**Domain:** E-commerce website for Computers, Laptops and Spare parts

**Summary:**

A Ecommerce website using spring MVC configuration where in Supplier can post his products such as Laptops, Computers and spare parts of computers/External peripheral devices like Usb Sticks, Hard disks, RAM etc. Currently there are only a few pre-defined categories namely Laptops, RAM, USB Sticks and Portable Hard Drives. These can be posted and bought in bulk by customers. The admin entity approves the sale. This project has a vast future scope where in there could be multiple categories as well as products and could be made similar to trending E-commerce websites.

**Technologies Used:**

1. Spring MVC Framework.

2. Hibernate.

3. MySql Server for databse.

4. Java Script for validations.

5. Bootstrap for Front end layout

**Roles and Tasks**

**1. Admin:**

* Adds a supplier
* Deletes a supplier if needed
* Approves orders made by customers
* Declines orders made by customer

**2. Supplier:**

* Add and set category of a product
* Upload Images for the product
* Delete product

**3. Customer:**

* Searches for products based on category
* Places Order for products

**Key Functionalities/Implementations**

* JavaScript validations for forms
* Backend validations for existing users
* Mappings using annotations
* Bootstrap for responsive UI.
* Main admin is added through database queries, he is responsible for adding suppliers via UI.
* The supplier adds products, right now there are limited categories provided in future we can let the admin or supplier create categories.
* The customer can register himself and check the products based on categories and order the based on quantities its total price is calculated.

**Screen Shots:**

**1. Home Page**

**A screenshot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generated**

**2. Login Page**

**A screenshot of a computer

Description automatically generated**

**3. Admin**

**A screenshot of a cell phone

Description automatically generated**

**A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated**

**4. SupplierA screenshot of a cell phone

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**

**5. Customer**

**A screenshot of a cell phone

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**

**SQL DUMP**

-- MySQL dump 10.13 Distrib 8.0.12, for macos10.13 (x86\_64)

--

-- Host: localhost Database: webtools\_final\_project

-- ------------------------------------------------------

-- Server version 8.0.12

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

SET NAMES utf8 ;

/\*!40103 SET @OLD\_TIME\_ZONE=@@TIME\_ZONE \*/;

/\*!40103 SET TIME\_ZONE='+00:00' \*/;

/\*!40014 SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0 \*/;

/\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/;

/\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/;

/\*!40111 SET @OLD\_SQL\_NOTES=@@SQL\_NOTES, SQL\_NOTES=0 \*/;

--

-- Table structure for table `WT\_address`

--

DROP TABLE IF EXISTS `WT\_address`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_address` (

`addressId` int(11) NOT NULL AUTO\_INCREMENT,

`addressLine1` varchar(255) DEFAULT NULL,

`addressLine2` varchar(255) DEFAULT NULL,

`city` varchar(255) DEFAULT NULL,

`country` varchar(255) DEFAULT NULL,

`state` varchar(255) DEFAULT NULL,

`zipCode` varchar(255) DEFAULT NULL,

PRIMARY KEY (`addressId`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_address`

--

LOCK TABLES `WT\_address` WRITE;

/\*!40000 ALTER TABLE `WT\_address` DISABLE KEYS \*/;

INSERT INTO `WT\_address` VALUES (1,'26A Saint Alphonsus Street','Boston Mas','Bos','USA','Mas','2120'),(2,'14 Fulton Drive','Boston, Mas','Boston','Usa','Massachusetts','390239'),(3,'15 Mass Ave','Boston Mas','Boston','USA','Mas','029384'),(4,'21 North Carolina','North Carolina','Carolina','Usa','North Carolina','012930'),(5,'12 granide street','Mas Bos','Boston','USA','Massachusetts','021202');

/\*!40000 ALTER TABLE `WT\_address` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_admin`

--

DROP TABLE IF EXISTS `WT\_admin`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_admin` (

`adminId` int(11) NOT NULL,

PRIMARY KEY (`adminId`),

CONSTRAINT `FKfspnguhuxfjv222d5ut9qoyjy` FOREIGN KEY (`adminId`) REFERENCES `wt\_person` (`personid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_admin`

--

LOCK TABLES `WT\_admin` WRITE;

/\*!40000 ALTER TABLE `WT\_admin` DISABLE KEYS \*/;

INSERT INTO `WT\_admin` VALUES (1);

/\*!40000 ALTER TABLE `WT\_admin` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_customer`

--

DROP TABLE IF EXISTS `WT\_customer`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_customer` (

`customerId` int(11) NOT NULL,

PRIMARY KEY (`customerId`),

CONSTRAINT `FKklb2b2bmw9859f94ucb1hue59` FOREIGN KEY (`customerId`) REFERENCES `wt\_person` (`personid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_customer`

--

LOCK TABLES `WT\_customer` WRITE;

/\*!40000 ALTER TABLE `WT\_customer` DISABLE KEYS \*/;

/\*!40000 ALTER TABLE `WT\_customer` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_order`

--

DROP TABLE IF EXISTS `WT\_order`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_order` (

`orderId` int(11) NOT NULL AUTO\_INCREMENT,

`status` varchar(255) DEFAULT NULL,

`adminId` int(11) DEFAULT NULL,

`customerId` int(11) DEFAULT NULL,

PRIMARY KEY (`orderId`),

KEY `FKrd2xrvxojhn5wqxqbm8831bov` (`adminId`),

KEY `FKp9sa0lx2bq3fmcfiba43q0lbx` (`customerId`),

CONSTRAINT `FKp9sa0lx2bq3fmcfiba43q0lbx` FOREIGN KEY (`customerId`) REFERENCES `wt\_customer` (`customerid`),

CONSTRAINT `FKrd2xrvxojhn5wqxqbm8831bov` FOREIGN KEY (`adminId`) REFERENCES `wt\_admin` (`adminid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_order`

--

LOCK TABLES `WT\_order` WRITE;

/\*!40000 ALTER TABLE `WT\_order` DISABLE KEYS \*/;

/\*!40000 ALTER TABLE `WT\_order` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_orderItem`

--

DROP TABLE IF EXISTS `WT\_orderItem`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_orderItem` (

`orderItemId` int(11) NOT NULL AUTO\_INCREMENT,

`quantity` int(11) DEFAULT NULL,

`orderId` int(11) DEFAULT NULL,

`productId` int(11) DEFAULT NULL,

PRIMARY KEY (`orderItemId`),

KEY `FK7rm288r9mxc4nlfis71iyhpgw` (`orderId`),

KEY `FK47hir86w71gm5pd7jbu4tl3rd` (`productId`),

CONSTRAINT `FK47hir86w71gm5pd7jbu4tl3rd` FOREIGN KEY (`productId`) REFERENCES `wt\_product` (`productid`),

CONSTRAINT `FK7rm288r9mxc4nlfis71iyhpgw` FOREIGN KEY (`orderId`) REFERENCES `wt\_order` (`orderid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_orderItem`

--

LOCK TABLES `WT\_orderItem` WRITE;

/\*!40000 ALTER TABLE `WT\_orderItem` DISABLE KEYS \*/;

/\*!40000 ALTER TABLE `WT\_orderItem` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_person`

--

DROP TABLE IF EXISTS `WT\_person`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_person` (

`personId` int(11) NOT NULL AUTO\_INCREMENT,

`age` int(11) DEFAULT NULL,

`firstName` varchar(255) DEFAULT NULL,

`gender` varchar(255) DEFAULT NULL,

`lastName` varchar(255) DEFAULT NULL,

`middleName` varchar(255) DEFAULT NULL,

`phoneNumber` varchar(255) DEFAULT NULL,

`addressId` int(11) DEFAULT NULL,

PRIMARY KEY (`personId`),

KEY `FKq7loxhfhlqhruhv88qrk78htl` (`addressId`),

CONSTRAINT `FKq7loxhfhlqhruhv88qrk78htl` FOREIGN KEY (`addressId`) REFERENCES `wt\_address` (`addressid`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_person`

--

LOCK TABLES `WT\_person` WRITE;

/\*!40000 ALTER TABLE `WT\_person` DISABLE KEYS \*/;

INSERT INTO `WT\_person` VALUES (1,30,'admin','Male','admin','admin','4044356713',1),(2,0,NULL,NULL,NULL,NULL,'3827515161',2),(3,0,NULL,NULL,NULL,NULL,'6362617283',3),(4,0,NULL,NULL,NULL,NULL,'2838290193',4),(5,0,NULL,NULL,NULL,NULL,'123345465',5);

/\*!40000 ALTER TABLE `WT\_person` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_product`

--

DROP TABLE IF EXISTS `WT\_product`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_product` (

`productId` int(11) NOT NULL AUTO\_INCREMENT,

`availability` int(11) DEFAULT NULL,

`description` varchar(255) DEFAULT NULL,

`photoName` varchar(255) DEFAULT NULL,

`price` float DEFAULT NULL,

`prod\_category` varchar(255) DEFAULT NULL,

`productName` varchar(255) DEFAULT NULL,

`supplierId` int(11) DEFAULT NULL,

PRIMARY KEY (`productId`),

KEY `FKkm7tyamneocdxextfkvgurvnx` (`supplierId`),

CONSTRAINT `FKkm7tyamneocdxextfkvgurvnx` FOREIGN KEY (`supplierId`) REFERENCES `wt\_supplier` (`supplierid`)

) ENGINE=InnoDB AUTO\_INCREMENT=16 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_product`

--

LOCK TABLES `WT\_product` WRITE;

/\*!40000 ALTER TABLE `WT\_product` DISABLE KEYS \*/;

INSERT INTO `WT\_product` VALUES (1,50,'DDR1 - Ram','/cassian/resources/images/DDR1-Ram1556046927725.jpg',200,'Ram','DDR1 - Ram',2),(2,100,'DDR2 - Ram','/cassian/resources/images/DDR2-Ram1556046981258.jpg',230,'Ram','DDR2 - Ram',2),(3,80,'DDR3 - Ram','/cassian/resources/images/DDR3-Ram1556047011259.jpg',240,'Ram','DDR3 - Ram',2),(4,30,'DDR4 - Ram','/cassian/resources/images/DDR4-Ram1556047045164.jpg',290,'Ram','DDR4 - Ram',2),(5,80,'FD-1TB-HDD','/cassian/resources/images/FD-1Tb-Hdd1556047303963.jpg',300,'Portable Hard-Disk','FD-1TB-HDD',3),(6,70,'Segate-1TB-Hdd','/cassian/resources/images/Segate-1Tb-Hdd1556047396460.jpg',250,'Portable Hard-Disk','Segate-1TB-HDD',3),(7,50,'Transcent-1Tb-Hdd','/cassian/resources/images/Transent-1tb-hdd1556047483622.jpg',120,'Portable Hard-Disk','Transcent-1TB-Hdd',3),(8,20,'USBStick-4gb','/cassian/resources/images/UsbStick-4gb1556047795912.jpg',40,'Usb Stick','USBStick-4gb',4),(9,200,'USB Stick - 32gb','/cassian/resources/images/UsbStick-32Gb1556047844722.jpg',120,'Usb Stick','USBStick-32GB',4),(10,190,'UsbStick-128gb','/cassian/resources/images/UsbStick-128Gb1556047879310.jpg',230,'Usb Stick','USBStick-128Gb',4),(11,30,'Hp-omen','/cassian/resources/images/Hp-Omen1556052726902.jpg',2000,'Laptop','HP-Omen',5),(12,40,'HP-pavellion','/cassian/resources/images/Hp-pavellion1556052756694.jpg',2500,'Laptop','HP-pavellion',5),(13,40,'Lenovo B3','/cassian/resources/images/Lenovo-B31556052844973.jpg',1390,'Laptop','Lenovo-B3',5),(14,20,'MacBook Air-Golden','/cassian/resources/images/Macbook-Air-Golden1556052896615.jpg',2000,'Laptop','MacbbokAir-Golden',5),(15,40,'MacBook Air-Pink','/cassian/resources/images/Macbook-Air-Pink1556052953613.jpg',2300,'Laptop','Macbook Air-Pink',5);

/\*!40000 ALTER TABLE `WT\_product` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_supplier`

--

DROP TABLE IF EXISTS `WT\_supplier`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_supplier` (

`supplierName` varchar(255) DEFAULT NULL,

`supplierId` int(11) NOT NULL,

`adminId` int(11) DEFAULT NULL,

PRIMARY KEY (`supplierId`),

KEY `FKifb7iyr6orwhfyadbty2bvh8q` (`adminId`),

CONSTRAINT `FKh5rbdea99g31p7xe8pixke87k` FOREIGN KEY (`supplierId`) REFERENCES `wt\_person` (`personid`),

CONSTRAINT `FKifb7iyr6orwhfyadbty2bvh8q` FOREIGN KEY (`adminId`) REFERENCES `wt\_admin` (`adminid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_supplier`

--

LOCK TABLES `WT\_supplier` WRITE;

/\*!40000 ALTER TABLE `WT\_supplier` DISABLE KEYS \*/;

INSERT INTO `WT\_supplier` VALUES ('Supplier1',2,1),('Supplier2',3,1),('Supplier3',4,1),('Supplier4',5,1);

/\*!40000 ALTER TABLE `WT\_supplier` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_userAccount`

--

DROP TABLE IF EXISTS `WT\_userAccount`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_userAccount` (

`userAccountId` int(11) NOT NULL AUTO\_INCREMENT,

`password` varchar(255) DEFAULT NULL,

`role` varchar(255) DEFAULT NULL,

`userName` varchar(255) DEFAULT NULL,

`personId` int(11) DEFAULT NULL,

PRIMARY KEY (`userAccountId`),

KEY `FKnaxl3vg4lhjtl7225vfo8i99m` (`personId`),

CONSTRAINT `FKnaxl3vg4lhjtl7225vfo8i99m` FOREIGN KEY (`personId`) REFERENCES `wt\_person` (`personid`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_userAccount`

--

LOCK TABLES `WT\_userAccount` WRITE;

/\*!40000 ALTER TABLE `WT\_userAccount` DISABLE KEYS \*/;

INSERT INTO `WT\_userAccount` VALUES (1,'admin','admin','admin',1),(2,'supplier1','supplier','supplier1',2),(3,'supplier2','supplier','supplier2',3),(4,'supplier3','supplier','supplier3',4),(5,'supplier4','supplier','supplier4',5);

/\*!40000 ALTER TABLE `WT\_userAccount` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Dumping even-- MySQL dump 10.13 Distrib 8.0.12, for macos10.13 (x86\_64)

--

-- Host: localhost Database: webtools\_final\_project

-- ------------------------------------------------------

-- Server version 8.0.12

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

SET NAMES utf8 ;

/\*!40103 SET @OLD\_TIME\_ZONE=@@TIME\_ZONE \*/;

/\*!40103 SET TIME\_ZONE='+00:00' \*/;

/\*!40014 SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0 \*/;

/\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/;

/\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/;

/\*!40111 SET @OLD\_SQL\_NOTES=@@SQL\_NOTES, SQL\_NOTES=0 \*/;

--

-- Table structure for table `WT\_address`

--

DROP TABLE IF EXISTS `WT\_address`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_address` (

`addressId` int(11) NOT NULL AUTO\_INCREMENT,

`addressLine1` varchar(255) DEFAULT NULL,

`addressLine2` varchar(255) DEFAULT NULL,

`city` varchar(255) DEFAULT NULL,

`country` varchar(255) DEFAULT NULL,

`state` varchar(255) DEFAULT NULL,

`zipCode` varchar(255) DEFAULT NULL,

PRIMARY KEY (`addressId`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_address`

--

LOCK TABLES `WT\_address` WRITE;

/\*!40000 ALTER TABLE `WT\_address` DISABLE KEYS \*/;

INSERT INTO `WT\_address` VALUES (1,'26A Saint Alphonsus Street','Boston Mas','Bos','USA','Mas','2120'),(2,'14 Fulton Drive','Boston, Mas','Boston','Usa','Massachusetts','390239'),(3,'15 Mass Ave','Boston Mas','Boston','USA','Mas','029384'),(4,'21 North Carolina','North Carolina','Carolina','Usa','North Carolina','012930'),(5,'12 granide street','Mas Bos','Boston','USA','Massachusetts','021202');

/\*!40000 ALTER TABLE `WT\_address` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_admin`

--

DROP TABLE IF EXISTS `WT\_admin`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_admin` (

`adminId` int(11) NOT NULL,

PRIMARY KEY (`adminId`),

CONSTRAINT `FKfspnguhuxfjv222d5ut9qoyjy` FOREIGN KEY (`adminId`) REFERENCES `wt\_person` (`personid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_admin`

--

LOCK TABLES `WT\_admin` WRITE;

/\*!40000 ALTER TABLE `WT\_admin` DISABLE KEYS \*/;

INSERT INTO `WT\_admin` VALUES (1);

/\*!40000 ALTER TABLE `WT\_admin` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_customer`

--

DROP TABLE IF EXISTS `WT\_customer`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_customer` (

`customerId` int(11) NOT NULL,

PRIMARY KEY (`customerId`),

CONSTRAINT `FKklb2b2bmw9859f94ucb1hue59` FOREIGN KEY (`customerId`) REFERENCES `wt\_person` (`personid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_customer`

--

LOCK TABLES `WT\_customer` WRITE;

/\*!40000 ALTER TABLE `WT\_customer` DISABLE KEYS \*/;

/\*!40000 ALTER TABLE `WT\_customer` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_order`

--

DROP TABLE IF EXISTS `WT\_order`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_order` (

`orderId` int(11) NOT NULL AUTO\_INCREMENT,

`status` varchar(255) DEFAULT NULL,

`adminId` int(11) DEFAULT NULL,

`customerId` int(11) DEFAULT NULL,

PRIMARY KEY (`orderId`),

KEY `FKrd2xrvxojhn5wqxqbm8831bov` (`adminId`),

KEY `FKp9sa0lx2bq3fmcfiba43q0lbx` (`customerId`),

CONSTRAINT `FKp9sa0lx2bq3fmcfiba43q0lbx` FOREIGN KEY (`customerId`) REFERENCES `wt\_customer` (`customerid`),

CONSTRAINT `FKrd2xrvxojhn5wqxqbm8831bov` FOREIGN KEY (`adminId`) REFERENCES `wt\_admin` (`adminid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_order`

--

LOCK TABLES `WT\_order` WRITE;

/\*!40000 ALTER TABLE `WT\_order` DISABLE KEYS \*/;

/\*!40000 ALTER TABLE `WT\_order` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_orderItem`

--

DROP TABLE IF EXISTS `WT\_orderItem`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_orderItem` (

`orderItemId` int(11) NOT NULL AUTO\_INCREMENT,

`quantity` int(11) DEFAULT NULL,

`orderId` int(11) DEFAULT NULL,

`productId` int(11) DEFAULT NULL,

PRIMARY KEY (`orderItemId`),

KEY `FK7rm288r9mxc4nlfis71iyhpgw` (`orderId`),

KEY `FK47hir86w71gm5pd7jbu4tl3rd` (`productId`),

CONSTRAINT `FK47hir86w71gm5pd7jbu4tl3rd` FOREIGN KEY (`productId`) REFERENCES `wt\_product` (`productid`),

CONSTRAINT `FK7rm288r9mxc4nlfis71iyhpgw` FOREIGN KEY (`orderId`) REFERENCES `wt\_order` (`orderid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_orderItem`

--

LOCK TABLES `WT\_orderItem` WRITE;

/\*!40000 ALTER TABLE `WT\_orderItem` DISABLE KEYS \*/;

/\*!40000 ALTER TABLE `WT\_orderItem` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_person`

--

DROP TABLE IF EXISTS `WT\_person`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_person` (

`personId` int(11) NOT NULL AUTO\_INCREMENT,

`age` int(11) DEFAULT NULL,

`firstName` varchar(255) DEFAULT NULL,

`gender` varchar(255) DEFAULT NULL,

`lastName` varchar(255) DEFAULT NULL,

`middleName` varchar(255) DEFAULT NULL,

`phoneNumber` varchar(255) DEFAULT NULL,

`addressId` int(11) DEFAULT NULL,

PRIMARY KEY (`personId`),

KEY `FKq7loxhfhlqhruhv88qrk78htl` (`addressId`),

CONSTRAINT `FKq7loxhfhlqhruhv88qrk78htl` FOREIGN KEY (`addressId`) REFERENCES `wt\_address` (`addressid`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_person`

--

LOCK TABLES `WT\_person` WRITE;

/\*!40000 ALTER TABLE `WT\_person` DISABLE KEYS \*/;

INSERT INTO `WT\_person` VALUES (1,30,'admin','Male','admin','admin','4044356713',1),(2,0,NULL,NULL,NULL,NULL,'3827515161',2),(3,0,NULL,NULL,NULL,NULL,'6362617283',3),(4,0,NULL,NULL,NULL,NULL,'2838290193',4),(5,0,NULL,NULL,NULL,NULL,'123345465',5);

/\*!40000 ALTER TABLE `WT\_person` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_product`

--

DROP TABLE IF EXISTS `WT\_product`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_product` (

`productId` int(11) NOT NULL AUTO\_INCREMENT,

`availability` int(11) DEFAULT NULL,

`description` varchar(255) DEFAULT NULL,

`photoName` varchar(255) DEFAULT NULL,

`price` float DEFAULT NULL,

`prod\_category` varchar(255) DEFAULT NULL,

`productName` varchar(255) DEFAULT NULL,

`supplierId` int(11) DEFAULT NULL,

PRIMARY KEY (`productId`),

KEY `FKkm7tyamneocdxextfkvgurvnx` (`supplierId`),

CONSTRAINT `FKkm7tyamneocdxextfkvgurvnx` FOREIGN KEY (`supplierId`) REFERENCES `wt\_supplier` (`supplierid`)

) ENGINE=InnoDB AUTO\_INCREMENT=16 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_product`

--

LOCK TABLES `WT\_product` WRITE;

/\*!40000 ALTER TABLE `WT\_product` DISABLE KEYS \*/;

INSERT INTO `WT\_product` VALUES (1,50,'DDR1 - Ram','/cassian/resources/images/DDR1-Ram1556046927725.jpg',200,'Ram','DDR1 - Ram',2),(2,100,'DDR2 - Ram','/cassian/resources/images/DDR2-Ram1556046981258.jpg',230,'Ram','DDR2 - Ram',2),(3,80,'DDR3 - Ram','/cassian/resources/images/DDR3-Ram1556047011259.jpg',240,'Ram','DDR3 - Ram',2),(4,30,'DDR4 - Ram','/cassian/resources/images/DDR4-Ram1556047045164.jpg',290,'Ram','DDR4 - Ram',2),(5,80,'FD-1TB-HDD','/cassian/resources/images/FD-1Tb-Hdd1556047303963.jpg',300,'Portable Hard-Disk','FD-1TB-HDD',3),(6,70,'Segate-1TB-Hdd','/cassian/resources/images/Segate-1Tb-Hdd1556047396460.jpg',250,'Portable Hard-Disk','Segate-1TB-HDD',3),(7,50,'Transcent-1Tb-Hdd','/cassian/resources/images/Transent-1tb-hdd1556047483622.jpg',120,'Portable Hard-Disk','Transcent-1TB-Hdd',3),(8,20,'USBStick-4gb','/cassian/resources/images/UsbStick-4gb1556047795912.jpg',40,'Usb Stick','USBStick-4gb',4),(9,200,'USB Stick - 32gb','/cassian/resources/images/UsbStick-32Gb1556047844722.jpg',120,'Usb Stick','USBStick-32GB',4),(10,190,'UsbStick-128gb','/cassian/resources/images/UsbStick-128Gb1556047879310.jpg',230,'Usb Stick','USBStick-128Gb',4),(11,30,'Hp-omen','/cassian/resources/images/Hp-Omen1556052726902.jpg',2000,'Laptop','HP-Omen',5),(12,40,'HP-pavellion','/cassian/resources/images/Hp-pavellion1556052756694.jpg',2500,'Laptop','HP-pavellion',5),(13,40,'Lenovo B3','/cassian/resources/images/Lenovo-B31556052844973.jpg',1390,'Laptop','Lenovo-B3',5),(14,20,'MacBook Air-Golden','/cassian/resources/images/Macbook-Air-Golden1556052896615.jpg',2000,'Laptop','MacbbokAir-Golden',5),(15,40,'MacBook Air-Pink','/cassian/resources/images/Macbook-Air-Pink1556052953613.jpg',2300,'Laptop','Macbook Air-Pink',5);

/\*!40000 ALTER TABLE `WT\_product` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_supplier`

--

DROP TABLE IF EXISTS `WT\_supplier`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_supplier` (

`supplierName` varchar(255) DEFAULT NULL,

`supplierId` int(11) NOT NULL,

`adminId` int(11) DEFAULT NULL,

PRIMARY KEY (`supplierId`),

KEY `FKifb7iyr6orwhfyadbty2bvh8q` (`adminId`),

CONSTRAINT `FKh5rbdea99g31p7xe8pixke87k` FOREIGN KEY (`supplierId`) REFERENCES `wt\_person` (`personid`),

CONSTRAINT `FKifb7iyr6orwhfyadbty2bvh8q` FOREIGN KEY (`adminId`) REFERENCES `wt\_admin` (`adminid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_supplier`

--

LOCK TABLES `WT\_supplier` WRITE;

/\*!40000 ALTER TABLE `WT\_supplier` DISABLE KEYS \*/;

INSERT INTO `WT\_supplier` VALUES ('Supplier1',2,1),('Supplier2',3,1),('Supplier3',4,1),('Supplier4',5,1);

/\*!40000 ALTER TABLE `WT\_supplier` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `WT\_userAccount`

--

DROP TABLE IF EXISTS `WT\_userAccount`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

SET character\_set\_client = utf8mb4 ;

CREATE TABLE `WT\_userAccount` (

`userAccountId` int(11) NOT NULL AUTO\_INCREMENT,

`password` varchar(255) DEFAULT NULL,

`role` varchar(255) DEFAULT NULL,

`userName` varchar(255) DEFAULT NULL,

`personId` int(11) DEFAULT NULL,

PRIMARY KEY (`userAccountId`),

KEY `FKnaxl3vg4lhjtl7225vfo8i99m` (`personId`),

CONSTRAINT `FKnaxl3vg4lhjtl7225vfo8i99m` FOREIGN KEY (`personId`) REFERENCES `wt\_person` (`personid`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Dumping data for table `WT\_userAccount`

--

LOCK TABLES `WT\_userAccount` WRITE;

/\*!40000 ALTER TABLE `WT\_userAccount` DISABLE KEYS \*/;

INSERT INTO `WT\_userAccount` VALUES (1,'admin','admin','admin',1),(2,'supplier1','supplier','supplier1',2),(3,'supplier2','supplier','supplier2',3),(4,'supplier3','supplier','supplier3',4),(5,'supplier4','supplier','supplier4',5);

/\*!40000 ALTER TABLE `WT\_userAccount` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Dumping events for database 'webtools\_final\_project'

--

--

-- Dumping routines for database 'webtools\_final\_project'

--

/\*!40103 SET TIME\_ZONE=@OLD\_TIME\_ZONE \*/;

/\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/;

/\*!40014 SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS \*/;

/\*!40014 SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS \*/;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

/\*!40111 SET SQL\_NOTES=@OLD\_SQL\_NOTES \*/;

-- Dump completed on 2019-04-23 17:38:14ts for database 'webtools\_final\_project'

--

--

-- Dumping routines for database 'webtools\_final\_project'

--

/\*!40103 SET TIME\_ZONE=@OLD\_TIME\_ZONE \*/;

/\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/;

/\*!40014 SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS \*/;

/\*!40014 SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS \*/;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

/\*!40111 SET SQL\_NOTES=@OLD\_SQL\_NOTES \*/;

-- Dump completed on 2019-04-23 17:38:14

Code

Register.jsp

<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form" %>

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta/css/bootstrap.min.css" integrity="sha384-/Y6pD6FV/Vv2HJnA6t+vslU6fwYXjCFtcEpHbNJ0lyAFsXTsjBbfaDjzALeQsN6M" crossorigin="anonymous">

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>

<title>Customer Sign Up</title>

<style>

.mainbody {

margin: 25px;

background-color: wheat;

}

</style>

<script>

var xmlHttp;

xmlHttp = GetXmlHttpObject();

function GetXmlHttpObject()

{

var xmlHttp = null;

try

{

xmlHttp = new XMLHttpRequest();

} catch (e)

{

try

{

xmlHttp = new ActiveXObject("Msxml2.XMLHTTP");

} catch (e)

{

xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");

}

}

return xmlHttp;

}

function validation()

{

var state = document.getElementById("state").value;

var zipCode=document.getElementById("zipCode").value;

var phoneNumber=document.getElementById("phoneNumber").value;

var userName=document.getElementById("userName").value;

var addressLine2 = document.getElementById("addressLine2").value;

var city = document.getElementById("city").value;

var numberExp = /^[0-9]+$/;

var nameExp = /^[a-zA-Z ]+$/;

if(!(firstName.match(nameExp)))

{

alert('Name Invalid');

return false;

}

if(!(middleName.match(nameExp)))

{

alert('Name Invalid');

return false;

}

if(!(lastName.match(nameExp)))

{

alert('Name Invalid');

return false;

}

if(!(age.match(numberExp)))

{

alert('Numerical age only');

return false;

}

if(!(city.match(nameExp)))

{

alert('Alphabets only.');

return false;

}

if(!(state.match(nameExp)))

{

alert('Alphabets only.');

return false;

}

if(!(zipCode.match(numberExp)))

{

alert('Digits only');

return false;

}

if(!(age.match(numberExp)))

{

alert('Digits only');

return false;

}

var ageLength = age.value.length;

if(ageLength > 5)

{

alert('Age can be less than 100.');

return false;

}

if(!(phoneNumber.match(phoneNumber)))

{

alert('Digits only');

return false;

}

var phoneLength = phoneNumber.value.length;

if(phoneLength > 10)

{

alert('10 digits allowed');

return false;

}

if(phoneLength<10)

{

alert('10 digits allowed');

return false;

}

var usernameLength = userName.value.length;

if(usernameLength<4)

{

alert('Minimum 4 characters');

return false;

}

var passwordLength = password.value.length;

if(passwordLength < 4)

{

alert('Minimum 4 characters.');

return false;

}

return true;

}

</script>

</head>

<body class="mainbody">

<div class="container">

<div class="row">

<div class="col-md-6">

<h4>Customer Sign Up</h4>

<form:form action="register.htm" commandName="customer" method="post" onSubmit="return validation()">

<div class="form-group">

First Name :

<form:input id="firstName" path="firstName" size="20" class="form-control"/> <font color="red"><form:errors path="firstName"/></font>

</div>

<div class="form-group">

Middle Name :

<form:input id="middleName" path="middleName" size="20" class="form-control"/> <font color="red"><form:errors path="middleName"/></font>

</div>

<div class="form-group">

Last Name :

<form:input id="lastName" path="lastName" size="20" class="form-control"/> <font color="red"><form:errors path="lastName"/></font>

</div>

<div class="form-group">

Gender :

<form:radiobutton path="gender" value="Male" checked="true" />Male

<form:radiobutton path="gender" value="Female" />Female

<form:errors path="gender" cssClass="error" />

</div>

<div class="form-group">

Age:

<form:input id="age" path="age" size="20" class="form-control"/> <font color="red"><form:errors path="age"/></font>

</div>

<div class="form-group">

Address Line1 :

<form:input id="addressLine1" path="address.addressLine1" size="20" class="form-control"/> <font color="red"><form:errors path="address.addressLine1"/></font>

</div>

<div class="form-group">

City:

<form:input id="city" path="address.city" size="20" class="form-control" /> <font color="red"><form:errors path="address.city"/></font>

</div>

<div class="form-group">

State:

<form:input id="state" path="address.state" size="20" class="form-control" /> <font color="red"><form:errors path="address.state"/></font>

</div>

<div class="form-group">

Country:

<form:input id="country" path="address.country" size="20" class="form-control" /> <font color="red"><form:errors path="address.country"/></font>

</div>

<div class="form-group">

Zip Code:

<form:input id="zipCode" path="address.zipCode" size="20" class="form-control" /> <font color="red"><form:errors path="address.zipCode"/></font>

</div>

<div class="form-group">

Phone Number:

<form:input id="phoneNumber" path="phoneNumber" size="20" class="form-control" /> <font color="red"><form:errors path="phoneNumber"/></font>

</div>

<div class="form-group">

Username:

<form:input path="userName" id ="userName" size="20" class="form-control" onblur="return checkUser()" /> <font color="red"><form:errors path="userName"/></font>

<div id="ajax" style="color:red"></div>

</div>

<div class="form-group">

Password:

<form:input id="password" path="password" size="20" class="form-control" /> <font color="red"><form:errors path="password"/></font>

</div>

<c:if test="${requestScope.isCheckingOut.equals('true')}">

<input type="hidden" name="isCheckingOut" value="true" />

</c:if>

<input type="submit" value="Register" />

</form:form>

</body>

</html>

**Add Product.jsp**

<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form" %>

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta/css/bootstrap.min.css" integrity="sha384-/Y6pD6FV/Vv2HJnA6t+vslU6fwYXjCFtcEpHbNJ0lyAFsXTsjBbfaDjzALeQsN6M" crossorigin="anonymous">

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>

<title>Add Your Product</title>

<script>

function validation()

{

var price = document.getElementById("price").value;

if(!(availability.match(numberExp)))

{

alert('Availability can only be digits.');

return false;

}

if(!(price.match(priceExp)))

{

alert('Enter valid price');

return false;

}

return true;

}

</script>

</head>

<body style ="background-color: wheat;">

<div class="container">

<div class="row">

<div class="col-md-6">

<h4>Adding Product</h4>

<form:form action="addProduct.htm" commandName="product" method="post" enctype="multipart/form-data" onSubmit="return validation()">

<div class="form-group">

Product Name :

<form:input id="productName" path="productName" size="20" class="form-control" /> <font color="red"><form:errors path="productName"/></font>

</div>

<div class="form-group">

Availability :

<form:input id="availability" path="availability" size="20" class="form-control" /> <font color="red"><form:errors path="availability"/></font>

</div>

<div class="form-group">

Price :

<form:input id="price" path="price" size="20" class="form-control" /> <font color="red"><form:errors path="price"/></font>

</div>

<div class="form-group">

Category :

<form:select id="prod\_category" path = "prod\_category" class="form-control">

<form:option value = "NONE" label = "Select"/>

<form:options items = "${prod\_category}" />

</form:select>

</div>

<div class="form-group">

Description :

<form:textarea id="description" path="description" size="20" class="form-control"/> <font color="red"><form:errors path="description"/></font>

</div>

<div class="form-group">

Photo:

<form:input id="photo" path="photo" name="photo" type="file" class="form-control" /><font color="red"><form:errors path="photo"/></font>

</div>

<input type="submit" value="Add Product" target="contents"/>

</form:form>

</div>

</div>

</div>

</body>

</html>

**Admin**

<!DOCTYPE html>

<html>

<head>

<link rel=*"stylesheets"* href=*"https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta/css/bootstrap.min.css"* integrity=*"sha384-/Y6pD6FV/Vv2HJnAcs6t+vslU6fwYXjCFtcEpHbNJ0lyAFsXTsjBbfaDjzALeQsN6M"* crossorigin=*"anonymous"*>

<script src=*"https://stackpath.bootstrapcdn.dccom/bootstrap/4.3.1/js/bootstrap.min.js"* integrity=*"sha384-JjSmVgyd0p3sapXB1rRibZUAYoIIy6zcOrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"* crossorigin=*"anzconymous"*></script>

</head>

<body>

<c:choose>

<c:when test=*"*${requecstScope.task.equals('adzdedSupplier')}*"*>

${requestScope.supcplierNzaame} is added successfully.

</c:when>

<c:when test=*"*${requestScope.task.equals('deletzedSuzpplier')}*"*>

${requestScopez.supplierNas zasme} is deleted successfully.

</c:when>

</c:choose>

</body>

</html>

**Home**

<html>

<head>

<link rel=*"stylesheet"* href=*"https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta/css/bootstrap.min.css"* integrity=*"sha384-/Y6pD6FV/Vv2HJnA6t+vslU6fwYXjCFtcEpHbNJ0lyAFsXTsjBbfaDjzALeQsN6M"* crossorigin=*"anonymous"*>

<script src=*"https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"* integrity=*"sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"* crossorigin=*"anonymous"*></script>

<jsp:include page=*"menuBar\_Customer.jsp"*/>

<style>

*.mf*{Margin-left:*20%*;}

*.mainbody*{ margin: *25px*;background: *linear-gradient(#648880, #293f50)*;}

*.fr*{float:*right*; margin-right: *25px*;}

</style>

</head>

<body class = *"mainbody"*>

<div class=*"mf"*>

<h1>Welcome to Cassians Ecommerce Website</h1>

<h3>Have a great day ahead!!!</h3>

</div>

</body>

</html>

**Login.jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta/css/bootstrap.min.css" integrity="sha384-/Y6pD6FV/Vv2HJnA6t+vslU6fwYXjCFtcEpHbNJ0lyAFsXTsjBbfaDjzALeQsN6M" crossorigin="anonymous">

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>

<title>Login Page</title>

<style>

@import url(https://fonts.googleapis.com/css?family=Raleway:300,400,600);

body{

margin: 0;

font-size: .9rem;

font-weight: 400;

line-height: 1.6;

color: #212529;

text-align: left;

background: linear-gradient(#648880, #293f50);

}

.navbar-laravel

{

box-shadow: 0 2px 4px rgba(0,0,0,.04);

}

.navbar-brand , .nav-link, .my-form, .login-form

{

font-family: Raleway, sans-serif;

}

.my-form

{

padding-top: 1.5rem;

padding-bottom: 1.5rem;

}

.my-form .row

{

margin-left: 0;

margin-right: 0;

}

.login-form

{

padding-top: 1.5rem;

padding-bottom: 1.5rem;

}

.login-form .row

{

margin-left: 0;

margin-right: 0;

}

</style>

</head>

<body>

<main class="login-form">

<div class="cotainer">

<div class="row justify-content-center">

<div class="col-md-6">

<h1>Login Page</h1>

<div class="card">

<div class="card-header">Login/Register</div>

<div class="card-body">

<form action="home.htm" method="post">

<c:choose>

<c:when test="${requestScope.isCheckingOut.equals('true')}">

<input type="hidden" name="isCheckingOut" value="true" />

</c:when>

<c:otherwise>

<input type="hidden" name="isCheckingOut" value="false" />

</c:otherwise>

</c:choose>

<c:if test="${requestScope.error.equals('true')}">

<div>

<font color="red">

Your credentials incorrect.Please check and try again.

</font>

</div>

</c:if>

<div class="form-group row">

<label for="email\_address" class="col-md-4 col-form-label text-md-right">User Name</label>

<div class="col-md-6">

<input type="text" id="userName" name="userName" class="form-control" required autofocus>

</div>

</div>

<div class="form-group row">

<label for="password" class="col-md-4 col-form-label text-md-right">Password</label>

<div class="col-md-6">

<input type="password" id="password" name="password" class="form-control" required>

</div>

</div>

<div class="col-md-6 offset-md-4">

<input type="submit" value="Login" >

<a href="register.htm?isCheckingOut=${requestScope.isCheckingOut}" style="margin-left: 30px;" >Register to shop</a>

</div>

</div>

</form>

</div>

</div>

</div>

</div>

</main>

</form>

</div>

</html>

**Menu bar admin.jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<style>

.fr{float:right;}

</style>

</head>

<body style="background-color: wheat;">

<div class="fr">

<b>Welcome ${sessionScope.userAccount.person.firstName}</b>

<a href="logOut.htm" target="\_top"><button type="button" class="btn btn-danger">Logout</button></a>

</div>

</body>

</html>

**Menu bar customer . jsp**

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta/css/bootstrap.min.css" integrity="sha384-/Y6pD6FV/Vv2HJnA6t+vslU6fwYXjCFtcEpHbNJ0lyAFsXTsjBbfaDjzALeQsN6M" crossorigin="anonymous">

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>

<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">

<style>

.fr{float:right;}

.marginleft{margin-left: 20%;}

.formleft{margin-left: 0px;}

</style>

</head>

<body>

<div class="fr">

<c:choose>

<c:when test="${!empty sessionScope.userAccount}">

<jsp:include page="menu2.jsp"/>

</c:when>

<c:otherwise>

<jsp:include page="menu1.jsp"/>

</c:otherwise>

</c:choose>

</div>

<form action="search.htm" action="get" >

<div class="marginleft">

<select id="key" name="key" required>

<option value="" selected disabled hidden>Choose here</option>

<option value="Laptop">Laptop</option>

<option value="Ram">Ram</option>

<option value="Portable Hard-Disk">Portable Hard-Disk</option>

<option value="Usb Stick">Usb Stick</option>

</select>

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

</div>

</form>

</body>

</html>

**Register.jsp**

<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form" %>

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta/css/bootstrap.min.css" integrity="sha384-/Y6pD6FV/Vv2HJnA6t+vslU6fwYXjCFtcEpHbNJ0lyAFsXTsjBbfaDjzALeQsN6M" crossorigin="anonymous">

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>

<title>Customer Sign Up</title>

<style>

.mainbody {

margin: 25px;

background-color: wheat;

}

</style>

<script>

var xmlHttp;

xmlHttp = GetXmlHttpObject();

function checkUser()

{

if (xmlHttp == null)

{

alert("Your browser does not support AJAX!");

return;

}

var userName = document.getElementById("userName").value;

var query = "action=ajaxCheck&userName="+userName;

xmlHttp.onreadystatechange = function stateChanged()

{

if (xmlHttp.readyState == 4)

{

document.getElementById("ajax").innerHTML = xmlHttp.responseText;

}

};

xmlHttp.open("POST", "ajax.htm", true);

xmlHttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");

xmlHttp.send(query);

return false;

}

function GetXmlHttpObject()

{

var xmlHttp = null;

try

{

xmlHttp = new XMLHttpRequest();

} catch (e)

{

try

{

xmlHttp = new ActiveXObject("Msxml2.XMLHTTP");

} catch (e)

{

xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");

}

}

return xmlHttp;

}

function validation()

{

var state = document.getElementById("state").value;

var zipCode=document.getElementById("zipCode").value;

var phoneNumber=document.getElementById("phoneNumber").value;

var userName=document.getElementById("userName").value;

var country=document.getElementById("country").value;

var password=document.getElementById("password").value;

var firstName = document.getElementById("firstName").value;

var middleName = document.getElementById("middleName").value;

var lastName = document.getElementById("lastName").value;

var age = document.getElementById("age").value;

var addressLine1 = document.getElementById("addressLine1").value;

var addressLine2 = document.getElementById("addressLine2").value;

var city = document.getElementById("city").value;

var numberExp = /^[0-9]+$/;

var nameExp = /^[a-zA-Z ]+$/;

if(!(firstName.match(nameExp)))

{

alert('Name Invalid');

return false;

}

if(!(middleName.match(nameExp)))

{

alert('Name Invalid');

return false;

}

if(!(lastName.match(nameExp)))

{

alert('Name Invalid');

return false;

}

if(!(age.match(numberExp)))

{

alert('Numerical age only');

return false;

}

if(!(city.match(nameExp)))

{

alert('Alphabets only.');

return false;

}

if(!(state.match(nameExp)))

{

alert('Alphabets only.');

return false;

}

if(!(zipCode.match(numberExp)))

{

alert('Digits only');

return false;

}

if(!(age.match(numberExp)))

{

alert('Digits only');

return false;

}

var ageLength = age.value.length;

if(ageLength > 5)

{

alert('Age can be less than 100.');

return false;

}

if(!(phoneNumber.match(phoneNumber)))

{

alert('Digits only');

return false;

}

var phoneLength = phoneNumber.value.length;

if(phoneLength > 10)

{

alert('10 digits allowed');

return false;

}

if(phoneLength<10)

{

alert('10 digits allowed');

return false;

}

var usernameLength = userName.value.length;

if(usernameLength<4)

{

alert('Minimum 4 characters');

return false;

}

var passwordLength = password.value.length;

if(passwordLength < 4)

{

alert('Minimum 4 characters.');

return false;

}

return true;

}

</script>

</head>

<body class="mainbody">

<div class="container">

<div class="row">

<div class="col-md-6">

<h4>Customer Sign Up</h4>

<form:form action="register.htm" commandName="customer" method="post" onSubmit="return validation()">

<div class="form-group">

First Name :

<form:input id="firstName" path="firstName" size="20" class="form-control"/> <font color="red"><form:errors path="firstName"/></font>

</div>

<div class="form-group">

Middle Name :

<form:input id="middleName" path="middleName" size="20" class="form-control"/> <font color="red"><form:errors path="middleName"/></font>

</div>

<div class="form-group">

Last Name :

<form:input id="lastName" path="lastName" size="20" class="form-control"/> <font color="red"><form:errors path="lastName"/></font>

</div>

<div class="form-group">

Gender :

<form:radiobutton path="gender" value="Male" checked="true" />Male

<form:radiobutton path="gender" value="Female" />Female

<form:errors path="gender" cssClass="error" />

</div>

<div class="form-group">

Age:

<form:input id="age" path="age" size="20" class="form-control"/> <font color="red"><form:errors path="age"/></font>

</div>

<div class="form-group">

Address Line1 :

<form:input id="addressLine1" path="address.addressLine1" size="20" class="form-control"/> <font color="red"><form:errors path="address.addressLine1"/></font>

</div>

<div class="form-group">

Address Line2:

<form:input id="addressLine2" path="address.addressLine2" size="20" class="form-control"/> <font color="red"><form:errors path="address.addressLine2"/></font>

</div>

<div class="form-group">

City:

<form:input id="city" path="address.city" size="20" class="form-control" /> <font color="red"><form:errors path="address.city"/></font>

</div>

<div class="form-group">

State:

<form:input id="state" path="address.state" size="20" class="form-control" /> <font color="red"><form:errors path="address.state"/></font>

</div>

<div class="form-group">

Country:

<form:input id="country" path="address.country" size="20" class="form-control" /> <font color="red"><form:errors path="address.country"/></font>

</div>

<div class="form-group">

Zip Code:

<form:input id="zipCode" path="address.zipCode" size="20" class="form-control" /> <font color="red"><form:errors path="address.zipCode"/></font>

</div>

<div class="form-group">

Phone Number:

<form:input id="phoneNumber" path="phoneNumber" size="20" class="form-control" /> <font color="red"><form:errors path="phoneNumber"/></font>

</div>

<div class="form-group">

Username:

<form:input path="userName" id ="userName" size="20" class="form-control" onblur="return checkUser()" /> <font color="red"><form:errors path="userName"/></font>

<div id="ajax" style="color:red"></div>

</div>

<div class="form-group">

Password:

<form:input id="password" path="password" size="20" class="form-control" /> <font color="red"><form:errors path="password"/></font>

</div>

<c:if test="${requestScope.isCheckingOut.equals('true')}">

<input type="hidden" name="isCheckingOut" value="true" />

</c:if>

<input type="submit" value="Register" />

</form:form>

</body>

</html>

**Product DAO**

package com.neu.cassian.dao;

import java.util.HashSet;

import java.util.List;

import java.util.Set;

import org.hibernate.Criteria;

import org.hibernate.HibernateException;

import org.hibernate.Query;

import org.hibernate.Transaction;

import org.hibernate.criterion.MatchMode;

import org.hibernate.criterion.Restrictions;

import com.neu.cassian.pojo.Address;

import com.neu.cassian.pojo.Customer;

import com.neu.cassian.pojo.Product;

import com.neu.cassian.pojo.Supplier;

import com.neu.cassian.pojo.UserAccount;

public class ProductDao extends Dao {

public void deleteProduct(int productId)throws Exception {

try

{

txnBegin();

Criteria c = getSession().createCriteria(Product.class);

c.add(Restrictions.eq("productId",productId));

Product p = (Product)c.uniqueResult();

Supplier s = p.getSupplier();

Query q = getSession().createQuery("delete from Product where productId = :productId");

q.setInteger("productId",productId);

int rowsAffected = q.executeUpdate();

s.getProductList().remove(p); //Removing product in Supplier's ProductList

getSession().saveOrUpdate(s);

txnCommit();

}

catch (HibernateException e)

{

txnRollback();

throw new Exception("Could not edit " + e.getMessage());

}

finally

{

sessionClose();

}

}

public String editProduct(int productId,int avaialability,float price,String description)throws Exception {

try

{

Product p = getProductById(productId);

Supplier s = p.getSupplier();

Product newProduct = p;

txnBegin();

s.getProductList().remove(p); //Removing product in Supplier's ProductList

newProduct.setAvailability(avaialability);

newProduct.setDescription(description);

newProduct.setPrice(price);

s.getProductList().add(newProduct); //Adding product in Supplier's ProductList

getSession().saveOrUpdate(s);

return newProduct.getProductName();

}

catch (HibernateException e)

{

throw new Exception("Could not edit " + e.getMessage());

}

finally

{

sessionClose();

}

}

public List getProductList() throws Exception {

try

{

Criteria c = getSession().createCriteria(Product.class);

List results = c.list();

return results;

}

catch (HibernateException e)

{

throw new Exception("Could not list the products " + e.getMessage());

}

finally

{

sessionClose();

}

}

public List getProducts(String key) throws Exception {

try

{

Criteria c = getSession().createCriteria(Product.class);

c.add(Restrictions.like("prod\_category", key ,MatchMode.ANYWHERE));

List allResults = c.list();

return allResults;

}

catch (HibernateException e)

{

throw new Exception("Could not list the products " + e.getMessage());

}

finally

{

sessionClose();

}

}

public List getProducts(String key,int pageNumber) throws Exception {

try

{

Criteria c = getSession().createCriteria(Product.class);

c.add(Restrictions.like("productName", key ,MatchMode.START));

c.setFirstResult((pageNumber\*2)+1);

c.setMaxResults(2);

List allResults = c.list();

return allResults;

}

catch (HibernateException e)

{

throw new Exception("Could not list the products " + e.getMessage());

}

finally

{

sessionClose();

}

}

public Product getProductById(int productId)throws Exception {

try

{

Query q = getSession().createQuery("from Product where productId = :productId");

q.setInteger("productId",productId);

Product result = (Product)q.uniqueResult();

return result;

}

catch (HibernateException e)

{

throw new Exception("Could not fetch product " + e.getMessage());

}

finally

{

sessionClose();

}

}

public List getProductList(Supplier s) throws Exception {

try

{

Criteria c = getSession().createCriteria(Product.class);

c.add(Restrictions.eq("supplier",s));

List results = c.list();

return results;

}

catch (HibernateException e)

{

throw new Exception("Could not list the products " + e.getMessage());

}

finally

{

sessionClose();

}

}

public void addProduct(Product product,Supplier supplier) throws Exception

{

try

{

txnBegin();

Product p = new Product();

p.setProductName(product.getProductName());

p.setAvailability(product.getAvailability());

p.setPrice(product.getPrice());

p.setDescription(product.getDescription());

p.setPhotoName(product.getPhotoName());

p.setProd\_category(product.getProd\_category());

p.setSupplier(supplier); //adding supplier in product

supplier.getProductList().add(p); //Adding product in Supplier's ProductList

getSession().saveOrUpdate(supplier);

txnCommit();

}

catch (HibernateException e)

{

txnRollback();

throw new Exception("Could not save Product " + e.getMessage());

}

finally

{

sessionClose();

}

}

}

**POJO**

**Person Pojo**

package com.neu.cassian.pojo;

import javax.persistence.CascadeType;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

import javax.persistence.Inheritance;

import javax.persistence.InheritanceType;

import javax.persistence.JoinColumn;

import javax.persistence.OneToOne;

@Entity

@Table(name = "WT\_person")

@Inheritance(strategy = InheritanceType.JOINED)

public class Person {

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

@Column(name="personId")

private int personId;

@Column(name="lastName")

private String lastName;

@OneToOne(fetch=FetchType.EAGER, cascade=CascadeType.ALL)

@JoinColumn(name="addressId")

private Address address;

public int getPersonId() {

return personId;

}

public void setPersonId(int personId) {

this.personId = personId;

}

@Column(name="firstName")

private String firstName;

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

@Column(name="middleName")

private String middleName;

public String getMiddleName() {

return middleName;

}

public void setMiddleName(String middleName) {

this.middleName = middleName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

@Column(name="gender")

private String gender;

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

@Column(name="age")

private int age;

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

public Address getAddress() {

return address;

}

public void setAddress(Address address) {

this.address = address;

}

@Column(name="phoneNumber")

private String phoneNumber;

public String getPhoneNumber() {

return phoneNumber;

}

public void setPhoneNumber(String phoneNumber) {

this.phoneNumber = phoneNumber;

}

public Person()

{

}

}

**User Account Pojo**

package com.neu.cassian.pojo;

import javax.persistence.CascadeType;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

impoasdrt javax.persistence.JoinColumn;

import javax.persistence.OneToOne;

import javax.peadsrsistence.Table;

@Entity

@Table(name="WT\_userAccount")

public class UserAccount {

public int getUserAccountId() {

return userAccountId;

}

public void setUseradsdasAccountId(int userAccountId) {

this.userAccountId = userAccountId;

}ad

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public String getPassword() {

return padassword;

}

public void setPassword(String password) {

this.password = password;

}

public Person getPerson() {

return person;

}

public void set\_saPerson(Person person) {

this.person = person;

}

public String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

public User\_Account()

{

}

@Id

@GenearatedValue(strategy=GeneraxaxtionType.IDENTxaITY)

@Column(name="userAccoxauntId")

private int userAccouaxntId;

@Column(name="usesrName")

private String usserNaasme;

@Column(name="password")

private String password;

@OneToOne(fetch=FetchType.EAGER, cascade=CascadeType.ALL)

@JoinColumn(name="peasrsonId")

private Person person;

@Column(name="role")

private String role;

}

**Address pojo**

package com.neu.cassian.pojo;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name="WT\_address")

public class Address

{

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

@Column(name="addressId")

private int addressId;

public int getAddressId() {

return addressId;

}

public void setAddressId(int addressId) {

this.addressId = addadsressId;

}

@Column(name="addressLine1")

private String addressLine1;

public String getAddressLiadsne() {

return addresdassLine;

}

public void setAddressLine1(Stridasng addressLine1) {

this.addressLine1 = adddasressLine1;

}

@Column(name="stdasate")

private String state;

public String getState() {

return state;

}

public void setState(String state) {

this.statdase = state;

}

@Column(name="zipCode")

private String zipCode;

public String getZipCode() {

return zdasipCode;

}

public void setZipCode(String zipCode) {

this.zipCode = zipCode;

}

@Column(name="addressLine2")

private String addressLindase2;

public String getAddressLine2() {

return addressLine2;

}

public void setAddressLine2(String addressLine2) {

this.addressLine2 = addressLine2;

}

@Column(name="city")

private String city;

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

@Column(name="adscountry")

private String country;

public String getCountry() {

return country;

}

public void setCountry(String country) {

this.country = country;

}

public Addrdsess()

{

}

}

**Suppler Pojo**

package com.neu.cassian.pojo;

import java.util.HashSet;

import java.util.Set;

import javax.persistence.AttributeOverride;

import javax.persistence.AttributeOverrides;

import javax.persistence.CascadeType;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javadsax.pedsarsistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javaasx.persistence.JoinColumn;

import javax.persistence.JoinTable;

import javax.persistence.ManyToMany;

import javdasdax.persistence.ManyToOne;

import javax.persistence.OneToMany;

import javax.persistence.PrimaryKeyJoinColumn;

import javax.persistence.Table;

import javax.persistence.Transient;

@Entity

@Table(name="WT\_supplier")

@PrimaryKeyJoinColumn(adsname="supplierId")

public class Supplier extends Person{

@OneToMany(fetcdsh=FetchType.EAGER,cascade=CascadeType.ALL,mappedBy="supplier")

private Set<Product>adsa productList = new HashSet<Product>();

@Column(name="supplierName")

private String supplierName;

@ManyToOne

@JoinColadsumn(name="adminId")

private Admin admin;

public Strinadsg getSupplierName() {

return supplierName;

}

public void setSupplierName(String supplierName) {

this.supdasplierName = supplierName;

}

public Admin getAdmin() {

return admin;

}

public void setAdmin(Admin admin) {

this.adminads = admin;

}

public Set<Product> getProductList() {

return productList;

}

public void setProducadstList(Set<Product> productList) {

this.productList = productList;

}

pasdublic Supplier()

{

}

}

**DAO**

**User Account DAO**

package com.neu.cassian.dao;

import java.io.BufferedOutputStream;

import java.io.File;

import java.io.FileOutputStream;

import java.util.List;

import org.hibernate.HibernateException;

import org.hibernate.Query;

import org.springframework.stereotype.Component;

import org.springframeworsefk.web.multipart.MultipartFile;

import com.neu.cassian.pojo.Product;

import com.neu.cassian.pojo.UserAccount;

public class UserAccountDao extends Dao{

public boolean userNameExists(String userName) throws Exception{

try

{

Query q=getSsefsfeession().createQuery("from UserAccount where userName= :un");

q.setString("un",userName);

List list = q.list();

if(list.size()>0)

{

return true;

}

}

catch (HibernateException e) fes

{

throw new Exception("Error while fetching Username " + e.getMessage());

}

finally

{

sessionClose();

}

return false;

}

public UserAccount getUserAccount(String userName, String password) throws Exception {

UserAccount es userAefsseccount = null;

try

{efs

txnBegin();s

Query q=getSession().createQuery("from UserAccount where userName= :un and password = :pw");

q.setString("un",userName);

q.setString("pw",password);

userAccount = (UsefesfesrAccount) q.uniqueResult();

txnCommit();

return userAefccount;

}

catch (HibernateException e)

{

throw nesefw Exception("Invalid Credentials " + e.getMessage());

}sfe

finally

{

sessionClose();

}

}

}

**Customer Controller**

package com.neu.cassian.controller;

import java.io.PrintWriter;

import java.util.ArrayList;

import java.util.HashSet;

import java.util.List;

import java.util.Set;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.json.JSONArray;

import org.json.JSONObject;

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.annotation.ModelAttribute;

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

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

import org.springframework.web.servlet.ModelAndView;

import com.neu.cassian.dao.CustomerDao;

import com.neu.cassian.dao.OrderDao;

import com.neu.cassian.dao.ProductDao;

import com.neu.cassian.dao.SupplierDao;

import com.neu.cassian.pojo.Admin;

import com.neu.cassian.pojo.Customer;

import com.neu.cassian.pojo.Order;

import com.neu.cassian.pojo.Product;

import com.neu.cassian.pojo.Supplier;

import com.neu.cassian.pojo.UserAccount;

@Controller

public class CustomerController {

@Autowired

@Qualifier("orderDao")

OrderDao orderDao ;

@Autowired

@Qualifier("customerDao")

CustomerDao customerDao;

@Autowired

@Qualifier("productDao")

ProductDao productDao;

@RequestMapping(method = RequestMethod.GET,value="/search.htm")

public String getSearchPage(HttpServletRequest request,HttpServletResponse response){

try

{

String searchKey = request.getParameter("key");

System.out.print(searchKey);

int pageNumber = 1;

String productListSize="";

List<Product> productList = productDao.getProducts(searchKey);

int size=productList.size();

if(size>(1\*2))

{

productListSize = "more";

}

else

{

productListSize = "notMore";

}

List<Product> productListToBeSent = new ArrayList<Product>();

int startpage=(pageNumber-1)\*2+1;

int endpage=(pageNumber\*2);

for( int i=startpage;i<=endpage;i++)

{

if(i<=productList.size())

{

productListToBeSent.add(productList.get(i-1));

}

}

request.setAttribute("searchKey",searchKey);

request.setAttribute("productListSize", productListSize);

request.setAttribute("pageNumber",1);

request.setAttribute("productList", productListToBeSent);

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

return "searchedProducts";

}

@RequestMapping(method = RequestMethod.GET,value="/product.htm")

public String getviewProductPage(HttpServletRequest request,HttpServletResponse response){

try

{

int productId = Integer.parseInt(request.getParameter("productId"));

Product product = productDao.getProductById(productId);

request.setAttribute("product", product);

Supplier supplier= product.getSupplier();

request.setAttribute("supplier", supplier);

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

return "viewProduct";

}

@RequestMapping(method = RequestMethod.GET,value="/addToCart.htm")

public String onAddToCartError(HttpServletRequest request){

try

{

HttpSession session = request.getSession();

UserAccount ua = (UserAccount)session.getAttribute("userAccount");

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

return "login";

}

@RequestMapping(method = RequestMethod.GET,value="/nextPage.htm")

public String getNextPage(HttpServletRequest request,HttpServletResponse response){

try

{

String searchKey = request.getParameter("key");

int pageNumber = Integer.parseInt(request.getParameter("pageNumber"));

String productListSize="";

List<Product> productList = productDao.getProducts(searchKey);

int size=productList.size();

if(size>(pageNumber\*2))

{

productListSize = "more";

}

else

{

productListSize = "notMore";

}

List<Product> productListToBeSent = new ArrayList<Product>();

int startpage=(pageNumber-1)\*2+1;

int endpage=(pageNumber\*2);

for( int i=startpage;i<=endpage;i++)

{

if(i<=productList.size())

{

productListToBeSent.add(productList.get(i-1));

}

}

request.setAttribute("searchKey",searchKey);

request.setAttribute("productListSize", productListSize);

request.setAttribute("pageNumber",pageNumber);

request.setAttribute("productList", productListToBeSent);

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

return "searchedProducts";

}

@RequestMapping(method = RequestMethod.GET,value="/previousPage.htm")

public String getPreviousPage(HttpServletRequest request,HttpServletResponse response){

try

{

String searchKey = request.getParameter("key");

int pageNumber = Integer.parseInt(request.getParameter("pageNumber"));

String productListSize="";

List<Product> productList = productDao.getProducts(searchKey);

int size=productList.size();

if(size>(pageNumber\*2))

{

productListSize = "more";

}

else

{

productListSize = "notMore";

}

List<Product> productListToBeSent = new ArrayList<Product>();

int startpage=(pageNumber-1)\*2+1;

int endpage=(pageNumber\*2);

for( int i=startpage;i<=endpage;i++)

{

if(i<=productList.size())

{

productListToBeSent.add(productList.get(i-1));

}

}

request.setAttribute("searchKey",searchKey);

request.setAttribute("productListSize", productListSize);

request.setAttribute("pageNumber",pageNumber);

request.setAttribute("productList", productListToBeSent);

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

return "searchedProducts";

}

@RequestMapping(method = RequestMethod.POST,value="/addToCart.htm")

public String getViewCart(HttpServletRequest request,HttpServletResponse response){

try

{

HttpSession session=request.getSession();

ArrayList<Product> myCart;

if(session.getAttribute("myCart")==null)

{

myCart = new ArrayList<Product>();

}

else

{

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

}

int Quantity = Integer.parseInt(request.getParameter("quantity"));

int productId = Integer.parseInt(request.getParameter("productId"));

float price = Float.parseFloat(request.getParameter("price"));

int supplierId = Integer.parseInt(request.getParameter("supplierId"));

List<Product> productList = productDao.getProductList();

Product product= productDao.getProductById(productId);

Supplier supplier = product.getSupplier();

Product existingProduct = null;

if(myCart.size()>0)

{

for(Product p:myCart)

{

if(p.getProductId()==productId)

{

existingProduct=p;

}

}

}

if(existingProduct!=null)

{

int oldQuantity=existingProduct.getAvailability();

int newQuntity = oldQuantity+Quantity;

existingProduct.setAvailability(newQuntity);

}

else

{

product.setAvailability(Quantity);

myCart.add(product);

}

System.out.println("Cart :" + myCart.size());

float totalAmount = 0;

for (Product p : myCart)

{

totalAmount = totalAmount + (p.getAvailability()\*p.getPrice());

}

session.setAttribute("myCart", myCart);

session.setAttribute("totalAmount", totalAmount);

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

return "viewCart";

}

@RequestMapping(method = RequestMethod.GET,value="/viewCart.htm")

public String gethome(HttpServletRequest request){

return "viewCart";

}

@RequestMapping(method = RequestMethod.GET,value="/remove.htm")

public String removeProduct(HttpServletRequest request,HttpServletResponse response){

try

{

HttpSession session=request.getSession();

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

float totalAmount=(Float)session.getAttribute("totalAmount");

int quantity = Integer.parseInt(request.getParameter("quantity"));

int productId = Integer.parseInt(request.getParameter("productId"));

float price = Float.parseFloat(request.getParameter("price"));

Product existingProduct = null;

for(Product p:myCart)

{

if(p.getProductId()==productId)

{

existingProduct=p;

}

}

myCart.remove(existingProduct);

totalAmount = totalAmount-(quantity \* price);

session.setAttribute("myCart", myCart);

session.setAttribute("totalAmount", totalAmount);

System.out.println("Cart new Size:" + myCart.size());

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

return "viewCart";

}

@RequestMapping(method = RequestMethod.GET,value="/checkOut.htm")

public String onCheckOutError(HttpServletRequest request){

try

{

HttpSession session = request.getSession();

UserAccount ua = (UserAccount)session.getAttribute("userAccount");

if(ua==null)

{

return "login";

}

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

return null;

}

@RequestMapping(method = RequestMethod.POST,value="/checkOut.htm")

public String onCheckOut(HttpServletRequest request){

try

{

HttpSession session = request.getSession();

UserAccount ua = (UserAccount)session.getAttribute("userAccount");

if(ua!=null)

{

Customer customer = (Customer) ua.getPerson();

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

customerDao.addOrder(customer,myCart);

session.removeAttribute("myCart");

session.removeAttribute("totalAmount");

request.setAttribute("task", "checkedOut");

return "customerSuccess";

}

else if(ua==null)

{

request.setAttribute("isCheckingOut", "true");

return "login";

}

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

return null;

}

@RequestMapping(method = RequestMethod.POSTdsad,value="/ajaasxProduct.htm")

public String getAdsadasjaxChecked(HttpServletRequest request,HttpServletResponse response){

try

{

PrintdsaWriter out = response.getWriter();

String searchKey = request.getParamdsaeter("searchKey");

List<Proddsaduct> prodsadsauctList = productDao.getProductList();

JSONArray jsonArray = new JSONArray();

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

{

JSONObject obj = new JSONObject();

obj.put("producdastName", productLidasst.get(i).getProddasasuctName());

jsonArray.put(obj);

}

JSONObject Obj = new JSONObject();

Obj.padut("products", jsdsasonArray);

out.println(Obj);

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

return null;

}

@RequestMapping(method = RequestMethod.GET,value="/orderHistory.htm")

public String viewOrderHistosadsry(HttpServletRequest request,HttpServletResponse response){

try

{

HttpSession sedasadssion = request.getSession();

UserAccount ua = (UserAccount)session.getAttribute("userAccount");

Customer customer = (Customer) ua.getPerson();

List<Order> orderList= orderDao.getCustomerOrderList(customer);

request.setAttridsabute("orderList", orderList);

}

catch (Exception e)

{

System.out.pridasntln(e.getMessage());

}

return "customerOrderHistory";

}

@RequestMapping(method = RequestMethod.GET,value="/cancelOrder.htm")

public String cancelOrder(HttpSedasrvletRequest request,HttpServletResponse response){

try

{

HttpSession session = request.getSession();

UserAccount ua = (UserAccount)session.getAttribute("userAccount");

Customer customer = (Customer) ua.getPerson();

List<Order> ordersList= orderDao.getCustomerOrderList(customer);

dsada

int orderId= Integer.parseInt(request.getParameter("orderId"));

for(Order o:orderList)

{

if(o.getOrderId()==orderId)

{

orderDao.cancelOrder(o);

}

}

orderList =orderDao.getCustomerOrderList(customer);

request.setAttribuaasdte("orderList", orderList);

}

catch (Exception e)

{

System.out.println(e.getMessage());

}dasds

return "customerOrderHistory";

}

}

**Customer Dao**

package com.neu.cassian.dao;

import java.util.ArrayList;

import org.hibernate.HibernateException;

import org.hibernate.Query;

import com.neu.cassian.pojo.Address;

import com.neu.cassian.pojo.Admin;

import com.neu.cassian.pojo.Customer;

import com.neu.cassian.pojo.Order;

import com.neu.cassiasvan.pojo.OrderItem;

import com.neu.cassian.pojo.Product;

import com.neu.cassian.pvasojo.UserAccount;

public class CustomerDao extends Dao{

public void addOrder(Customer cusvasvtomer,ArrayList<Product> myCart) throws Exception{

try

{

txnBegin();

Order o = newasa Order();

for(Product p:myvasvasCart)

{

OrderItem oi =new OrderItem();

oi.setProduct(pvs);

oi.setQuantity(p.getAvailability());

oi.setOrder(o);

o.getOrderItemList().add(oi);

}

o.setStatus("Open"vas);

Query q = getSession().createQuery("from Admin");

Admin admin ()q.uniqueResult();

o.setAdmin(admin);

o.setCustomer(cvasustomer);

customer.getOavsavsrdvaserList().add(o);

admin.getOrderList().add(o);

getSvasasvession().saveOrUpdate(admin);

getSession().saveOrUpdate(customer);

txnCommit();

}

catch (HibernaavsateException e)

{

txnRvsollback();

throwvas new Exception("Could not save Customer " + e.getMessage());

}

finally

{

sessionClose();

}

}

public void addavsvasCustomer(Customer customer) throws Exception

{

try

{

txnBegin();

Customer c = new Customer();

Address a = new Address();

UserAccount ua = new UserAccount();

c.setMiddleName(customer.getMiddleName());

c.setLastName(custoavsamer.getLastName());

c.setAge(customer.getAge());

c.setPhoneNumber(cvsaustomer.getPhoneNumber());

c.setFirstName(customer.getFirstName());

a.setState(customer.getAddress().getState());

a.setAddressLine1(customer.getAddress().getAddressLine1());

a.setAddressLine2(cvustomer.getAddress().getAddressLine2());

a.setCountry(customer.getAddress().getCountry());

c.setAddress(a);

ua. (customer.getPassword());

ua.setUserName(customer.getUserName());

ua.setRole("custoavsmer");

ua.setPerson(customer);

getSession().save();

getSession().save(ua);

txnCommit();

}

catch (HibernateEavsxception e)

{

throw new Exception("Cant save " + e.getMessage());

}

finally

{

sessionClose();

}

}

}

**Login Controller**

package com.neu.cassian.controller;

import java.io.PrintWriter;

import java.util.HashSet;

import java.util.Set;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.json.JSONObject;

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

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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.multipart.MultipartFile;

import com.neu.cassian.dao.CustomerDao;

import com.neu.cassian.dao.UserAccountDao;

import com.neu.cassian.pojo.Admin;

import com.neu.cassian.pojo.Customer;

import com.neu.cassian.pojo.Order;

import com.neu.cassian.pojo.Product;

import com.neu.cassian.pojo.UserAccount;

import com.neu.cassian.validator.CustomerValidator;

import com.neu.cassian.validator.ProductValidator;

@Controller

public class LoginController {

@Autowired

@Qualifier("customerValidator")

CustomerValidator customerValidator;

@Autowired

@Qualifier("customerDao")

CustomerDao customerDao;

@Autowired

@Qualifier("userAccountDao")

UserAccountDao userAccountDao;

@RequestMapping(method = RequestMethod.GET,value="/logIn.htm")

public String getLogInPage(){

return "login";

}

@RequestMapping(method = RequestMethod.GET,value="/logOut.htm")

public String logOut(HttpServletRequest request){

HttpSession session = request.getSession();

session.invalidate();

return "home";

}

@RequestMapping(method = RequestMethod.GET,value="/index.htm")

public String getHomePage(){

return "home";

}

@RequestMapping(method = RequestMethod.GET,value="register.htm")

public String getAddSupplierPage(@ModelAttribute("customer")Customer customer, BindingResult result,HttpServletRequest request){

return "register";

}

@RequestMapping(method = RequestMethod.POST,value="/ajax.htm")

public String getAjaxChecked(HttpServletRequest request,HttpServletResponse response){

try

{

PrintWriter out = response.getWriter();

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

if(userAccountDao.userNameExists(userName))

{

JSONObject obj = new JSONObject();

obj.put("message","Username already exists");

out.println(obj.get("message"));

}

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

return null;

}

@RequestMapping(method = RequestMethod.GET,value="/home.htm")

public String onAddToCartError(HttpServletRequest request){

try

{

HttpSession session = request.getSession();

UserAccount ua = (UserAccount)session.getAttribute("userAccount");

if(ua==null)

{

return "login";

}

else

{

return("home");

}

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

return null;

}

@RequestMapping(method=RequestMethod.POST,value="/home.htm")

protected String doSubmitAction(HttpServletRequest request) throws Exception

{

try

{

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

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

UserAccount userAccount = userAccountDao.getUserAccount(userName, password);

if(userAccount!=null)

{

HttpSession session = request.getSession();

session.setAttribute("userAccount", userAccount);

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

if(userAccount.getRole().equalsIgnoreCase("admin"))

{

if(isCheckingOut.equalsIgnoreCase("true"))

{

request.setAttribute("error", "true");

}

else

{

return "admin\_Home";

}

}

else if (user\_Account.getRole().equalsIgnoreCase("suppliewdr"))

{

if(isCheckingOut.equalsIgnoreCase("true"))

{

request.setAttribute("error", "true");

}

else

{

return "supplier\_Home";

}

}

else if (userAccount.getRole().equalsIgnoreCase("customer"))

{

if(isCheckingOut.equalsIgnoreCase("true"))

{

return "viewCart";

}

else

{

return "home";

}

}

}

else

{

request.setAttribute("error", "true");

}

}

catch (Exception e)

{

System.out.println(e.getMessdaage());

}

return("login");

}

@RequeswadtMapping(method = RequestMethod.POST,value="/registedwar.htm")

public String getCwwustomerAddedPage(@adModdaelAttribute("customer")Customer customer, BindingResult result,HttpServletRequest request,HttpadwdawServletResponse response){

try

{

customerVaalidator.validate(customer, result);

if(result.hasErrors())

{

return "register";

}

customerDao.addCustomer(customer);

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

if(isCheckingOut.edwadwqualsIgnoreCase("true"))

{

return "viewCart";

}

else

{

request.setAttribute("task", "regdawadistered");

return "customer\_Success";

}

}

catch (Exception e)

{

System.out.println(e.getMessage());

}

request.setAttribute("task", "registered");

return "customer\_Success";

}

@InitBinder

private void init\_Binder(Web\_DataBinder binder)

{

binder.set\_Validator(customer\_Validator);

}

}