```
//Spring Boot Application Coding

package com.shoes;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.autoconfigure.security.servlet.SecurityAutoConfiguration;

@SpringBootApplication(exclude = {SecurityAutoConfiguration.class })
public class AwsomeSportyshoesApplication {

    public static void main(String[] args) {
        SpringApplication.run(AwsomeSportyshoesApplication.class, args);
    }

}


//Controller-Authentication.java

package com.shoes.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;

import com.shoes.dao.DAO;
import com.shoes.model.User;

@Controller
public class Authentication {
    @Autowired
    DAO dao;

    @GetMapping("/")
    public String welcome(Model model) {
        model.addAttribute("user", new User());
        return "index";
    }

    @PostMapping("/login")
    public String login(@ModelAttribute("user") User user, Model model) {
        String us = user.getName();
        String pw = user.getPassword();
        String resultpw = dao.authenticate(us, pw);
        model.addAttribute("user", user);
        if (pw.equals(resultpw)) {
            return "welcome";
        } else {
            model.addAttribute("error", "Password invalid");
            return "index";
        }
    }

    @PostMapping("/changePW")
    public String changePW(@ModelAttribute("user") User user, Model model) {
        String us = user.getName();
        String pw = user.getPassword();
        int results = dao.changePw(us, pw);
        System.out.println("Results: " + results);
        if (results == 1) {
            model.addAttribute("message", "Password updated");
        } else {
            model.addAttribute("message", "Password not updated");
        }
        model.addAttribute("user", user);
        return "welcome";
    }
}


//Controller-Product.java

package com.shoes.controller;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
```

```java
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;

import com.shoes.dao.DAO;
import com.shoes.model.Product;

@Controller
public class Products {
    @Autowired
    DAO dao;

    @GetMapping("/manageProducts")
    public String listUser(Model model) {
        List<Product> products = dao.getProducts();
        model.addAttribute("products", products);
        model.addAttribute("product", new Product());
        return "Products";
    }

    @PostMapping("/addProduct")
    public String addProduct(@ModelAttribute("product") Product product, Model model) {
        String name = product.getName();
        String category = product.getCategory();
        int affected = dao.setProduct(name, category);
        if (affected == 1) {
            model.addAttribute("message", "Product added");
        } else {
            model.addAttribute("message", "Product not added");
        }
        return "Products";
    }

    @PostMapping("/delProduct")
    public String delProduct(@ModelAttribute("product") Product product, Model model) {
        String name = product.getName();
        String category = product.getCategory();
        int affected = dao.delProduct(name, category);
        if (affected == 1) {
            model.addAttribute("message2", "Product deleted");
        } else {
            model.addAttribute("message2", "Product not deleted");
        }
        return "Products";
    }
}

//Controller- SearchPurchase.java

package com.shoes.controller;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;

import com.shoes.dao.DAO;
import com.shoes.model.Product;

@Controller
public class Products {
    @Autowired
    DAO dao;

    @GetMapping("/manageProducts")
    public String listUser(Model model) {
        List<Product> products = dao.getProducts();
        model.addAttribute("products", products);
        model.addAttribute("product", new Product());
        return "Products";
    }

    @PostMapping("/addProduct")
    public String addProduct(@ModelAttribute("product") Product product, Model model) {
        String name = product.getName();
        String category = product.getCategory();
        int affected = dao.setProduct(name, category);
        if (affected == 1) {
            model.addAttribute("message", "Product added");
```

```
        } else {
            model.addAttribute("message", "Product not added");
        }
        return "Products";
    }

    @PostMapping("/delProduct")
    public String delProduct(@ModelAttribute("product") Product product, Model model) {
        String name = product.getName();
        String category = product.getCategory();
        int affected = dao.delProduct(name, category);
        if (affected == 1) {
            model.addAttribute("message2", "Product deleted");
        } else {
            model.addAttribute("message2", "Product not deleted");
        }
        return "Products";
    }
}

//Controller- SearchUser.java

package com.shoes.controller;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;

import com.shoes.dao.DAO;
import com.shoes.model.User;

@Controller
public class SearchUsers {
    @Autowired
    DAO dao;
    @PostMapping("/listUsers")
    public String listUser(Model model) {
        List<User> users = dao.getUsers();
        model.addAttribute("users",users);
        return "listUser";
        }
    @PostMapping("/searchUser")
    public String searchUser(@ModelAttribute("user") User user, Model model) {
        String userName = user.getSearchName();
        //System.out.println("Username: " + userName);
        String u = dao.searchUser(userName);
        if(u.equals(userName)) {
            model.addAttribute("userFound", "User " + user.getSearchName()+" found");
            }
        else
        {
                model.addAttribute("userFound", "User " + user.getSearchName()+" not found");
        }
        //model.addAttribute("user",user);
        //System.out.println("Searchuser: " + user.getName());
        //System.out.println("userName: " + user.getSearchName());

        return "searchUser";
        }
}

//DAO- DAO.java

package com.shoes.dao;

import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.PreparedStatementSetter;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.jdbc.core.namedparam.MapSqlParameterSource;
import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;
import org.springframework.jdbc.core.namedparam.SqlParameterSource;
import org.springframework.stereotype.Repository;

import com.shoes.model.Product;
import com.shoes.model.Purchase;
import com.shoes.model.User;

import java.sql.PreparedStatement;
import java.sql.ResultSet;
```

```java
import java.sql.SQLException;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;

@Repository
public class DAO {
    @Autowired
    NamedParameterJdbcTemplate NPjdbcTemplate;
    @Autowired
    JdbcTemplate jdbcTemplate;

    public String authenticate(String user, String password) {
        SqlParameterSource namedParameters = new MapSqlParameterSource().addValue("user", user);
        return NPjdbcTemplate.queryForObject("select password from user where name =:user", namedParameters,
                String.class);
    }

    public int changePw(String user, String password) {
        int result = jdbcTemplate.update("update user set password = ? where name = ?", password, user);
        return result;
    }

    public List<User> getUsers() {
        return jdbcTemplate.query("select * from user", new RowMapper<User>() {
            @Override
            public User mapRow(ResultSet rs, int rowNum) throws SQLException {
                User u = new User();
                u.setId(rs.getInt(1));
                u.setName(rs.getString(2));
                return u;
            }
        });
    }

    public String searchUser(String user) {
        SqlParameterSource namedParameters = new MapSqlParameterSource().addValue("user", user);
        String us = null;
        try {
            us = NPjdbcTemplate.queryForObject("select name from user where name=:user", namedParameters, String.class);
        } catch (Exception ex) {
            System.out.println("Empty Resultset");
        }
        if (us != null && !us.isEmpty() && us.equals(user)) {

            return us;

        } else {
            return "not found";
        }
    }

    public List<Product> getProducts() {
        return jdbcTemplate.query("select * from product", new RowMapper<Product>() {
            @Override
            public Product mapRow(ResultSet rs, int rowNum) throws SQLException {
                Product p = new Product();
                p.setId(rs.getInt(1));
                p.setName(rs.getString(2));
                p.setCategory(rs.getString(3));
                return p;
            }
        });
    }

    public int setProduct(String name, String category) {
        int result = jdbcTemplate.update("insert into product (name, category) values(?, ?)", name, category);
        return result;
    }

    public int delProduct(String name, String category) {
        int result = jdbcTemplate.update("delete from product where name =? and category=?", name, category);
        return result;
    }

    public List<Purchase> searchAllPurchases() {
        return jdbcTemplate.query(
                "select u.name, p.name, p.category, pu.timestamp from purchase pu left join (user u, product p) on (pu.id
                new RowMapper<Purchase>() {
                    @Override
                    public Purchase mapRow(ResultSet rs, int rowNum) throws SQLException {
                        Purchase pu = new Purchase();
                        pu.setUser(rs.getString(1));
```

```
                pu.setProductName(rs.getString(2));
                pu.setProductCategory(rs.getString(3));
                pu.setTimestamp(rs.getDate(4));
                return pu;
            }
        });
    }

    public List<Purchase> searchPurchasesByDate(String date) {
        return jdbcTemplate.query(
                "select u.name, p.name, p.category, pu.timestamp from purchase pu left join (user u, product p) on (pu.id
                new PreparedStatementSetter() {
                    public void setValues(PreparedStatement preparedStatement) throws SQLException {
                        preparedStatement.setString(1, date);
                    }
                }, new RowMapper<Purchase>() {
                    @Override
                    public Purchase mapRow(ResultSet rs, int rowNum) throws SQLException {
                        Purchase pu = new Purchase();
                        pu.setUser(rs.getString(1));
                        pu.setProductName(rs.getString(2));
                        pu.setProductCategory(rs.getString(3));
                        pu.setTimestamp(rs.getDate(4));
                        return pu;
                    }
                });
    }

    public List<Purchase> searchPurchasesByCategory(String category) {
        return jdbcTemplate.query(
                "select u.name, p.name, p.category, pu.timestamp from purchase pu left join (user u, product p) on (pu.id
                new PreparedStatementSetter() {
                    public void setValues(PreparedStatement preparedStatement) throws SQLException {
                        preparedStatement.setString(1, category);
                    }
                }, new RowMapper<Purchase>() {
                    @Override
                    public Purchase mapRow(ResultSet rs, int rowNum) throws SQLException {
                        Purchase pu = new Purchase();
                        pu.setUser(rs.getString(1));
                        pu.setProductName(rs.getString(2));
                        pu.setProductCategory(rs.getString(3));
                        pu.setTimestamp(rs.getDate(4));
                        return pu;
                    }
                });
    }
}


//Model -Product.java

package com.shoes.model;

import lombok.Data;

@Data
public class Product
{
    private int id;
    private String name;
    private String category;

    public int getId()  {
        return id;
        }

    public void setId(int id) {
        this.id = id;
        }

    public String getName() {
        return name;
        }

    public void setName(String name) {
        this.name = name;
        }

    public String getCategory() {
        return category;
        }
```

```
    public void setCategory(String category) {
        this.category = category;
        }
}



//Model- Purchase.java

package com.shoes.model;

import java.sql.Date;

import lombok.Data;

@Data
public class Purchase {
    private String id;
    private String user;
    private String productName;
    private String productCategory;
    private Date timestamp;

    public String getId() {
        return id;
        }

    public void setId(String id) {
        this.id = id;
        }

    public String getUser() {
        return user;
        }

    public void setUser(String user) {
        this.user = user;
        }

    public String getProductName() {
        return productName;
        }

    public void setProductName(String productName) {
        this.productName = productName;
        }

    public String getProductCategory() {
        return productCategory;
        }

    public void setProductCategory(String productCategory) {
        this.productCategory = productCategory;
        }

    public Date getTimestamp() {
        return timestamp;
        }

    public void setTimestamp(Date timestamp) {
        this.timestamp = timestamp;
        }
}


//Model- User.java

package com.shoes.model;

import lombok.Data;

@Data
public class User {
    public int id;
    public String name;
    public String password;
    public String searchName;

    public int getId() {
        return id;
    }
```

```
    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getSearchName() {
        return searchName;
    }

    public void setSearchName(String searchName) {
        this.searchName = searchName;
    }
}
```