Sprint 1

Day 1:-

Created the maven webapp front end

Added spring mvc dependencies.

Create the MVC structure--> com.HiFiShopee.controller, web-inf/views/index.jsp [bootstrap 3]

Created web.xml

Created dispatcher-servlet.xml

Day 2 :-

Created the maven quick start back end project

Added dependencies in back end pom.xml

{spring-core, spring-web, spring-tx, hibernate-core, entitymanager, validator,

etc etc}.Also added the backend pom jar to front end pom.

Created com.shoppingbackend.model,com.shoppingbackend.config,com.shoppingbackend.dao, com.shoppingbackend.dao.impl packages.

Day 3:-

Configured hiberConfig.java

a) created autowired beans for database connectivity

b) created autowired beans for hibernate dialect and syntax

c) created autowired beans for hibernate transaction management

Sprint 2

Day 1:-

Create the user,address,cart,cartline,supplier,orderdetail,orderitem,category model class,

Create the table in H2

Create CRUD method in DAO--> Defined it in DaoImpl

Day 2:-

Added the autowired DaoImpl bean in hiberConfig.java

Created the register web-flow to direct user to fill up forms for address as per the attributed in User,Address class using Spring form

Day 3:-

Created the controller methods to save the registration data by taking it from spring form and sending it as user object parameter to CRUD method of DaoImpl.

Created the supplier and category tables in H2.

Created Supplier and Category Dao, DaoImpl.

Included Supplier and Category DaoImpl @Autowired bean in hiberConfig.java

Sprint 3

Day 1:-

Created manageProduct.jsp (SPA) with supplier and category tab and forms.

Created controller methods for Supplier and Category CRUD.

Performed CRUD operations for Supplier and Category.

insert()

retrieve<ArrayList>()

getByID()

Day 2:-

Created Product.java(MultipartFile-->Transient), ProductDao.java, ProductDaoImpl.java

insert()

retrieve<ArrayList>()

getByID()

Day 3:-

Add product form in the of SPA along with multipart file upload option.

Create the controller methods for calling all the crud operations.

Sprint 4

Day 1:-

Create the login.jsp

Create the login option in header.jsp on top navbar

Write the controller method to go to the login.jsp(String +@RequestMapping)

Day 2:-

Implement spring security in xml configuration file to intercept every user request and check the authenticationa and authorization of user.

Add spring security dependencies in pom.xml

Add the filterChainProxy class in web.xml

Add the filter-mapping of security class in web.xml

Day 3:-

Modify all CRUD operation URLs and prefix "/admin" to each of the URLs

Add the userLogged() and error() in adminController()

Display the username/email/id on the top navbar once any user logs in

(hint: use--> userPrincipal.name in header.jsp)

Create the CategoryList dropdown on top navbar which will be populated by category names. Clicking on each category should take us to productCustList.jsp having the filtered product list based on category id.

Sprint 5

Day 1:-

Created Cart.java and Order.java

Created CartDao.java and OrderDao.java

Day 2:-

Created CartDaoImpl.java and CartDaoImpl.java

Added the annotated Cart and Order model class in hiberConfig.java

Day 3:-

Added the cart.jsp

Created checkout flow to maintain the logical flow of form where user could select at which address product should be delivered.

Sprint 6

Day 1:-

Added cart controller for /prodDetails,/addToCart, /checkout,/invoice, /deletePCart,/goToCart/Orderprocess

Added cart in header.jsp

Day 2:-

Add hibernate/java validation dependencies in your pom.xml and add the validation annotation in your model classes of Category/Product/Supplier/User against every relevant attribute. {@Size,@NotNull, @Len, @Pattern}

Day 3:-

You will add the BindingResult object and @Valid annotation in the Controller method where you are using those attributes for any CRUD operation.

Create css, and js on JSP pages for all forms to show error messages if invalid data in entered.