Project 3- Sporty Shoes

Project and developer details

The Project is developed by Aabitha Bathool

The all code of project is hosted on GitHub: -

https://github.com/Aabidha/Phase3Project_Sporty-Shoes

Sprints planned and the tasks achieved in them

Creating a document and file structure of the project.

Firstly, I create a Git repo and do Git Clone.

Then create a Maven Project which contains java files in src main and jsp files in views folder.

After completing all code, I have performed different test case on this project.

At last, I push all correct to GitHub.

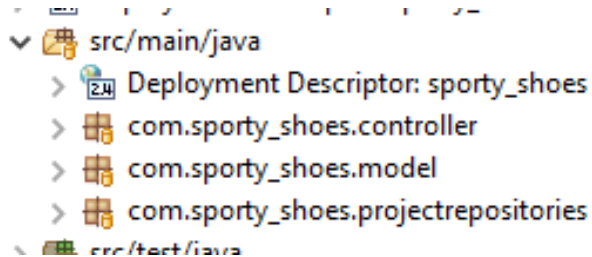## Algorithms and flowcharts of the application Core concepts used in the project

Searching, Collections Frameworks, MySQL Connector, JSP, HTML, CSS, Hibernate, ORM, Controller, Autowired, RequestMapping.
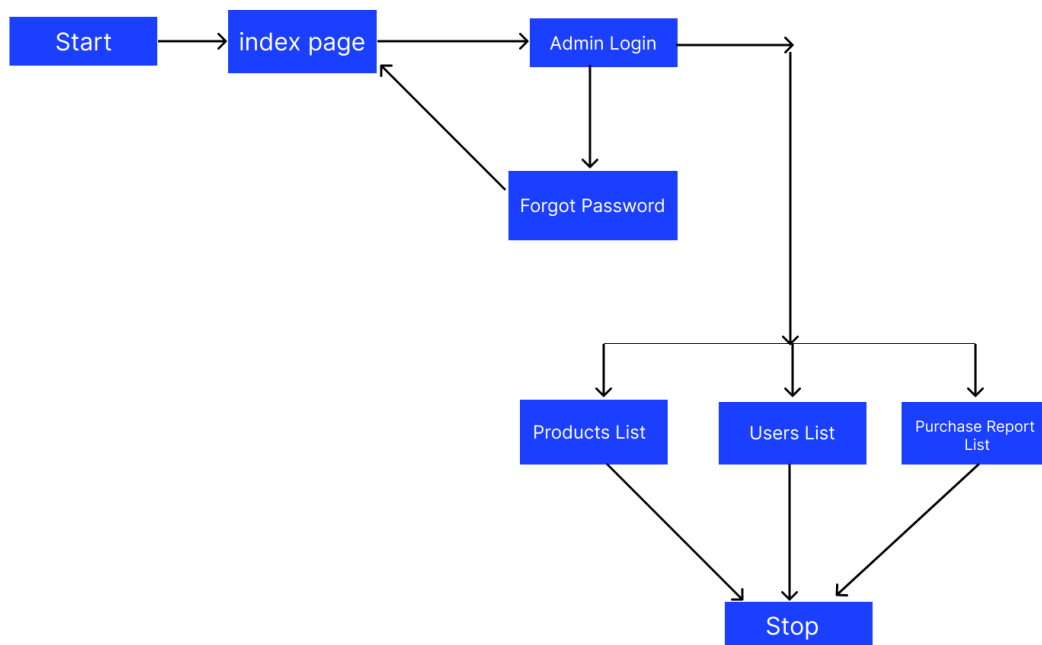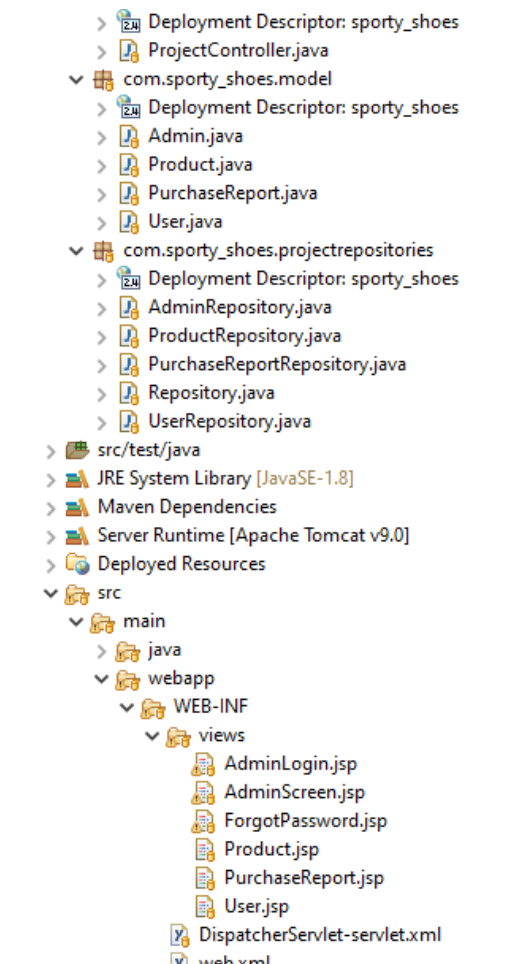
Project Details

Create a Maven Project select archetype maven-archetype-webapp to created maven web project then fill groupID, ArtifactID.

Creating packages like

- src/main/java
  - Deployment Descriptor: sporty_shoes
  - com.sporty_shoes.controller
  - com.sporty_shoes.model
  - com.sporty_shoes.projectrepositories
  - src/test/java

Create a structure

- Deployment Descriptor: sporty_shoes
- ProjectController.java
- com.sporty_shoes.model
  - Deployment Descriptor: sporty_shoes
  - Admin.java
  - Product.java
  - PurchaseReport.java
  - User.java
- com.sporty_shoes.projectrepositories
  - Deployment Descriptor: sporty_shoes
  - AdminRepository.java
  - ProductRepository.java
  - PurchaseReportRepository.java
  - Repository.java
  - UserRepository.java
- src/test/java
- JRE System Library [JavaSE-1.8]
- Maven Dependencies
- Server Runtime [Apache Tomcat v9.0]
- Deployed Resources
- src
  - main
    - java
    - webapp
      - WEB-INF
        - views
          - AdminLogin.jsp
          - AdminScreen.jsp
          - ForgotPassword.jsp
          - Product.jsp
          - PurchaseReport.jsp
          - User.jsp
        - DispatcherServlet-servlet.xml
        - web.xml

Some Important Code Snippets –

Autowired and RequesMapping in ProjectController

```java
@Autowired
AdminRepository adminRepository;
@Autowired
ProductRepository productRepository;
@Autowired
UserRepository  userRepository;
@Autowired
PurchaseReportRepository purchaseReportRepository;
@Autowired
Repository repository;

@RequestMapping("/")
public String showHome() {
    return "Home";
}

@RequestMapping(value="adminLogin",method =RequestMethod.GET)
public String adminLoginPage() {
    return "AdminLogin";
}

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

@RequestMapping(value="adminForgotPassword",method =RequestMethod.POST)
public String adminForgotPasswordPage(@RequestParam("email")String email,
        @RequestParam("password")String password,ModelMap map) {


    if(repository.forgotPassword(email,password))
```

```java
@RequestMapping(value="adminForgotPassword",method =RequestMethod.POST)
public String adminForgotPasswordPage(@RequestParam("email")String email,
        @RequestParam("password")String password,ModelMap map) {


    if(repository.forgotPassword(email,password))
        map.addAttribute("message","Password Updated");
    else
        map.addAttribute("message","Invalid Details");

    return "ForgotPassword";
}


@RequestMapping(value="adminscreen",method=RequestMethod.POST)
public String adminPage(@RequestParam(name="email")String email,
        @RequestParam(name="password")String password,ModelMap map) {

    if(adminRepository.verifyAdmin(new Admin(email,password)))
        return "AdminScreen";
    else {
        map.addAttribute("message", "Invalid Details");
        return "AdminLogin";
    }

}

@RequestMapping(value="product",method=RequestMethod.GET)
public String getAllProducts(ModelMap map) {

    map.addAttribute("productList",productRepository.getAllProducts());
    return "Product";

}

@RequestMapping(value="user",method=RequestMethod.GET)
public String getAllUsers(ModelMap map) {

    map.addAttribute("userList",userRepository.getAllUsers());

    return "User";

}

@RequestMapping(value="searchUser",method=RequestMethod.GET)
public String searchUser(@RequestParam("email")String email,ModelMap map) {

    map.addAttribute("userList",userRepository.searchUser(email));
    return "User";

}

@RequestMapping(value="purchaseReport",method=RequestMethod.GET)
public String getRport(ModelMap map) {

    map.addAttribute("report",purchaseReportRepository.getReport());
    return "PurchaseReport";

}

@RequestMapping(value="deleteProduct",method=RequestMethod.GET)
public String deleteProduct(@RequestParam("id")int id,ModelMap map) {

    if(productRepository.deleteProduct(id))
        map.addAttribute("message","Product Deleted Successfully");
    else
        map.addAttribute("message","Try after few minutes");

    map.addAttribute("productList",productRepository.getAllProducts());

    return "Product";

}
```

Project Model

Admin model

```java
@Id
private String email;
private String password;

public Admin() {
    super();
}

public Admin(String email, String password) {
    super();
    this.email = email;
    this.password = password;
}

public String getEmail() {
    return email;
}

public void setEmail(String email) {
    this.email = email;
}

public String getPassword() {
    return password;
}

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

@Override
public String toString() {
    return String.format("Admin [email=%s, password=%s]", email, password);
}
```

Product Model

```java
@Id
@GeneratedValue(strategy=GenerationType.AUTO)
private int productId;
private String productName;
private int productPrice;
private String productDiscription;


public Product() {
    super();
}


public Product(String productName, int productPrice, String productDiscription) {
    super();
    this.productName = productName;
    this.productPrice = productPrice;
    this.productDiscription = productDiscription;
}


public int getProductId() {
    return productId;
}


public void setProductId(int productId) {
    this.productId = productId;
}


public String getProductName() {
    return productName;
}


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

PurchaseReport Model

```java
@Id
@GeneratedValue(strategy=GenerationType.AUTO)
private int reportId;
private String reportProductName;
private String reportUserEmail;
private String reportDate;
private int reportPrice;

public PurchaseReport() {
    super();
}

public PurchaseReport(String reportProductName, String reportUserEmail, String reportDate, int reportPrice) {
    super();
    this.reportProductName = reportProductName;
    this.reportUserEmail = reportUserEmail;
    this.reportDate = reportDate;
    this.reportPrice = reportPrice;
}

public int getReportId() {
    return reportId;
}

public void setReportId(int reportId) {
    this.reportId = reportId;
}

public String getReportProductName() {
    return reportProductName;
}

public void setReportProductName(String reportProductName) {
    this.reportProductName = reportProductName;
}

public String getReportUserEmail() {
```

User Model

```java
@Id
@GeneratedValue(strategy=GenerationType.AUTO)
private int userId;
private String userName;
private String userEmail;
private int userAge;
private String userGender;
private String userAddress;

public User() {
    super();
}

public User(String userName, String userEmail, int userAge, String userGender, String userAddress) {
    super();
    this.userName = userName;
    this.userEmail = userEmail;
    this.userAge = userAge;
    this.userGender = userGender;
    this.userAddress = userAddress;
}

public int getUserId() {
    return userId;
}

public void setUserId(int userId) {
    this.userId = userId;
}

public String getUserName() {
    return userName;
}

public void setUserName(String userName) {
    this.userName = userName;
}

public String getUserEmail() {
```

Project Repositories

Admin Repository

```java
public HibernateTemplate getHibernateTemplate() {
    return hibernateTemplate;
}

public void setHibernateTemplate(HibernateTemplate hibernateTemplate) {
    this.hibernateTemplate = hibernateTemplate;
}


public boolean verifyAdmin(Admin a) {
    Session session=hibernateTemplate.getSessionFactory().openSession();
    Transaction transaction=session.beginTransaction();

    String hql = "from Admin where email=:email and password=:password";
    Query<Admin> query = session.createQuery(hql,Admin.class);
    query.setParameter("email", a.getEmail());
    query.setParameter("password", a.getPassword());
    List<Admin> results = query.getResultList();
    transaction.commit();
    session.close();

    return results.size()>0;
}
```

Product Repository

```java
public List<Product> getAllProducts(){

    List<Product> productList=hibernateTemplate.loadAll(Product.class);
    return productList;

}

public Boolean deleteProduct(int id) {

    try {
    Session session=hibernateTemplate.getSessionFactory().openSession();
    Transaction transaction=session.beginTransaction();
    Product p=session.get(Product.class, id);
    session.delete(p);
    transaction.commit();
    session.close();
    }
    catch(Exception e) {
        return false;
    }
    return true;
}
```

Purchase Report Repository

```java
public class PurchaseReportRepository {

    HibernateTemplate hibernateTemplate;

    public HibernateTemplate getHibernateTemplate() {
        return hibernateTemplate;
    }

    public void setHibernateTemplate(HibernateTemplate hibernateTemplate) {
        this.hibernateTemplate = hibernateTemplate;
    }

    @SuppressWarnings("deprecation")
    public List<PurchaseReport> getReport(){

        @SuppressWarnings("unchecked")
        List<PurchaseReport> report=(List<PurchaseReport>) hibernateTemplate.find("from PurchaseReport order by reportProductName,reportDate");

        return report;
    }

}
```

User Repository

```java
    public List<User> getAllUsers(){

        List<User> userList=hibernateTemplate.loadAll(User.class);
        return userList;

    }


    public List<User> searchUser(String email){

        Session session=hibernateTemplate.getSessionFactory().openSession();
        Transaction transaction=session.beginTransaction();

        String hql = "from User where userEmail=:email";
        Query<User> query = session.createQuery(hql,User.class);
        query.setParameter("email", email);
        List<User> user = query.getResultList();
        transaction.commit();
        session.close();

        return user;
```

Index jsp file main screen

```jsp
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Sporty Shoes</title>
</head>
<body style="background-color:powderblue">

<h1 align="center">Sporty Shoes--An-E-commerce Website</h1><br><br>
<p align="center"><a href="adminLogin" style="font-size:25px;"><input type="button" value="A
</body>
</html>
```

Admin Table

```
1 •    SELECT * FROM sporty_shoes.admin;
```

Result Grid	Filter Rows:	Edit:

email	password
admin@gmail.com	test
NULL	NULL

Purchase Report Table

```
1 •    SELECT * FROM sporty_shoes.purchase_report;
```

Result Grid	Filter Rows:	Edit:	Export/Imp

reportId	reportDate	reportPrice	reportProductName	reportUserEmail
1	12/3/2022	5500	Nike Air	abhi@gmail.com
2	13/4/2022	4500	Addidas	abhi@gmail.com
3	13/5/2022	2500	Bata	abhi@gmail.com
NULL	NULL	NULL	NULL	NULL

User Table

```
1 •   SELECT * FROM sporty_shoes.user;
```

userId	userAddress	userAge	userEmail	userGender	userName
1	102/106 Delhi	33	raj@gmail.com	Male	Raj
2	109/106 Mumbai	37	abhi@gmail.com	Male	Abhi
3	107/108	37	ravi@gmail.com	Male	Ravi
NULL	NULL	NULL	NULL	NULL	NULL

Product Table

```
1 •   SELECT * FROM sporty_shoes.product;
```

productId	productDiscription	productName	productPrice
1	Best Shoes	Nike Air	5500
2	Best Confort Shoes	Addidas	4500
3	Best Durable Shoes	Bata	2500
NULL	NULL	NULL	NULL

Your conclusion on enhancing the application and defining the USPs (Unique Selling Points):

A website with simple user interface and user experience.

A website which admin to maintain data of the website.

Admin can its password by clicking on forgot password.

Admin can see the product, purchase reports and users in website (track all data of the website).

Admin can delete the product also.

Admin can search the user by their email id.

Some USPs –

The application works very smoothly while taking inputs from the user.

The admin can easily.

The admin can change the password if he forgets

The admin can track all the data on the website such as see available products, users, Purchase reports.

The admin can search the users by their emails.

The admin can delete the products also.