

IMPLEMENT SPRING SECURITY WITH AUTHENTICATION SOURCE CODE

Prepared By: Deepa Tamilselvan

CommonController.java

```
package com.mjava.controller;

import java.sql.Date; import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller; import
org.springframework.ui.ModelMap; import
org.springframework.web.bind.annotation.GetMapping; 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.bind.annotation.SessionAttributes; import
org.springframework.web.servlet.ModelAndView;

import com.mjava.model.OrderedShoeModel;
import com.mjava.model.ShoesDataModel;
import com.mjava.model.UserInfoModel;
import com.mjava.service.OrdersService;
import com.mjava.service.ShoesService;
import com.mjava.service.UsersService;
public String homeForm() { return
"index";
    }

    @Autowired private ShoesService
    shoeservice;
```

```
@Autowired private OrdersService
orderservice;
@Autowired private UsersService
userservice;

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

public String menShoesForm(ModelMap model) {
    List<ShoesDataModel> mensData = shoeservice.getMensShoeData();
    model.put("menshoeData", mensData);

    return "menShoesForm";
}

@RequestMapping(value = "/womenshoes", method = RequestMethod.GET)
public String womenMethod(ModelMap model) {
    List<ShoesDataModel> womensData =
shoeservice.getWomensShoeData();
    model.put("womenshoeData", womensData);

    return "womenShoesForm";
}
@RequestMapping("/kidshoes")
public String kidMethod(ModelMap model) {

    List<ShoesDataModel> kidsData = shoeservice.getKidsShoeData();
    model.put("kidshoeData", kidsData); return "kidsShoesForm";
}
@RequestMapping(value = "/signin", method = RequestMethod.GET)
public String signinMethod() {
```

```

        return "signinForm";
    }
    @RequestMapping(value = "/signup", method = RequestMethod.GET)
    public String sigUpMethod() { return "signUpForm";
    }

    @RequestMapping("/selectedShoe") public String
    selectedShoeInfotMethod(ModelMap model,
    @RequestParam int id) {

        try {
            ShoesDataModel selectedShoesData =
shoeservice.getshoesDataByIdService(id);

            model.put("selectedShoesData", selectedShoesData); return
            "selectedShoeInfoForm";

        } catch (Exception e) {
            System.out.println( "Exception
                at
com.mjava.controllers.CommonController.selectedShoeInfotMethod(HttpSer
vletRequest, HttpServletResponse) "
                + e.getMessage());
            return "selectedShoeInfoForm"; }
    }

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

```

```

    public String orderInforMethod(ModelMap model, @RequestParam int
id, @RequestParam String name,
        @RequestParam int categeory, @RequestParam double price,
@RequestParam String imglink,
        @RequestParam int quantity) { try
    {

        OrderedShoeModel orderedshoemodel = new
OrderedShoeModel();

        double totalPrice = price * quantity;

        orderedshoemodel.setShoeid(id);
        orderedshoemodel.setName(name);
        orderedshoemodel.setCategeory(categeory);
        orderedshoemodel.setPrice(price);
        orderedshoemodel.setImagelink(imglink);
        orderedshoemodel.setQuantity(quantity);
        orderedshoemodel.setTotalprice(totalPrice);

        model.put("orderedshoemodel", orderedshoemodel);
        return "bookShoeForm"; } catch (Exception e) {
            System.out.println( "Exception
                at
com.mjava.controllers.CommonController.orderInforMethod(HttpServletRequestReq
uest, HttpServletResponse) "
                    + e.getMessage());
            return "bookShoeForm";
        }
    }
}

```

```

    @GetMapping("/bookandpaymentreturntoHome") public String
    bookAndPaymentreturntoHomeMethod(ModelMap model,
    @RequestParam(name = "id") int shoeid,
        @RequestParam String name, @RequestParam int categeory,
    @RequestParam double price,
        @RequestParam String imglink, @RequestParam int quantity,
    @RequestParam double totalprice) {
        try { long millis =
            System.currentTimeMillis(); Date date =
            new java.sql.Date(millis);

            orderservice.insertBookingInfoService(shoeid, name,
categeory, price, imglink, quantity, totalprice, date);
            return "orderplaceForm";
        } catch (Exception e) {
            System.out.println( "Exception
                at
com.mjava.controllers.CommonController.bookAndPaymentreturntoHomeMetho
d(HttpServletRequest, HttpServletResponse) " + e.getMessage());
            return "orderplaceForm";
        }
    }

    @GetMapping("/adminsignin")
    public String adminSignInMethod(ModelMap model, @RequestParam
String username, @RequestParam String password) {

        int rollid = 0; try
        {

```

```
        UserInfoModel userInfoModel =
userService.isValidAdminUserService(username, password);

        if (userInfoModel == null) { model.put("errorMessage",
            "Invalid Credentials"); return "signinForm";
        } rollid =
        userInfoModel.getRollid();

        if (rollid == 1) {
            List<ShoesDataModel> mensData =
shoeservice.getMensShoeData();

            model.put("men_women_kids_ShoeDataName", mensData);
            model.put("username", username.toUpperCase());
            model.put("password", password); return
            "adminForm";

        } else if (rollid == 2) {
            model.put("username", username);
            model.put("password", password); return
            "index";
        } else if (rollid == 0) { model.put("errorMessage",
            "Invalid Credentials"); return "signinForm";

        }

        model.put("errorMessage", "Invalid Credentials");
return "signinForm"; } catch (Exception e) {
System.out.println(
```

```

        "Exception at
com.mjava.controllers.CommonController.signinMethod(HttpServletRequest
, HttpServletResponse) "
        + e.getMessage());
        model.put("errorMessage", "Exception check Log File");
        return "signinForm";
    }
}

@GetMapping("/getcompletemensshoesforAdmin") public String
getcompletemensshoesforAdmin(ModelMap model) { try {
    List<ShoesDataModel> mensData =
shoeservice.getMensShoeData();
    model.put("men_women_kids_ShoeDataname", mensData);
    return "adminForm";
} catch (Exception e) {
    System.out.println("Exception at
com.mjava.controllers.CommonController.getcompletemensshoesforAdmin() "
        + e.getMessage());
    return "adminForm";
}
}

@GetMapping("/getcompletewomensshoesforAdmin")
public String getcompletewomensshoesforAdmin(ModelMap model) { try
{
    List<ShoesDataModel> womensData =
shoeservice.getWomensShoeData();
    model.put("men_women_kids_ShoeDataname", womensData);

    return "adminForm"; }
catch (Exception e) {

```



```

        System.out.println("Exception at
com.mjava.controllers.CommonController.getcompletewomenshoesforAdmin()
"
        + e.getMessage());
        return "adminForm";
    }
}

@GetMapping("/getcompletekidshoesforAdmin") public String
getcompletekidshoesforAdmin(ModelMap model) { try {
    List<ShoesDataModel> kidsData =
shoeservice.getKidsShoeData();
    model.put("men_women_kids_ShoeDataname", kidsData); return
    "adminForm";
} catch (Exception e) {
    System.out.println("Exception at
com.mjava.controllers.CommonController.getcompletekidshoesforAdmin() "
    + e.getMessage());
    return "adminForm";
}
}

@RequestMapping(value = "/changeadminpassword", method =
RequestMethod.GET) public String
changeAdminPasswordForm() { return
    "changeAdminPasswordForm"; }

@RequestMapping(value = "/addNewProductForm", method =
RequestMethod.GET) public String
addNewProductForm() { return
    "addNewProductForm";
}
}

```

```

@GetMapping("/customersList")
public String customersListMethod(ModelMap model) {
    List<UserInfoModel> usersList = null; try {
        usersList = userservice.getUserDataService(); if
        (!usersList.isEmpty()) { model.put("usersList",
        usersList);
            } else {
                System.out.println("No Users Found...");
            } return
            "customersListForm";
        } catch (Exception e) {
            System.out.println( "Exception
                at
com.mjava.controllers.CommonController.customersListMethod(HttpServletRequest
Request, HttpServletResponse) "
                    + e.getMessage());
            return "customersListForm"; }
    }

@GetMapping("/reports")
public String reportsMethod(ModelMap model) {
    List<OrderedShoeModel> orderedShoeList = null;
    try { orderedShoeList =
orderservice.getCompleteTransactionsDataService();
        if (orderedShoeList != null) { double totalSales = 0;
            for (OrderedShoeModel osl : orderedShoeList) {
                totalSales = totalSales + osl.getTotalprice();
            }
        }
    }
}

```

```

        model.put("orderedShoeList", orderedShoeList);
        model.put("totalSales", totalSales);
    } return
    "reportsForm";
} catch (Exception e) {
    System.out.println("Exception at
com.mjava.controllers.CommonController.reportsMethod() " +
e.getMessage());
    return "reportsForm";
}
}
@GetMapping("/changeadminPassword") public String
changeadminPasswordMethod(ModelMap model, @RequestParam
String currentpassword,
    @RequestParam String newpassword, @RequestParam String
confirmpassword) {
    UserInfoModel userInfoModel = null;
    boolean isCoreectPassword = false;
    try { userInfoModel =
userservice.isCorrectPassword_or_NotService(currentpassword);
        if (userInfoModel.getRollid() != 0) { boolean
            isUpdatedAdminPassword =
userservice.updateAdminPasswordService(userInfoModel, currentpassword,
                newpassword, confirmpassword);
            if (isUpdatedAdminPassword == true) {
                System.out.println("Password Updated"); return
                "signinForm";
            } else {
                System.out.println("Password Not Updated");
            }
        }
    }
}

```

```

        } else
        {
            System.out.println("No Data Found");
        } return
        "changeAdminPasswordForm";
    } catch (Exception e) {
        System.out.println( "Exception
                           at
com.mjava.controllers.CommonController.changeadminPasswordMethod(HttpS
ervletRequest, HttpServletResponse) "
                           + e.getMessage());
        return "changeAdminPasswordForm"; }
    }

    @GetMapping("/requiredreports")
    public String requiredReportsMethod(ModelMap model, @RequestParam
String category, @RequestParam String date) {
        List<OrderedShoeModel> orderedShoeList = null;
        try { int categoryId = 0;
            String categoryName = category;
            if (categoryName.equals("mens"))
                categoryId = 1;
            else if (categoryName.equals("womens")) categoryId
                = 2;
            else if (categoryName.equals("kids")) categoryId
                = 3;

            String jspdate = date; Date sqldate =
            Date.valueOf(jspdate); if
            (categoryId != 0) {

```

```

        orderedShoeList =
orderservice.getRequiredCompleteTransactionsDataService(categoryId,
sqldate);

        if (orderedShoeList != null) { double totalSales = 0;
            for (OrderedShoeModel osl : orderedShoeList) {
                totalSales = totalSales + osl.getTotalprice();
            } model.put("orderedShoeList",
                orderedShoeList); model.put("totalSales",
                totalSales); return "reportsForm";
        } } return
"reportsForm"; } catch
(Exception e) {
    System.out.println( "Exception
        at
com.mjava.controllers.CommonController.requiredReportsMethod() " +
e.getMessage());
    return "reportsForm";
}
}
@RequestMapping("/addnewproduct")
public String addNewProductMethod(ModelMap model, @RequestParam
String weartype, @RequestParam String prod_name,
    @RequestParam double prod_price, @RequestParam String
prod_img) {
    ModelAndView mv = null; try { int
    category = 0; if
    (weartype.equals("Mens_Wear")) {

```

```
        categoroy = 1;
    } else if (weartype.equals("Womens_Wear")) { categoroy
        = 2;
    } else if (weartype.equals("Kids_Wear")) { categoroy
        = 3;
    } boolean isInsertedNewProduct =
        false; isInsertedNewProduct =
shoeservice.insertNewProductService(categoroy, prod_price, prod_name,
prod_img);
        if (isInsertedNewProduct == true) { if
            (categoroy == 1) {
                List<ShoesDataModel> mensData =
shoeservice.getMensShoeData();
                model.put("men_women_kids_ShoeDataname",
mensData); return
"adminForm";
            }
            if (categoroy == 2) {
                List<ShoesDataModel> womensData =
shoeservice.getWomensShoeData();

                model.put("men_women_kids_ShoeDataname",
womensData); return
"adminForm";
            }
            if (categoroy == 3) {
                List<ShoesDataModel> kidsData =
shoeservice.getKidsShoeData();
```

```

        model.put("men_women_kids_ShoeDataname",
kidsData); return
"adminForm";

    }

    } else {
        System.out.println("Product Not Added");
    } return
"adminForm";

    } catch (Exception e) {
        System.out.println( "Exception
            at
com.mjava.controllers.CommonController.addNewProducttMethod(HttpServle
tRequest, HttpServletResponse) "
            + e.getMessage());
        return "adminForm";
    }
}

@GetMapping("/signupnewuser")
public String sigUpNewUserMethod(ModelMap model, @RequestParam
String firstname, @RequestParam String lastname,
    @RequestParam String password, @RequestParam String
confirmpassword, @RequestParam String roll,
    @RequestParam Long mobileno, @RequestParam String email) {
    ModelAndView mv = null; boolean isInserted = false; try { int
rollid = 0; if (roll.equals("admin")) { rollid = 1;

```

```

        } else { rollid
            = 2;
        }

        isInserted = true;
        userservice.insertUserDataService(firstname, lastname,
password, confirmpassword, rollid, mobileno, email);
        if (isInserted == true) {

            return "signinForm";
        }

        if (isInserted == false) {

            return "testErrorForm";
        } return
        "testErrorForm";

    } catch (Exception e) {
        System.out.println( "Exception
            at
com.mjava.controllers.CommonController.sigUpNewUserMethod(HttpServletRequestR
equest, HttpServletResponse) "
            + e.getMessage());
        return "testErrorForm";
    }
}

@GetMapping("/deleteProduct") public String
deleteProductMethod(ModelMap model, @RequestParam
int id) {
    boolean isdeleted = false;

```



```

        List<ShoesDataModel> shoesDatalist = null;
        ShoesDataModel findwhichCategeory = null;
        try { findwhichCategeory =
shoeservice.getshoesDataByIdService(id);
            int category = findwhichCategeory.getCategory();
            isdeleted = shoeservice.deleteProductwithIdService(id);
            if (isdeleted == true) { shoesDatalist =
shoeservice.getShoeDatabyCategory(category);
                model.put("men_women_kids_ShoeDataname",
shoesDatalist); } return
                "adminForm";
        } catch (Exception e) {
            System.out.println( "Exception
                at
com.mjava.controllers.CommonController.deleteProductMethod() " +
e.getMessage());
            return "adminForm";
        }
    }
    @GetMapping("/updateProduct")
    public String updateProductMethod(ModelMap model, @RequestParam
int id) {
        boolean isupdated = false;
        ShoesDataModel shoesData = null;
        // ShoesDataModel findwhichCategeory = null; try {
        shoesData = shoeservice.getshoesDataByIdService(id); if
        (shoesData.getId() > 0) {

```

```

        model.put("men_women_kids_ShoeDataName", shoesData);
        return "updateProductForm";
    } return
    "adminForm";
} catch (Exception e) {
    System.out.println( "Exception
                        at
com.mjava.controllers.CommonController.deleteProductMethod() " +
e.getMessage());
    return "adminForm";
}
}
@GetMapping("/updateToNewproduct")
public String updateProductMethod(ModelMap model, @RequestParam
String weartype, @RequestParam String prod_name,
    @RequestParam double prod_price, @RequestParam String
prod_img, @RequestParam int prod_id) {

    boolean isupdated = false;
    ShoesDataModel shoesData = null;
    List<ShoesDataModel> shoesDataList = null;
    try { int category = 0; if
    (weartype.equals("Mens_Wear")) { category
    = 1;
        } else if (weartype.equals("Womens_Wear")) { category
        = 2;
        } else if (weartype.equals("Kids_Wear")) { category
        = 3;
        }
    }

```

```

        isupdated =
shoeservice.updateShoeProductService(categeory, prod_price, prod_name,
prod_img, prod_id);
        if (isupdated == true) { shoesDataList
            =
shoeservice.getShoeDatabyCategeory(categeory);
model.put("men_women_kids_ShoeDataname",
shoesDataList); } return
    "adminForm";

    } catch (Exception e) {
        System.out.println( "Exception
            at
com.mjava.controllers.CommonController.deleteProductMethod() " +
e.getMessage());
        return "adminForm";
    }
}
}

```

Ordersdao.java

```

package com.mjava.dao; import java.sql.Date; import
java.util.List; import
org.springframework.data.jpa.repository.Query; import
org.springframework.data.repository.CrudRepository; import
org.springframework.stereotype.Repository;

```

```
import com.mjava.model.OrderedShoeModel;

@Repository
public interface OrdersDao extends CrudRepository<OrderedShoeModel,
Integer>{
    @Query("from OrderedShoeModel where categeory=:categeoryId and
date=:date") public List<OrderedShoeModel>
getRequiredCompleteTransactionsData(int categeoryId,Date date);
}
```

Shoesdao.java

```
package com.mjava.dao;

import org.springframework.data.jpa.repository.Modifying;
import org.springframework.data.jpa.repository.Query; import
org.springframework.data.repository.CrudRepository; import
org.springframework.data.repository.query.Param; import
org.springframework.stereotype.Repository;
import org.springframework.transaction.annotation.Transactional;
import com.mjava.model.ShoesDataModel; import
antlr.collections.List;

@Repository
public interface ShoesDao extends CrudRepository<ShoesDataModel,
Integer> {
    @Query("from ShoesDataModel where categeory=:categeory") public
        Iterable<ShoesDataModel> findByCategory(int
categeory);
```

```

        @Query("from ShoesDataModel where id=:seletedShoeId") public
        ShoesDataModel getshoesDataById(int
seletedShoeId);
        @Modifying
        @Transactional
        @Query("delete from ShoesDataModel where id=:id") public
        int deleteProductwithId(@Param("id") int id); @Modifying
        @Transactional
        @Query("update ShoesDataModel set categeory=:categeory,
price=:price, name=:name, imagelink=:imagelink where id=:id")
        public int updateShoeProduct(@Param("categeory") int
categeory, @Param("price") double price,
        @Param("name") String name, @Param("imagelink")
String imagelink,
        @Param("id") int id);
    }

```

UsersDao.java

```

package com.mjava.dao; import
org.springframework.data.jpa.repository.Modifying; import
org.springframework.data.jpa.repository.Query; import
org.springframework.data.repository.CrudRepository; import
org.springframework.data.repository.query.Param; import
org.springframework.stereotype.Repository; import
org.springframework.transaction.annotation.Transactional;
import com.mjava.model.UserInfoModel;
@Repository

```

```

public interface UsersDao extends
CrudRepository<UserInfoModel, Integer> {
    @Query("from UserInfoModel where firstname=:username
and password=:password") public UserInfoModel
isValidAdminUser(String
username,String password );
    @Query("from UserInfoModel where rollid=:rollid")
    public Iterable<UserInfoModel>
getUserDatabyRollid(int rollid);
    @Query("from UserInfoModel where password=:password")
    public UserInfoModel
isCorrectPassword_or_Not(String password);

    @Modifying
    @Transactional
    @Query("update UserInfoModel set password =
:password where rollid=1 and firstname=:username") public
    int updateAdminPassword(@Param("password")
String password, @Param("username") String username);
}

```

GlobalExceptionHandler.java

```
package com.mjava.exception;
import
org.springframework.web.bind.annotation.ControllerAdvice;

import
org.springframework.web.bind.annotation.ExceptionHandler;
@ControllerAdvice public class
GlobalExceptionHandler {
@ExceptionHandler(value=Exception.class)
public String handleAnyException()
    { return "Error";
    }
}
```

OrderedShoeModel.java


```
package com.mjava.model; import
java.sql.Date; import
javax.persistence.Entity; import
javax.persistence.GeneratedValue; import
javax.persistence.Id; import
javax.persistence.Table;
@Entity
@Table(name = "tbl_orderinfo") public
class OrderedShoeModel {
    @Id
    @GeneratedValue private
    int orderid; private
    int shoeid ; private
    int categeory; private
    double price; private
    int quantity ; private
    Date date;
```

```
private String name; private
String imagelink; private
double totalprice ; public
int getQuantity() { return
quantity;
} public void setQuantity(int quantity)
{ this.quantity = quantity;
} public int getOrderid()
{ return orderid;
} public void setOrderid(int orderid)
{ this.orderid = orderid;
} public int getShoeid()
{ return shoeid;
} public void setShoeid(int shoeid)
{ this.shoeid = shoeid;
} public int getCategeory()
{ return categeory;
} public void setCategeory(int categeory)
{ this.categeory = categeory;
} public double getPrice()
{ return price;
}
```

```
public void setPrice(double price) { this.price =  
    price;  
} public String getName() {  
return name;  
} public void setName(String name) {  
this.name = name;  
} public String getImagelink() {  
return imagelink;  
} public void setImagelink(String imagelink) {  
this.imagelink = imagelink;  
} public double getTotalprice() {  
return totalprice;  
} public void setTotalprice(double totalprice) {  
this.totalprice = totalprice;  
}  
  
public Date getDate() { return  
    date;  
} public void setDate(Date date) {  
this.date = date;  
}  
  
@Override public String  
toString() {
```

```

        return "OrderedShoeModel [orderid=" + orderid + ", shoeid=" +
shoeid + ", categeory=" + categeory + ", price="
        + price + ", quantity=" + quantity + ", date=" + date
+ ", name=" + name + ", imagelink=" + imagelink
        + ", totalprice=" + totalprice + "]);
    } public OrderedShoeModel()
    { super();
        // TODO Auto-generated constructor stub
    } public OrderedShoeModel(int shoeid, int categeory, double
price, int quantity, Date date, String name,
        String imagelink, double totalprice) {
    super(); this.shoeid = shoeid; this.categeory
= categeory; this.price = price; this.quantity
= quantity; this.date = date; this.name =
name; this.imagelink = imagelink;
    this.totalprice = totalprice; }
}

```

ShoesDataModel.java

```

package com.mjava.model; import
javax.persistence.Entity; import
javax.persistence.GeneratedValue; import
javax.persistence.Id;

```

```
import javax.persistence.Table;
@Entity
@Table(name = "tbl_shoesinfo") public
class ShoesDataModel {
    @Id
    @GeneratedValue private
    int id; private int
    categeory; private double
    price; private String
    name; private String
    imagelink; public int
    getId() { return id;
    } public void setId(int id)
    { this.id = id;
    } public int getCategeory()
    { return categeory;
    } public void setCategeory(int categeory)
    { this.categeory = categeory;
    } public double getPrice()
    { return price;
    } public void setPrice(double price)
    { this.price = price;
    } public String getName()
    { return name;
```

```

    } public void setName(String name)
    { this.name = name;
    } public String getImagelink()
    { return imagelink;
    } public void setImagelink(String imagelink)
    { this.imagelink = imagelink;
    }

    @Override
    public String toString() { return "ShoesDataModel [categeory=" +
        categeory + ", price=" +
price + ", name=" + name + ", imagelink="
        + imagelink + "]";
    }
    public ShoesDataModel(int categeory, double price, String name,
String imagelink) {
        super(); this.categeory =
        categeory; this.price =
        price; this.name = name;
        this.imagelink = imagelink;
    } public ShoesDataModel()
    { super();
        // TODO Auto-generated constructor stub }
}

```

UserInfoModel.java

```
package com.mjava.model; import
javax.persistence.Entity; import
javax.persistence.GeneratedValue; import
javax.persistence.Id; import
javax.persistence.Table;
@Entity
@Table(name="tbl_userinfo") public
class UserInfoModel {
    @Id
    @GeneratedValue private
    int id; private String
    firstname; private String
    lastname; private String
    password; private int
    rollid; private long
    phno; private String
    email; public int getId()
    { return id;
    } public void setId(int id)
    { this.id = id;
    } public String getFirstname()
    { return firstname;
    } public void setFirstname(String firstname)
    { this.firstname = firstname;
    } public String getLastname()
    {
```

```
        return lastname;
    } public void setLastname(String lastname) {
    this.lastname = lastname;
    } public String getPassword() {
    return password;
    } public void setPassword(String password) {
    this.password = password;
    } public int getRollid() {
    return rollid;
    } public void setRollid(int rollid) {
    this.rollid = rollid;
    } public long getPhno() {
    return phno;
    } public void setPhno(long phno) {
    this.phno = phno;
    } public String getEmail() {
    return email;
    } public void setEmail(String email) {
    this.email = email;
    }
    @Override public String
    toString() {
```



```

        return "UserInoModel [firstname=" + firstname + ", lastname="
+ lastname + ", password=" + password
        + ", rollid=" + rollid + ", phno=" + phno + ", email="
+ email + "]";
    } public UserInoModel()
    { super();
        // TODO Auto-generated constructor stub
    } public UserInoModel(String firstname, String lastname,
String
password, int rollid, long phno, String email) {
    super(); this.firstname =
    firstname; this.lastname =
    lastname; this.password =
    password; this.rollid =
    rollid; this.phno = phno;
    this.email = email;
    }
}

```

ShoesService.java

```

package com.mjava.service;
import java.sql.Date; import
java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.jpa.repository.Query; import
org.springframework.data.repository.query.Param; import
org.springframework.stereotype.Service;

```

```

import com.mjava.dao.OrdersDao; import
com.mjava.dao.ShoesDao; import
com.mjava.dao.UsersDao; import
com.mjava.model.OrderedShoeModel; import
com.mjava.model.ShoesDataModel; import
com.mjava.model.UserInfoModel;
@Service
public class ShoesService {
    @Autowired
    private ShoesDao shoesDao;
    @Autowired private
    OrdersDao orderDao;
    @Autowired private UsersDao userDao; public
    List<ShoesDataModel> getMensShoeData() {
        List<ShoesDataModel> mensData=
(List)shoesDao.findByCategory(1);
        System.out.println(mensData); return
        mensData;
    }
    public List<ShoesDataModel> getWomensShoeData() {
List<ShoesDataModel> womensData= (List)shoesDao.findByCategory(2);
        System.out.println(womensData); return
        womensData;
    }
    public List<ShoesDataModel> getKidsShoeData() {
        List<ShoesDataModel> kidsData= (List)shoesDao.findByCategory(3)
        return kidsData;
    } public List<ShoesDataModel> getShoeDatabyCategeory(int
categoryid) {

```

```

        List<ShoesDataModel> shoesDatabyCategeory=
(List)shoesDao.findByCategory(categeoryid);
        return shoesDatabyCategeory;
    }    public    ShoesDataModel    getshoesDataByIdService(int
seletedShoeId)
{
    ShoesDataModel selectedShoesData=
shoesDao.getshoesDataById(seletedShoeId);
    System.out.println(selectedShoesData); return
selectedShoesData;
    } public boolean insertNewProductService(int
categeory,double
price,String name,String imagelink)
{
    boolean isInsertedNewProduct = false; try
    { ShoesDataModel s=new
ShoesDataModel(categeory,price, name, imagelink);
    ShoesDataModel count= shoesDao.save(s);
    isInsertedNewProduct = true; return
isInsertedNewProduct;
    } catch (Exception e) {
        System.out.println("Exception at
insertNewProductService(int categeory,double price,String name,String
imagelink) " + e.getMessage());
        return false;
    }
} public boolean deleteProductwithIdService(int
id) {
    int count=0;
    boolean isdeleted=false; count=
shoesDao.deleteProductwithId(id);
    if(count>0)

```

```

        { isdeleted=true;
          } return
        isdeleted;
    }
    public boolean updateShoeProductService( int category, double
price,
        String name, String imagelink, int
        id) {
        int count=0; boolean isUpdated=false; count=
        shoesDao.updateShoeProduct(category, price, name,
imagelink, id);
        if(count>0)
        { isUpdated=true;
          } return
        isUpdated;
    }
}

```

SportyShoesOnlineApplication.java

```

package com.mjava.SportyShoesOnline; import
org.springframework.boot.SpringApplication; import
org.springframework.boot.autoconfigure.SpringBootApplication; import
org.springframework.boot.autoconfigure.domain.EntityScan; import
org.springframework.context.annotation.ComponentScan; import
org.springframework.data.jpa.repository.config.EnableJpaRepositories;

```

```

import
org.springframework.web.servlet.config.annotation.ResourceHandlerRegis
try; import
org.springframework.web.servlet.config.annotation.WebMvcConfigurer;
@SpringBootApplication
@ComponentScan("com.mjava")
@EntityScan("com.mjava.model")
@EnableJpaRepositories("com.mjava.dao") public
class SportyShoesOnlineApplication { public
static void main(String[] args) {
    SpringApplication.run(SportyShoesOnlineApplication.class,
args); }
}

```

Application.properties

```

logging.level.org.springframework.web: DEBUG
spring.mvc.view.prefix=/WEB-INF/view/
spring.mvc.view.suffix=.jsp server.port=8090
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://localhost:3306/sportyshoes_db
spring.datasource.username=root spring.datasource.password=root

```

=====X=====