();
        validator.validate(admin, result);
        String up=new String(decoder.decodeBuffer(admin.getAdminPassword()));
        admin.setAdminPassword(up);
        System.out.println(admin.getAdminName()+admin.getAdminPassword());
        if (result.hasErrors()) {
            return new ModelAndView("admin-register", "admin", admin);
        }

        try {

            System.out.print("register Success");

```

```

        Admin adm = adminDao.addAdmin(admin);
        request.getSession().setAttribute("admin", adm);
        return new ModelAndView("admin-addsuccess", "admin", adm);

    } catch (AdminException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("error", "errorMessage", "error while login");
    }
}
}

```

CustomerController

```
package com.shopping.control;
```

```

import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.List;
import java.util.Properties;

import javax.mail.internet.InternetAddress;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

```

```
import org.apache.commons.mail.DefaultAuthenticator;
import org.apache.commons.mail.Email;
import org.apache.commons.mail.SimpleEmail;
import org.json.JSONObject;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.mail.SimpleMailMessage;
import org.springframework.mail.javamail.JavaMailSenderImpl;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.WebDataBinder;
import org.springframework.web.bind.annotation.InitBinder;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;
```

```
//import com.shopping.dao.CartDAO;
import com.shopping.dao.CustomerDAO;
import com.shopping.dao.ProductDAO;
import com.shopping.exception.CustomerException;
//import com.shopping.pojo.Cart;
//import com.shopping.exception.CustomerException;
import com.shopping.pojo.Customer;
import com.shopping.pojo.Product;
import com.shopping.validator.CustomerValidator;
```

```
import sun.misc.BASE64Decoder;
```

```
@Controller
```



```

@RequestMapping("/customer/*")
public class CustomerController{

    @Autowired
    @Qualifier("customerDao")
    CustomerDAO customerDao;

    @Autowired
    @Qualifier("customerValidator")
    CustomerValidator validator;

    @Autowired
    @Qualifier("productDao")
    ProductDAO productDao;

    @InitBinder
    private void initBinder(WebDataBinder binder) {
        binder.setValidator(validator);
    }

    // Go to the account page, 1.login / 2.register
    @RequestMapping(value = "/account", method = RequestMethod.GET)
    protected String goToUserHome() throws Exception {
        return ("customer-home");
    }

    // 1. login
    @RequestMapping(value = "/login", method = RequestMethod.POST)

```

protected ModelAndView loginUser(HttpServletRequest request) throws Exception {

 HttpSession session = (HttpSession) request.getSession();

 try {

 System.out.print("loginCustomer");

 BASE64Decoder decoder = new BASE64Decoder();

 String valid=new
String(decoder.decodeBuffer(request.getParameter("customerPassword")));

 Customer cus =
customerDao.getNandP(request.getParameter("userName"),valid);

 if (cus == null) {

 System.out.println("UserName/Password does not exist");

 session.setAttribute("errorMessage", "UserName/Password does
not exist");

 return new ModelAndView("error");

 }

 session.setAttribute("cus", cus);

 return new ModelAndView("/index", "customer", cus);

 } catch (Exception e) {

 System.out.println("Exception: " + e.getMessage());

 session.setAttribute("errorMessage", "error while login");

 return new ModelAndView("error");

 }

}

//2-1. go to register page

```

@RequestMapping(value = "/customer/register", method = RequestMethod.GET)
protected ModelAndView registerCustomer() throws Exception {

    System.out.print("registerUser");

    return new ModelAndView("customer-register", "customer", new Customer());

}

```

//2-2. submit the information, then return the main page

```

@RequestMapping(value = "/customer/register", method = RequestMethod.POST)
protected ModelAndView registerSuccess(HttpServletRequest request,
@ModelAttribute("customer") Customer customer, BindingResult result) throws Exception {

    BASE64Decoder decoder = new BASE64Decoder();
    validator.validate(customer, result);

    String up=new String(decoder.decodeBuffer(customer.getCustomerPassword()));
    customer.setCustomerPassword(up);

    System.out.println(customer.getUserName()+"PPP:"+customer.getCustomerPassword());
    if (result.hasErrors()) {

        return new ModelAndView("customer-register", "customer", customer);
    }

    try {

        System.out.print("register Success");

        Customer cus = customerDao.register(customer);
        String name = cus.getUserName();
        System.out.println("User:"+cus.getUserName());
        String sendEmail = cus.getCustomerEmail();

```

```
request.getSession().setAttribute("customer", cus);
```

```
Email sendemail = new SimpleEmail();
```

sendemail.setHostName("smtp.gmail.com");//If a server is capable of sending email, then you don't need the authentication. In this case, an email server needs to be running on that machine. Since we are running this application on the localhost and we don't have a email server, we are simply asking gmail to relay this email.

```
sendemail.setSmtPort(587);
```

```
sendemail.setAuthenticator(new DefaultAuthenticator("leezz569@gmail.com",  
"*****")); // To prevent leakage of information, with * mark
```

```
sendemail.setSSLonConnect(true);
```

sendemail.setFrom("leezz569@gmail.com");//This email will appear in the from field of the sending email. It doesn't have to be a real email address.This could be used for phishing/spoofing!

```
sendemail.setSubject("Confirm Mail");
```

```
sendemail.setMsg("Sign up confirmation for " + name + ". This is a confirmation email  
for your registration");
```

```
sendemail.addTo(sendEmail);//Will come from the database
```

```
sendemail.send();
```

```
return new ModelAndView("/index", "customer", cus);
```

```
} catch (CustomerException e) {
```

```
System.out.println("Exception: " + e.getMessage());
```

```
return new ModelAndView("error", "errorMessage", "error while login");
```

```
}
```

```
}
```

//2.3 Logout the account, and return the main page

```
@RequestMapping(value = "/logout", method = RequestMethod.GET)
```

```
protected ModelAndView logoutCustomer(HttpServletRequest request,  
HttpServletResponse response) throws Exception {
```

```
HttpSession session = request.getSession(false);//avoid to create Session
```

```

        if(session == null){
            return new ModelAndView("/index");
        }
        session.removeAttribute("customer");
        System.out.print("Logout");

        return new ModelAndView("/index", "customer", new Customer());

    }

    @RequestMapping(value="aa",method = RequestMethod.GET)
    protected String ajaxsearch() throws Exception {
        return "/search";
    }

    //generalpage
    @RequestMapping(value="generalpage",method = RequestMethod.GET)
    protected String generalpage() throws Exception {
        return "/customer-general";
    }

    @RequestMapping(value="all",method = RequestMethod.GET)
    protected String Allpage() throws Exception {
        return "/index";
    }

}

```

BookController

```
package com.shopping.control;
```

```
import java.io.File;
```

```
import java.util.ArrayList;
```

```
import java.util.List;
```

```
import javax.servlet.ServletContext;
```

```
import javax.servlet.http.HttpServletRequest;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.beans.factory.annotation.Qualifier;
```

```
import org.springframework.stereotype.Controller;
```

```
import org.springframework.web.bind.WebDataBinder;
```

```
import org.springframework.web.bind.annotation.InitBinder;
```

```
import org.springframework.web.bind.annotation.RequestMapping;
```

```
import org.springframework.web.bind.annotation.RequestMethod;
```

```
import org.springframework.web.bind.annotation.RequestParam;
```

```
import org.springframework.web.multipart.commons.CommonsMultipartFile;
```

```
import org.springframework.web.servlet.ModelAndView;
```

```
import com.shopping.dao.BookDAO;
```

```
import com.shopping.dao.ProductDAO;
```

```
import com.shopping.dao.BookDAO;
```

```
import com.shopping.pojo.Book;
```

```
import com.shopping.pojo.Product;
```

```
import com.shopping.pojo.Product.Type;
```

```
import com.shopping.validator.BookValidator;
```

```
import com.shopping.validator.BookValidator;
```

```
@Controller
```

```
@RequestMapping("/book/*")
```

```
public class BookController {  
    private String[] name;  
    private String[] filename;  
    private String[] quantity;  
    private String[] price;  
    private String[] saleprice;  
    private String[] description;  
    private String[] isbn;  
    private String[] author;  
    //private String[] type;  
  
    @Autowired  
    @Qualifier("productDao")  
    ProductDAO productDao;  
  
    @Autowired  
    @Qualifier("bookDao")  
    BookDAO bookDao;  
  
    @Autowired  
    @Qualifier("bookValidator")  
    BookValidator bookValidator;  
  
    @Autowired  
    ServletContext servletContext;  
  
    @InitBinder  
    private void initBinder(WebDataBinder binder) {  
        binder.setValidator(bookValidator);  
    }  
}
```

```

private static List<Book> books = new ArrayList<Book>();

// go to addTable
@RequestMapping(value = "/addBook", method = RequestMethod.GET)
protected ModelAndView addGeneralTable() throws Exception {

    return new ModelAndView("admin-addBook");
}

// go to success page
@RequestMapping(value = "/addsuccess", method = RequestMethod.POST)
protected ModelAndView addSuccess(HttpServletRequest
request, @RequestParam("img") List<CommonsMultipartFile> img) throws Exception {

    System.out.println(request.getParameter("rows"));
    int num = Integer.parseInt(request.getParameter("rows"));
    Book[] books = new Book[num];
    System.out.println("length.." + books.length);
    for (int i = 0; i < num; i++)
    {
        books[i] = new Book();
        name = request.getParameterValues("name");
        quantity = request.getParameterValues("quantity");
        saleprice = request.getParameterValues("saleprice");
        description = request.getParameterValues("description");
        price = request.getParameterValues("price");
        isbn = request.getParameterValues("isbn");
        author = request.getParameterValues("author");

        // type

```



```

books[i].setName(name[i]);
books[i].setImg(img.get(i));
books[i].setQuantity(Integer.parseInt(quantity[i]));
books[i].setPrice(Float.valueOf(price[i]));
books[i].setSaleprice(Float.valueOf(saleprice[i]));
books[i].setDescription(description[i]);
books[i].setIsbn(isbn[i]);
books[i].setAuthor(author[i]);
books[i].setType(Type.Book);
File directory;
String check = File.separator; // Checking if system is linux"/"

```

// based on
 windows\" based by

// checking
 separator used.

```

String path = null;
if (check.equalsIgnoreCase("\\")) {
    path = servletContext.getRealPath("").replace("build\\", ""); //
gives real path as Lab9/build/web/
    // it's calculate start with where current servlet in tomcat
    // // so we need to replace build in the path
    System.out.println(path);
}

if (check.equalsIgnoreCase("/")) {
    path = servletContext.getRealPath("").replace("build/", "");
    System.out.println(path);
    System.out.println("111:" + servletContext.getRealPath(""));
    path += "/"; // Adding trailing slash for Mac systems.
}

```

```

        directory = new File(path + "\\Book");
        boolean temp = directory.exists();
        if (!temp) {
            temp = directory.mkdir();
        }
        if (temp) {
            // We need to transfer to a file
            CommonsMultipartFile photoInMemory = books[i].getImg();

            String fileName = photoInMemory.getOriginalFilename(); //get
the file's name

            // could generate file names as well
            File localFile = new File(directory.getPath(), fileName);

            // move the file from memory to the file
            photoInMemory.transferTo(localFile);
            System.out.println("File is stored at" + localFile.getPath());
            books[i].setFilename(localFile.getPath());
            bookDao.addBook(books[i]);
        } else {
            System.out.println("Failed to create directory!");
        }
    }
    List<Product>products = productDao.showAll();
    request.getSession().setAttribute("products",products);
    return new ModelAndView("admin-addsuccess");
}
}

```

GeneralController

```
package com.shopping.control;
```

```
import java.io.File;
```

```
import java.util.ArrayList;
```

```
import java.util.List;
```

```
import javax.servlet.ServletContext;
```

```
import javax.servlet.http.HttpServletRequest;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.beans.factory.annotation.Qualifier;
```

```
import org.springframework.stereotype.Controller;
```

```
import org.springframework.web.bind.WebDataBinder;
```

```
import org.springframework.web.bind.annotation.InitBinder;
```

```
import org.springframework.web.bind.annotation.ModelAttribute;
```

```
import org.springframework.web.bind.annotation.RequestMapping;
```

```
import org.springframework.web.bind.annotation.RequestMethod;
```

```
import org.springframework.web.bind.annotation.RequestParam;
```

```
import org.springframework.web.multipart.commons.CommonsMultipartFile;
```

```
import org.springframework.web.servlet.ModelAndView;
```

```
import com.shopping.dao.GeneralDAO;
```

```
import com.shopping.dao.ProductDAO;
```

```
import com.shopping.pojo.General;
```

```
import com.shopping.pojo.Product;
```

```
import com.shopping.pojo.Product.Type;
```

```
import com.shopping.validator.GeneralValidator;
```

```
@Controller
```

```
@RequestMapping("/general/*")
```

```
public class GeneralController {  
    private String[] name;  
    private String[] quantity;  
    private String[] price;  
    private String[] saleprice;  
    private String[] description;  
    private String[] type;  
    private String[] filename;  
  
    @Autowired  
    @Qualifier("productDao")  
    ProductDAO productDao;  
  
    @Autowired  
    @Qualifier("generalDao")  
    GeneralDAO generalDao;  
  
    @Autowired  
    @Qualifier("generalValidator")  
    GeneralValidator generalValidator;  
  
    @Autowired  
    ServletContext servletContext;  
  
    @InitBinder  
    private void initBinder(WebDataBinder binder) {  
        binder.setValidator(generalValidator);  
    }  
}
```

```

// go to addTable
@RequestMapping(value = "/addGeneral", method = RequestMethod.GET)
protected ModelAndView addGeneralTable() throws Exception {

    return new ModelAndView("admin-addGeneral");
}

// go to success page
@RequestMapping(value = "/addsuccess", method = RequestMethod.POST)
protected ModelAndView addSuccess(HttpServletRequest
request, @RequestParam("img") List<CommonsMultipartFile> img ) throws Exception {

    int num = Integer.parseInt(request.getParameter("rows"));
    General[] generals = new General[num];
    System.out.println("length.." + generals.length);
    for (int i = 0; i < num; i++)
    {

        generals[i] = new General();
        name = request.getParameterValues("name");
        quantity = request.getParameterValues("quantity");
        saleprice = request.getParameterValues("saleprice");
        description = request.getParameterValues("description");
        price = request.getParameterValues("price");
        generals[i].setName(name[i]);
        generals[i].setImg(img.get(i));
        generals[i].setQuantity(Integer.parseInt(quantity[i]));
        generals[i].setPrice(Float.valueOf(price[i]));
        generals[i].setSaleprice(Float.valueOf(saleprice[i]));
        generals[i].setDescription(description[i]);
        generals[i].setType(Type.General);
    }
}

```

```

File directory;
String check = File.separator; // Checking if system is linux"/"
//
based or windows"\\" based by
//
checking separator used.

String path = null;
if (check.equalsIgnoreCase("\\")) {
    path = servletContext.getRealPath("").replace("build\\", ""); //
gives real path as Lab9/build/web/
    // it's calculate start with where current servlet in tomcat
    // // so we need to replace build in the path
    System.out.println(path);
    System.out.println("111:" + servletContext.getRealPath(""));
}

if (check.equalsIgnoreCase("/")) {
    path = servletContext.getRealPath("").replace("build/", "");
    System.out.println(path);
    System.out.println("111:" + servletContext.getRealPath(""));
    path += "/"; // Adding trailing slash for Mac systems.
}

directory = new File(path + "\\General");
boolean temp = directory.exists();
if (!temp) {
    temp = directory.mkdir();
}
if (temp) {
    // We need to transfer to a file

```

```

generals[i].getImg();

CommonsMultipartFile photoInMemory =

String fileName = photoInMemory.getOriginalFilename();

//get the file's name

// could generate file names as well
File localFile = new File(directory.getPath(), fileName);
// move the file from memory to the file
photoInMemory.transferTo(localFile);
System.out.println("File is stored at " +

localFile.getPath());

generals[i].setFilename(localFile.getPath());

} else {
    System.out.println("Failed to create directory!");
}

General verifyGen = generalDao.verifyName(name[i]);
System.out.println("VerifyGEN: "+verifyGen);
if(verifyGen.getName().equals(name[i])){

verifyGen.setQuantity(Integer.parseInt(quantity[i])+verifyGen.getQuantity());
    verifyGen.setPrice(Float.valueOf(price[i]));
    verifyGen.setSaleprice(Float.valueOf(saleprice[i]));
    verifyGen.setDescription(description[i]);
    verifyGen.setImg(img.get(i));
    generalDao.update(verifyGen);
}
else{
    System.out.print("registerNewUser\n");
}

```

```

        generalDao.addGeneral(generals[i]);
    }

}

List<Product>products = productDao.showAll();
request.getSession().setAttribute("products",products);
return new ModelAndView("admin-addsuccess");
}

//Search

@RequestMapping(value="/search", method=RequestMethod.POST)
protected ModelAndView Search(HttpServletRequest request) throws
Exception {

    List<General> searchgeneral =
generalDao.getByName(request.getParameter("gname"));

    return new ModelAndView("general-
search","searchgeneral",searchgeneral);

}

}

```

ProductController

```

package com.shopping.control;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;

```



```

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;
import com.shopping.dao.ProductDAO;
import com.shopping.pojo.Product;

@Controller
@RequestMapping("/product/*")
public class ProductController {
    @Autowired
    @Qualifier("productDao")
    ProductDAO productDao;

    @RequestMapping(value="/deletesuccess", method=RequestMethod.POST)
    protected ModelAndView manage(HttpServletRequest request) throws Exception {
        System.out.println("pppppppppppppppp");
        int proId=Integer.parseInt(request.getParameter("proID"));
        productDao.delete(proId);
        List<Product>products = productDao.showAll();
        request.getSession().setAttribute("products",products);
        return new ModelAndView("admin-home");
    }
}

```

PageController

```

package com.shopping.control;

import java.io.File;

```

```
import java.util.ArrayList;
```

```
import java.util.List;
```

```
import javax.servlet.ServletContext;
```

```
import javax.servlet.http.HttpServletRequest;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.beans.factory.annotation.Qualifier;
```

```
import org.springframework.stereotype.Controller;
```

```
import org.springframework.web.bind.WebDataBinder;
```

```
import org.springframework.web.bind.annotation.InitBinder;
```

```
import org.springframework.web.bind.annotation.ModelAttribute;
```

```
import org.springframework.web.bind.annotation.RequestMapping;
```

```
import org.springframework.web.bind.annotation.RequestMethod;
```

```
import org.springframework.web.bind.annotation.RequestParam;
```

```
import org.springframework.web.multipart.commons.CommonsMultipartFile;
```

```
import org.springframework.web.servlet.ModelAndView;
```

```
import com.shopping.dao.GeneralDAO;
```

```
import com.shopping.dao.ProductDAO;
```

```
import com.shopping.pojo.General;
```

```
import com.shopping.pojo.Product;
```

```
import com.shopping.pojo.Product.Type;
```

```
import com.shopping.validator.GeneralValidator;
```

```
@Controller
```

```
@RequestMapping("/general/*")
```

```
public class GeneralController {
```

```
    private String[] name;
```

```
    private String[] quantity;
```

```
private String[] price;  
private String[] saleprice;  
private String[] description;  
private String[] type;  
private String[] filename;
```

```
@Autowired  
@Qualifier("productDao")  
ProductDAO productDao;
```

```
@Autowired  
@Qualifier("generalDao")  
GeneralDAO generalDao;
```

```
@Autowired  
@Qualifier("generalValidator")  
GeneralValidator generalValidator;
```

```
@Autowired  
ServletContext servletContext;
```

```
@InitBinder  
private void initBinder(WebDataBinder binder) {  
    binder.setValidator(generalValidator);  
}
```

```
// go to addTable  
@RequestMapping(value = "/addGeneral", method = RequestMethod.GET)  
protected ModelAndView addGeneralTable() throws Exception {
```

```

        return new ModelAndView("admin-addGeneral");
    }

    // go to success page
    @RequestMapping(value = "/addsuccess", method = RequestMethod.POST)
    protected ModelAndView addSuccess(HttpServletRequest
request, @RequestParam("img") List<CommonsMultipartFile> img ) throws Exception {
        int num = Integer.parseInt(request.getParameter("rows"));
        General[] generals = new General[num];
        System.out.println("length.." + generals.length);
        for (int i = 0; i < num; i++)
        {
            generals[i] = new General();
            name = request.getParameterValues("name");
            quantity = request.getParameterValues("quantity");
            saleprice = request.getParameterValues("saleprice");
            description = request.getParameterValues("description");
            price = request.getParameterValues("price");

            generals[i].setName(name[i]);
            generals[i].setImg(img.get(i));
            generals[i].setQuantity(Integer.parseInt(quantity[i]));
            generals[i].setPrice(Float.valueOf(price[i]));
            generals[i].setSaleprice(Float.valueOf(saleprice[i]));
            generals[i].setDescription(description[i]);
            generals[i].setType(Type.General);

            File directory;
            String check = File.separator; // Checking if system is linux"/"

```

```

//
based or windows"\" based by

//
checking separator used.

String path = null;
if (check.equalsIgnoreCase("\\")) {
    path = servletContext.getRealPath("").replace("build\\", ""); //
gives real path as Lab9/build/web/
    // it's calculate start with where current servlet in tomcat
    // // so we need to replace build in the path
    System.out.println(path);
    System.out.println("111:" + servletContext.getRealPath(""));
}

if (check.equalsIgnoreCase("/")) {
    path = servletContext.getRealPath("").replace("build/", "");
    System.out.println(path);
    System.out.println("111:" + servletContext.getRealPath(""));
    path += "/"; // Adding trailing slash for Mac systems.
}

directory = new File(path + "\\General");
boolean temp = directory.exists();
if (!temp) {
    temp = directory.mkdir();
}
if (temp) {
    // We need to transfer to a file
    CommonsMultipartFile photoInMemory =
generals[i].getImg();

```

```

        String fileName = photoInMemory.getOriginalFilename();
        //get the file's name

        // could generate file names as well
        File localFile = new File(directory.getPath(), fileName);
        // move the file from memory to the file
        photoInMemory.transferTo(localFile);
        System.out.println("File is stored at " +
localFile.getPath());

        generals[i].setFilename(localFile.getPath());
        System.out.print("registerNewUser");
        generalDao.addGeneral(generals[i]);
    } else {
        System.out.println("Failed to create directory!");
    }
}

List<Product>products = productDao.showAll();
request.getSession().setAttribute("products",products);
return new ModelAndView("admin-addsuccess");
}

//Search

@RequestMapping(value="/search", method=RequestMethod.POST)
protected ModelAndView Search(HttpServletRequest request) throws
Exception {

    List<General> searchgeneral =
generalDao.getByName(request.getParameter("gname"));

    return new ModelAndView("general-
search","searchgeneral",searchgeneral);

}

```

}