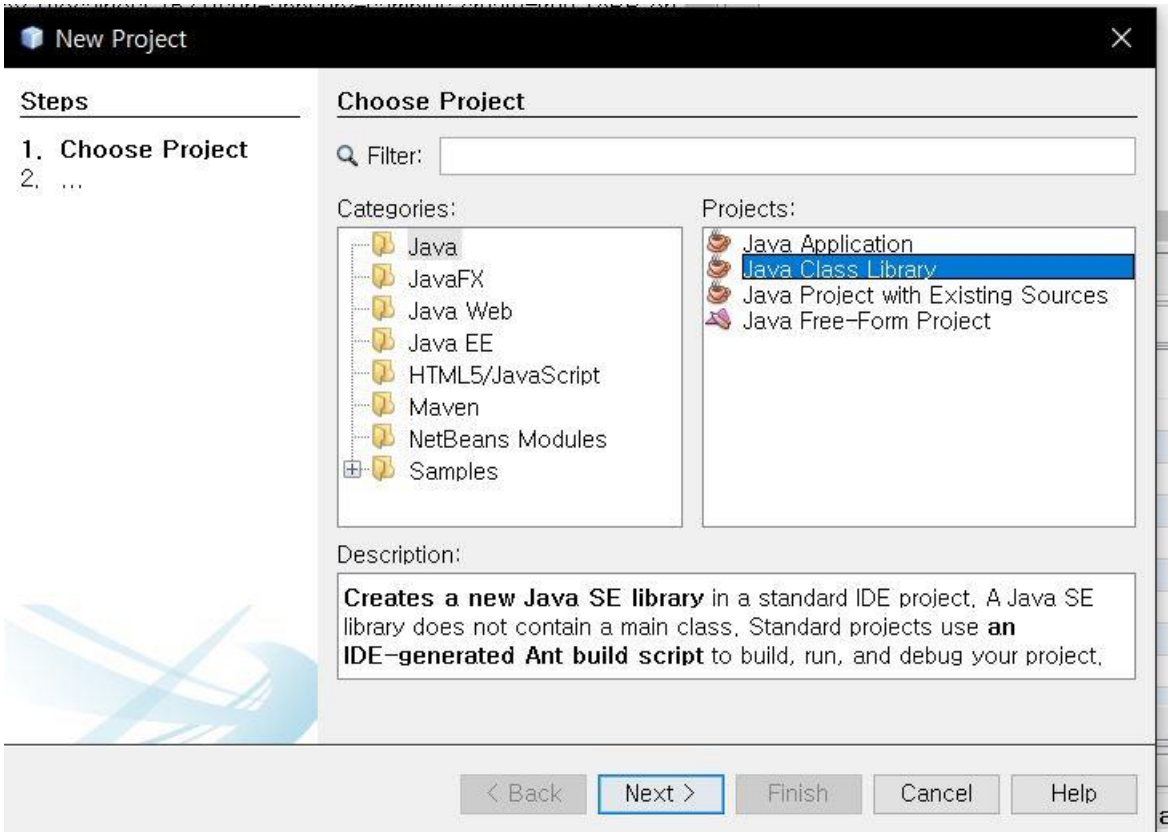
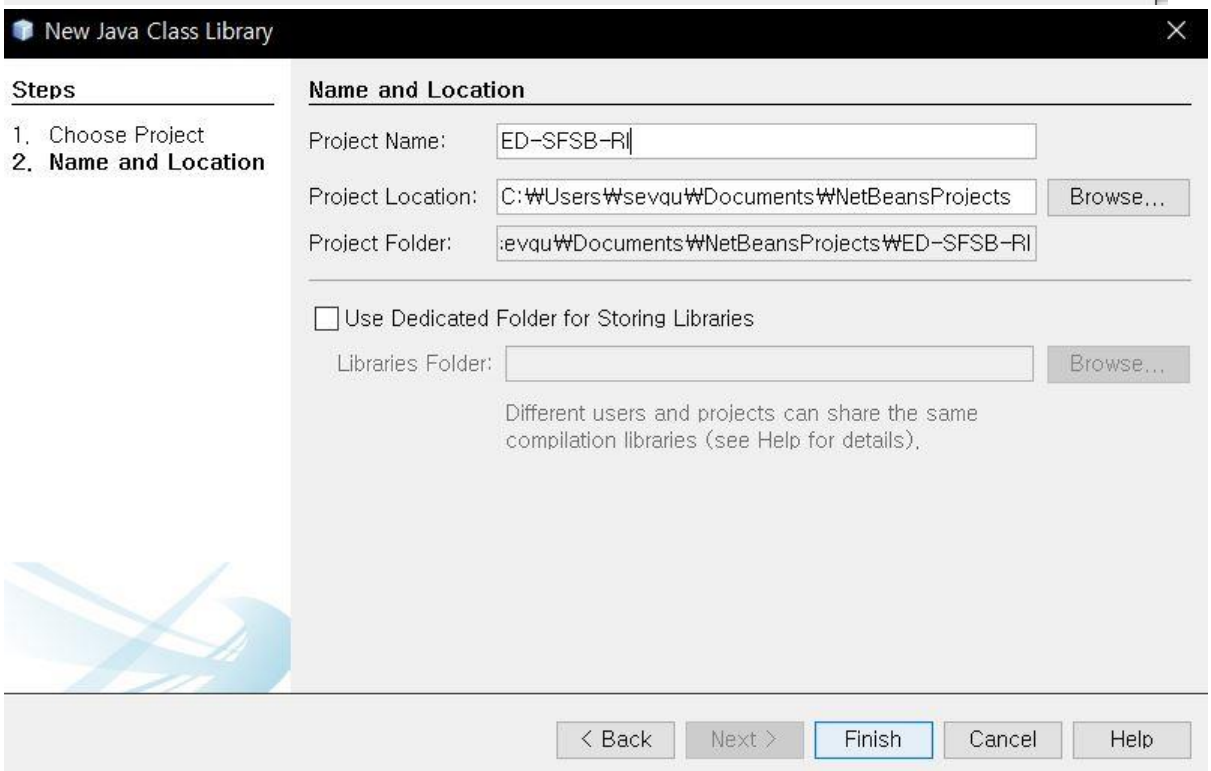
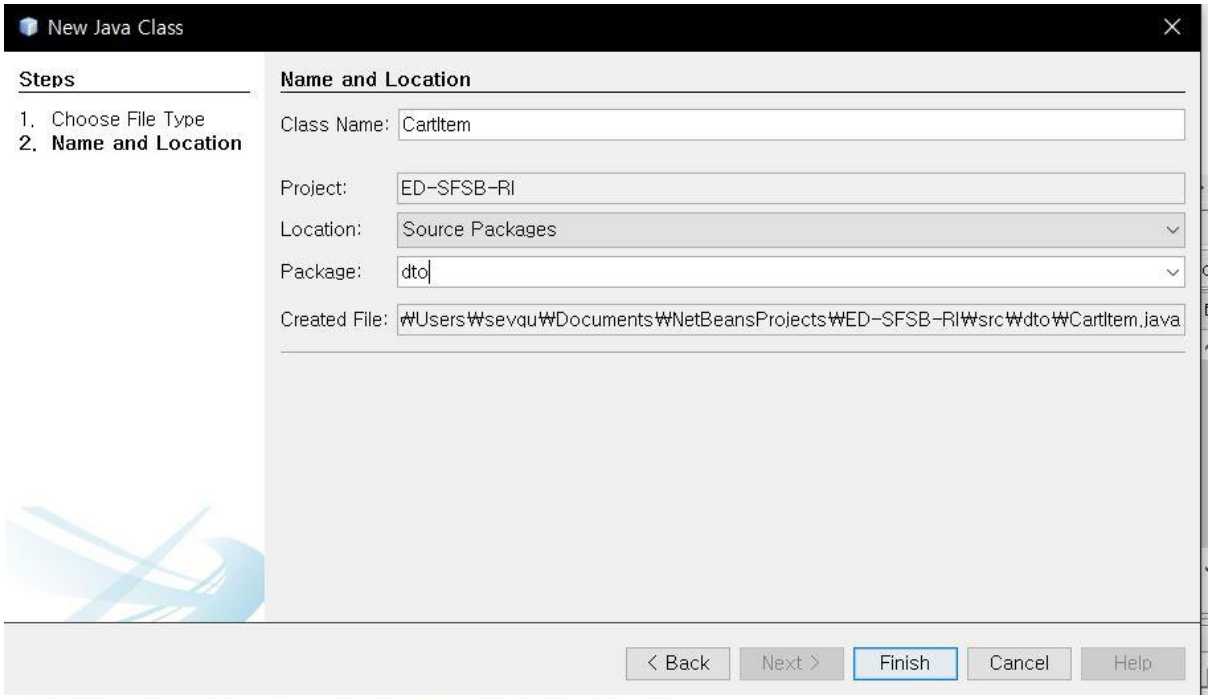
Task 1

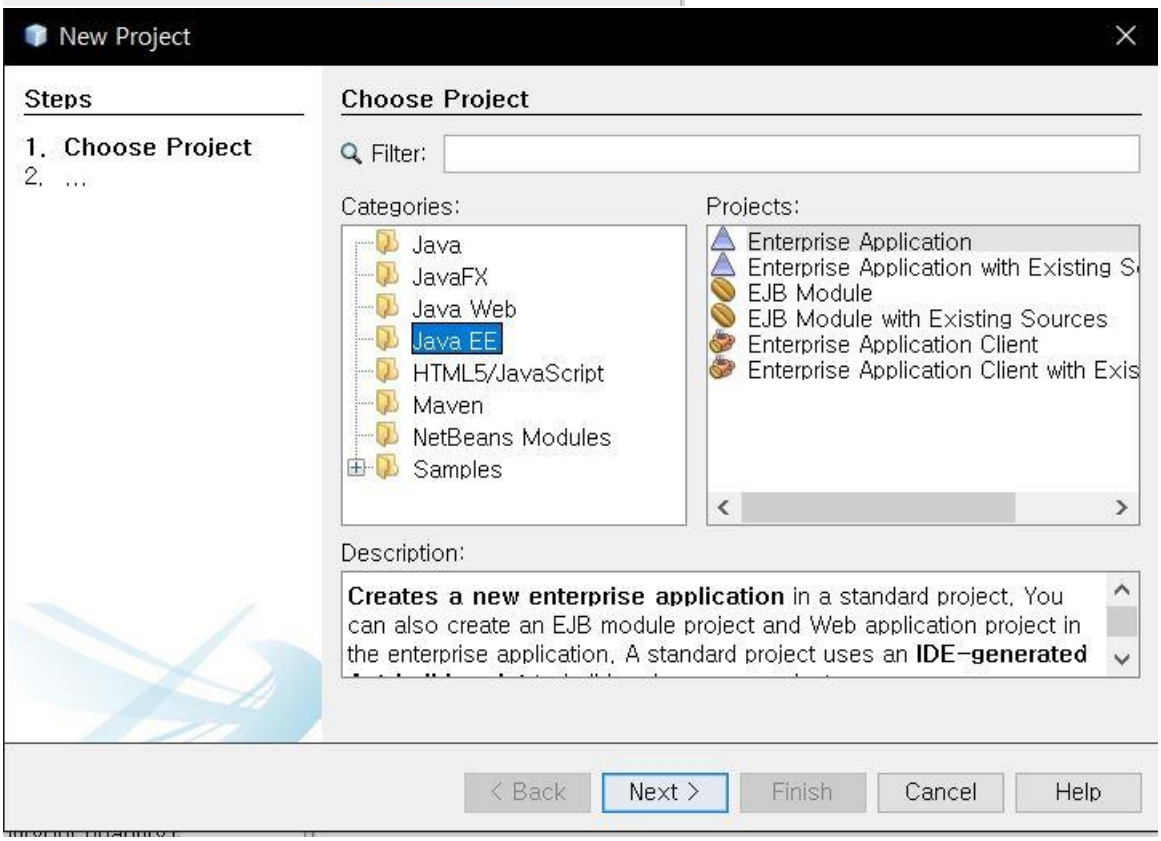

New Project
Steps
1. Choose Project
2. ...
Choose Project
Filter:
Categories:
Java
JavaFX
Java Web
Java EE
HTML5/JavaScript
Maven
NetBeans Modules
Samples
Projects:
Java Application
Java Class Library
Java Project with Existing Sources
Java Free-Form Project
Description:
Creates a new Java SE library in a standard IDE project. A Java SE library does not contain a main class. Standard projects use an IDE-generated Ant build script to build, run, and debug your project.
< Back
Next >
Finish
Cancel
Help

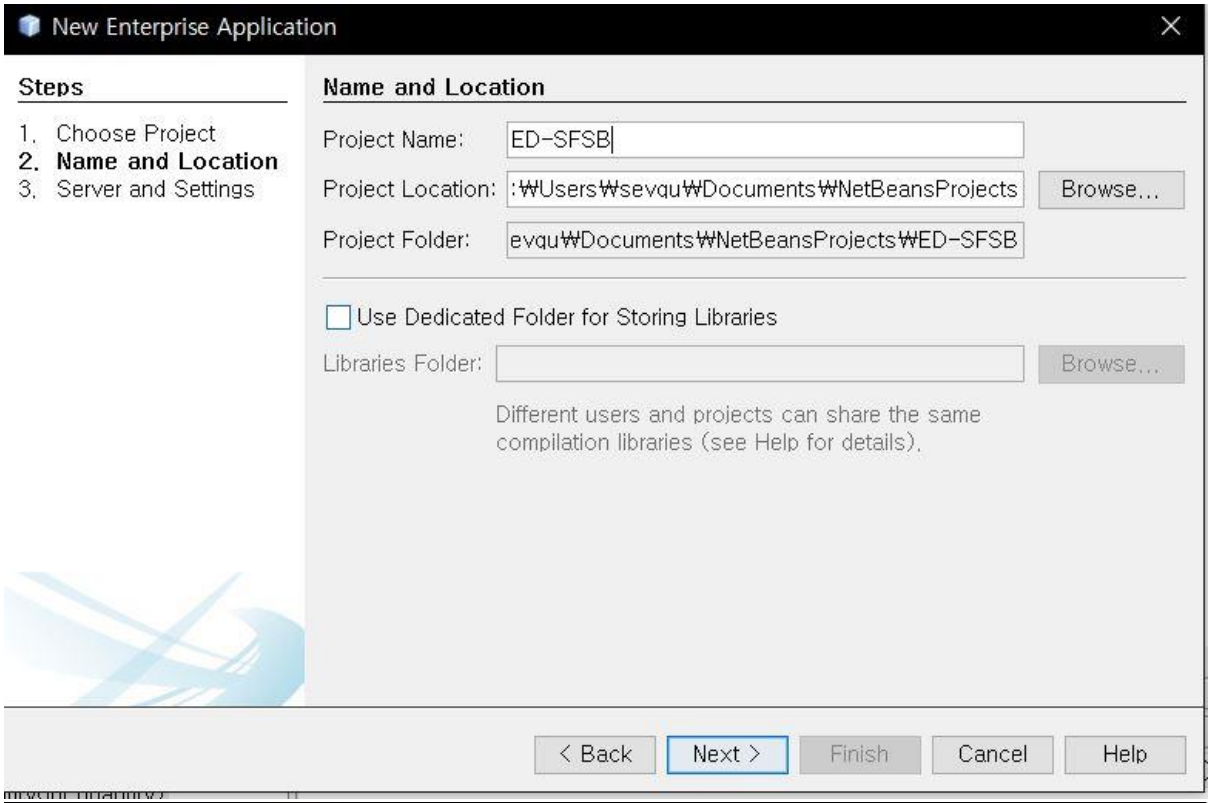

New Java Class Library
Steps
1. Choose Project
2. Name and Location
Name and Location
Project Name: ED-SFSB-RI
Project Location: C:₩Users₩sevqu₩Documents₩NetBeansProjects
Browse...
Project Folder: :evqu₩Documents₩NetBeansProjects₩ED-SFSB-RI
Use Dedicated Folder for Storing Libraries
Libraries Folder:
Browse...
Different users and projects can share the same compilation libraries (see Help for details).
< Back
Next >
Finish
Cancel
Help

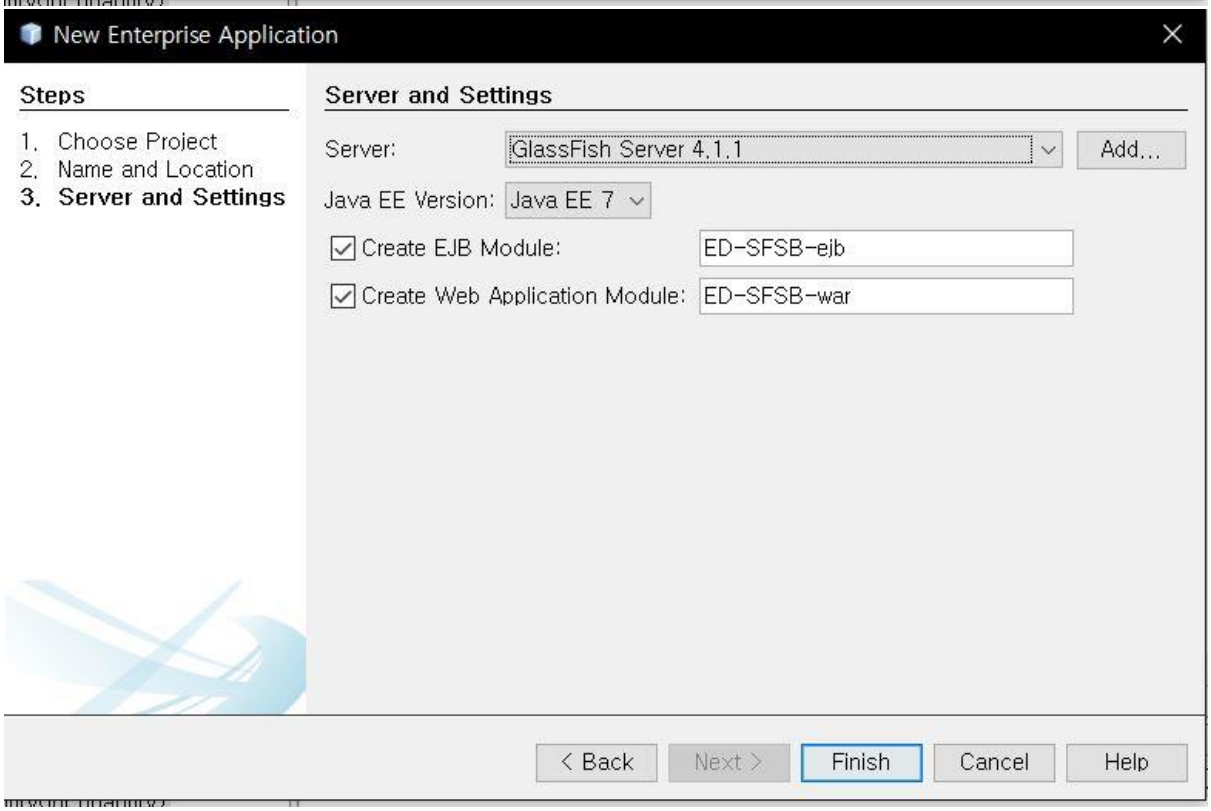


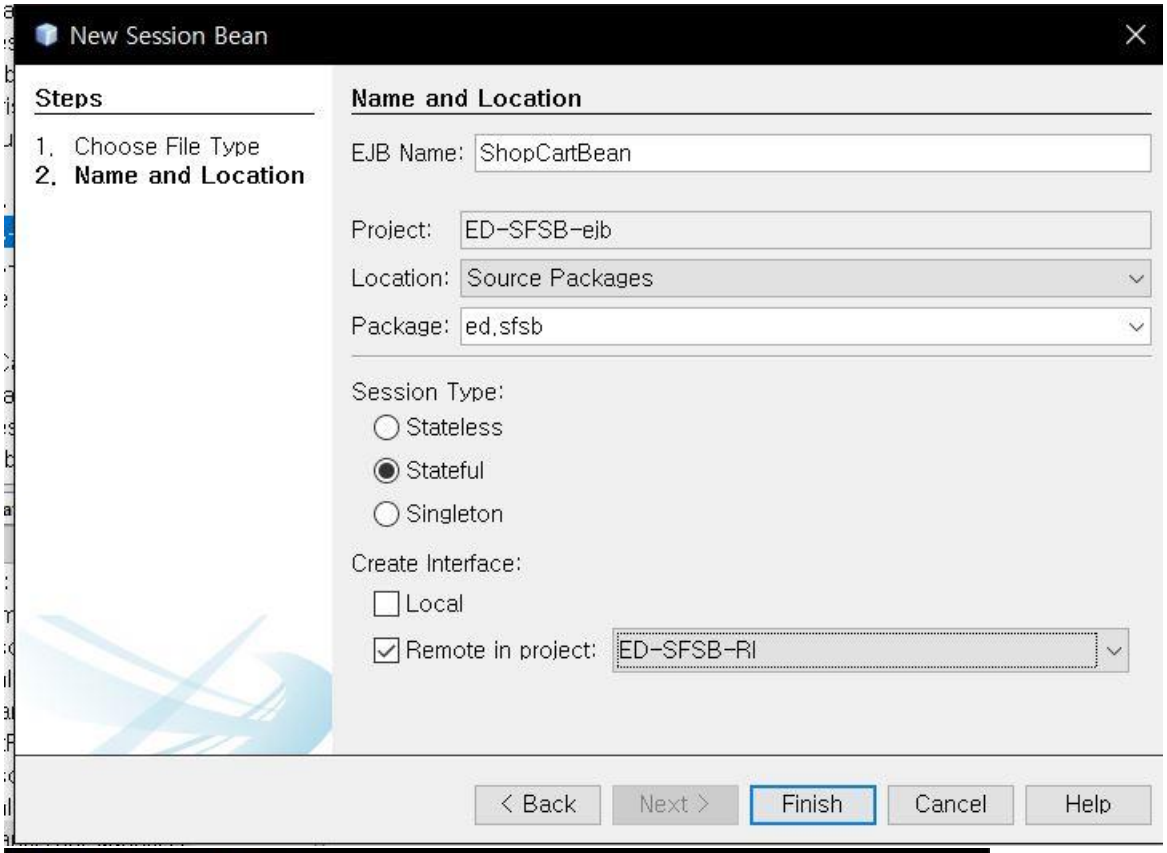

```java
public class CartItem implements Serializable {

    private String itemId;
    private String description;
    private double unitPrice;
    private int quantity;

    public CartItem(String id, String desc, double price, int qty) {
        itemId = id;
        description = desc;
        unitPrice = price;
        quantity = qty;
    }
}
```

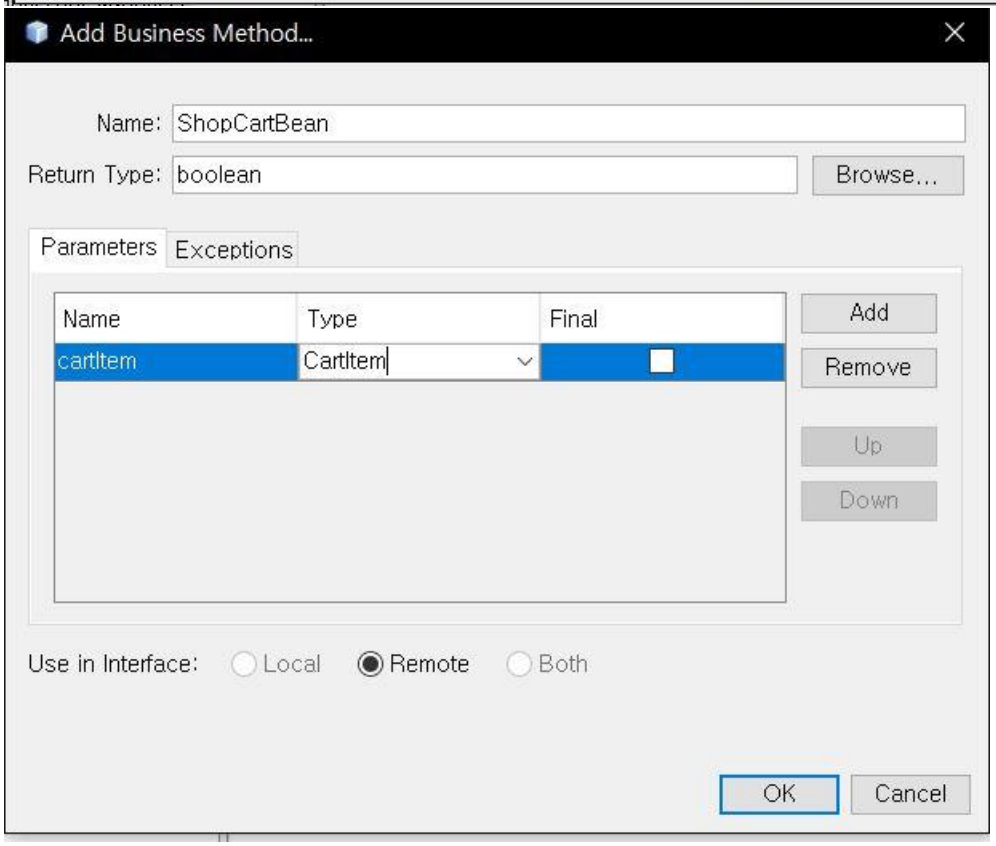

Generate Getters and Setters
Select fields to generate getters and setters for:
CartItem
description : String
itemId : String
quantity : int
unitPrice : double
Select All
Select None
Encapsulate Fields
Generate
Cancel


New Project
Steps
1. Choose Project
2. ...
Choose Project
Filter:
Categories:
Java
JavaFX
Java Web
Java EE
HTML5/JavaScript
Maven
NetBeans Modules
Samples
Projects:
Enterprise Application
Enterprise Application with Existing S
EJB Module
EJB Module with Existing Sources
Enterprise Application Client
Enterprise Application Client with Exis
Description:
Creates a new enterprise application in a standard project. You can also create an EJB module project and Web application project in the enterprise application. A standard project uses an IDE-generated
< Back
Next >
Finish
Cancel
Help

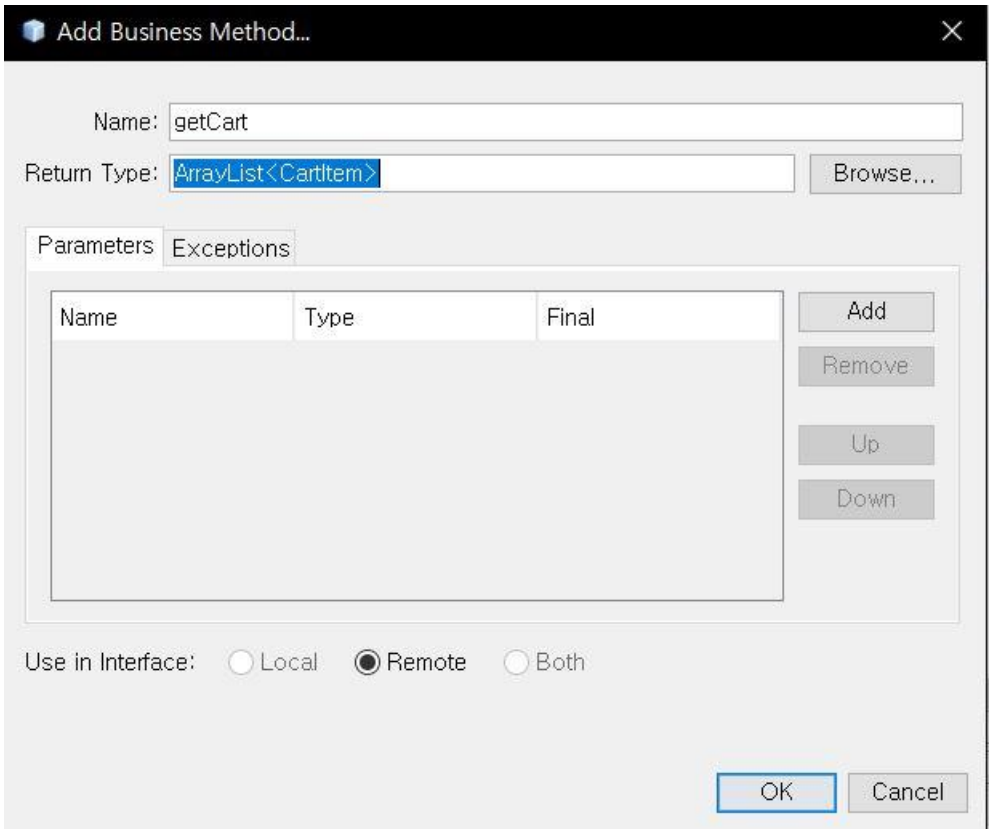

New Enterprise Application
Steps
1. Choose Project
2. Name and Location
3. Server and Settings
Name and Location
Project Name: ED-SFSB
Project Location: :₩Users₩sevqu₩Documents₩NetBeansProjects
Browse...
Project Folder: evqu₩Documents₩NetBeansProjects₩ED-SFSB
Use Dedicated Folder for Storing Libraries
Libraries Folder:
Browse...
Different users and projects can share the same compilation libraries (see Help for details).
< Back
Next >
Finish
Cancel
Help

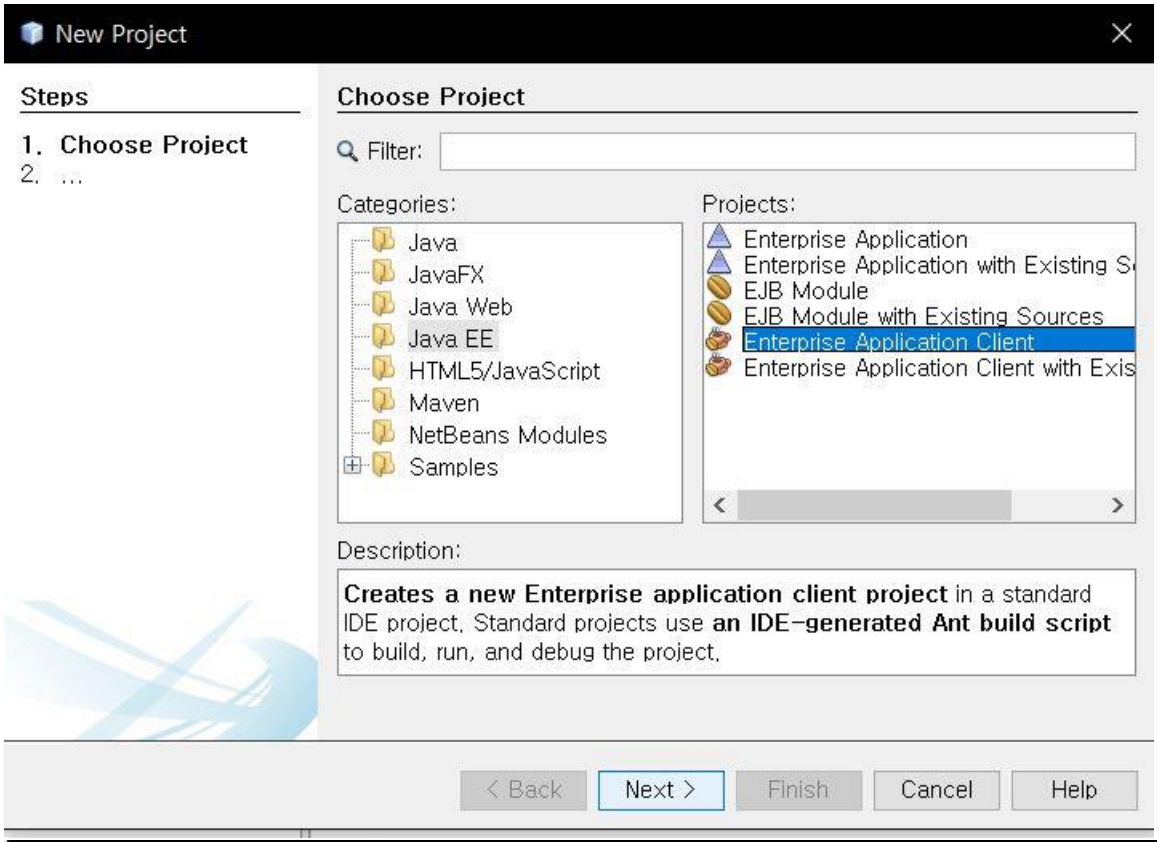

New Enterprise Application
Steps
1. Choose Project
2. Name and Location
3. Server and Settings
Server and Settings
Server: GlassFish Server 4.1.1
Add...
Java EE Version: Java EE 7
Create EJB Module: ED-SFSB-ejb
Create Web Application Module: ED-SFSB-war
< Back
Next >
Finish
Cancel
Help

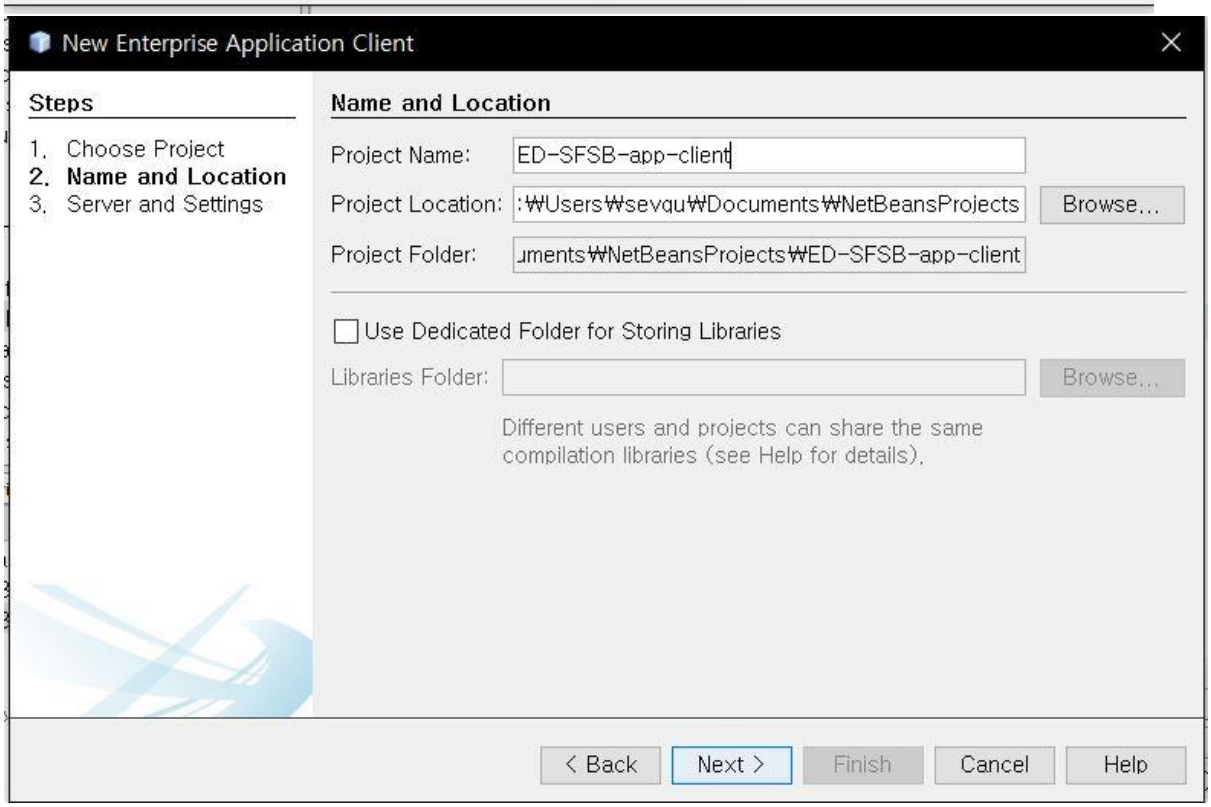

New Session Bean
Steps
1. Choose File Type
2. Name and Location
Name and Location
EJB Name: ShopCartBean
Project: ED-SFSB-ejb
Location: Source Packages
Package: ed.sfsb
Session Type:
Stateless
Stateful
Singleton
Create Interface:
Local
Remote in project: ED-SFSB-RI
< Back
Next >
Finish
Cancel
Help

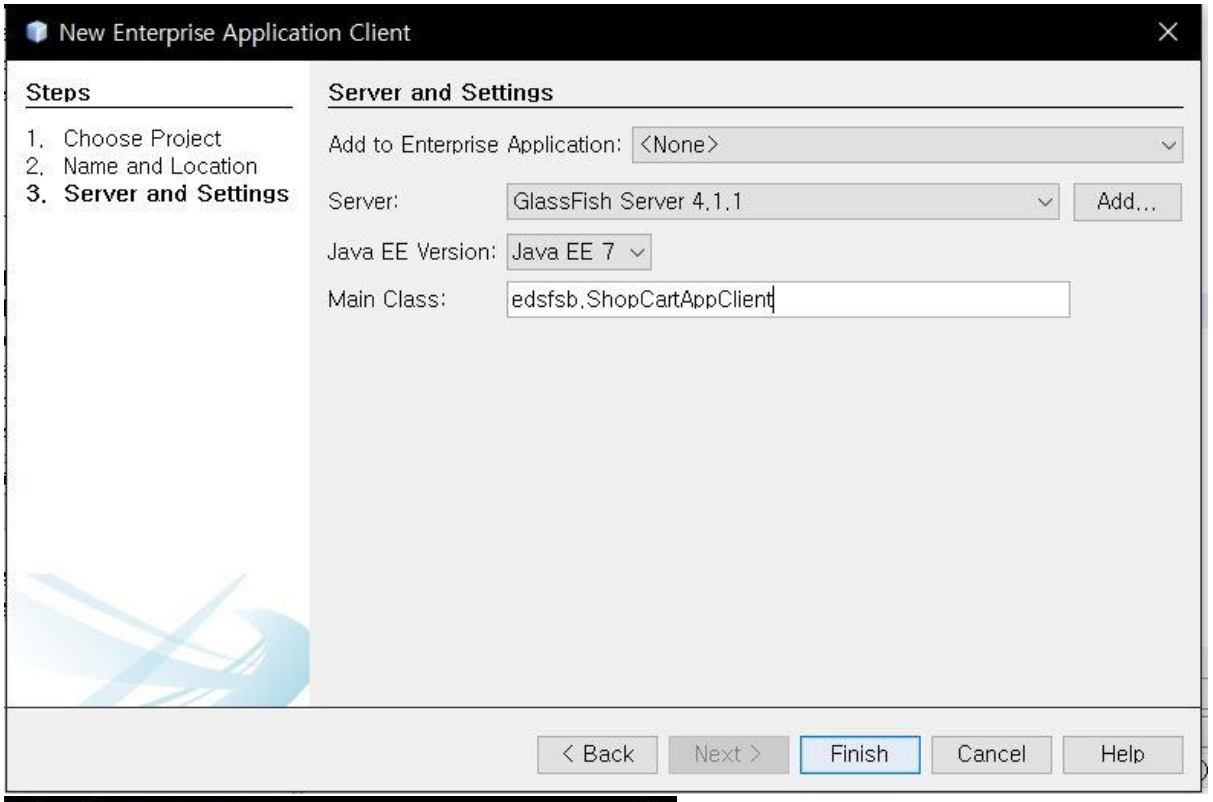

Add Business Method...
Name: ShopCartBean
Return Type: boolean
Browse...
Parameters
Exceptions
Name
Type
Final
cartItem
CartItem
Add
Remove
Up
Down
Use in Interface:
Local
Remote
Both
OK
Cancel

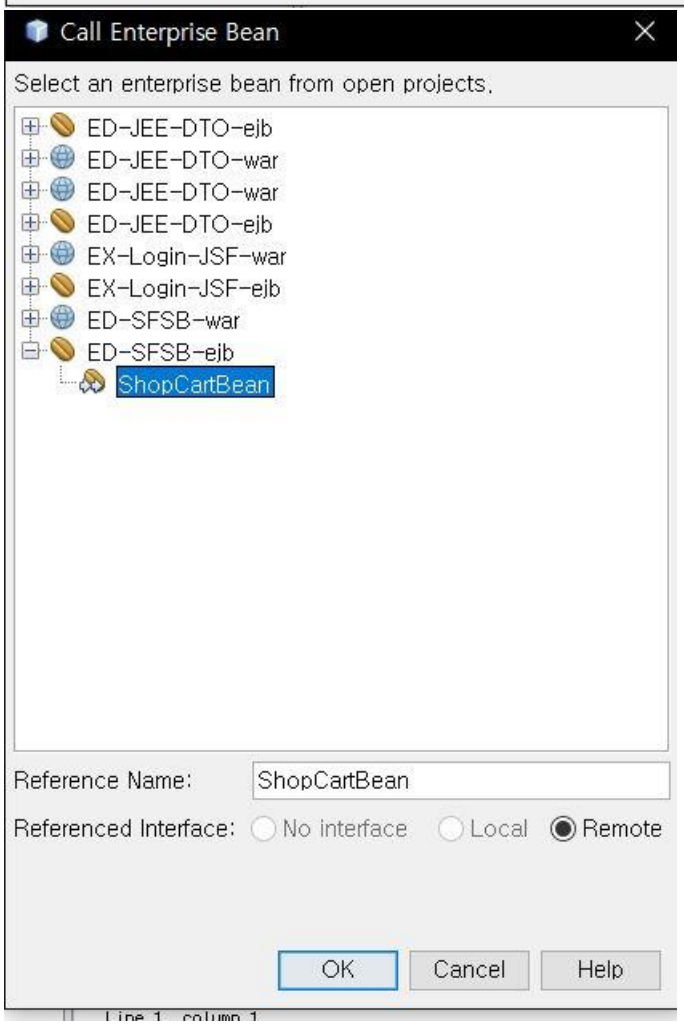
```
    @Override
    public boolean ShopCartBean(CartItem cartItem) {
        boolean result = false;
        try {
            result = cart.add(cartItem);
        } catch (Exception ex) {
        }
        return result;
    }

}
```




New Project
Steps
1. Choose Project
2. ...
Choose Project
Filter:
Categories:
Java
JavaFX
Java Web
Java EE
HTML5/JavaScript
Maven
NetBeans Modules
Samples
Projects:
Enterprise Application
Enterprise Application with Existing S
EJB Module
EJB Module with Existing Sources
Enterprise Application Client
Enterprise Application Client with Exis
Description:
Creates a new Enterprise application client project in a standard IDE project. Standard projects use an IDE-generated Ant build script to build, run, and debug the project.
< Back
Next >
Finish
Cancel
Help


New Enterprise Application Client
Steps
1. Choose Project
2. Name and Location
3. Server and Settings
Name and Location
Project Name: ED-SFSB-app-client
Project Location: :₩Users₩sevqu₩Documents₩NetBeansProjects
Browse...
Project Folder: ₩NetBeansProjects₩ED-SFSB-app-client
Use Dedicated Folder for Storing Libraries
Libraries Folder:
Browse...
Different users and projects can share the same compilation libraries (see Help for details).
< Back
Next >
Finish
Cancel
Help


New Enterprise Application Client
Steps
1. Choose Project
2. Name and Location
3. Server and Settings
Server and Settings
Add to Enterprise Application: <None>
Server: GlassFish Server 4.1.1
Add...
Java EE Version: Java EE 7
Main Class: edsfsb.ShopCartAppClient
< Back
Next >
Finish
Cancel
Help


Call Enterprise Bean
Select an enterprise bean from open projects.
ED-JEE-DTO-ejb
ED-JEE-DTO-war
ED-JEE-DTO-war
ED-JEE-DTO-ejb
EX-Login-JSF-war
EX-Login-JSF-ejb
ED-SFSB-war
ED-SFSB-ejb
ShopCartBean
Reference Name: ShopCartBean
Referenced Interface: No interface Local Remote
OK
Cancel
Help

```
    @EJB
    private static ShopCartBeanRemote shopCart;
```

```
do-compile:
post-compile:
compile:
pre-dist:
post-dist:
dist-directory-deploy:
pre-run-deploy:
Initial deploying ED-SFSB to C:\Users\sevqu\Documents\NetBeansProjects\ED-SFSB\dist\gfdeploy\ED-SFSB
Completed initial distribution of ED-SFSB
post-run-deploy:
run-deploy:
BUILD SUCCESSFUL (total time: 2 seconds)
```

```
The shopping cart is empty!
Adding item Intel Core i7 CPU to cart
Your order of 2 Intel Core i7 CPU has been added.
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349,99      Quantity: 2     Sub-Total: 699,98
---
Total price: 699,98
----End of Shopping Cart---
Adding item Intel SSD 512GB to cart
Your order of 3 Intel SSD 512GB has been added.
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349,99      Quantity: 2     Sub-Total: 699,98
Item: Intel SSD 512GB   Unit Price: 299,99      Quantity: 3     Sub-Total: 899,97
---
Total price: 1599,95
----End of Shopping Cart---
run:
BUILD SUCCESSFUL (total time: 26 seconds)
```

Task 2

```
@Override
public boolean addCartItem(CartItem cartItem) {
    CartItem old = null;
    boolean result = false;
    if (cartItem != null) {
        for (int i = 0; i < cart.size(); i++) {
            old = cart.get(i);
            if (old.getItemId().equals(cartItem.getItemId())) {
                //If it is repeated, it will be accumulated.
                old.setQuantity(old.getQuantity() + cartItem.getQuantity());
                result = true;

                return result;
            }
        }
        cart.add(cartItem);
        result = true;
        return result;
    }
    return result;
}
```

```
@Override
public boolean deleteCartItem(String itemId) {
    CartItem old = null;
    boolean result = false;
    for (int i = 0; i < cart.size(); i++) {
        old = cart.get(i);
        if (old.getItemId().equals(itemId)) {
            cart.remove(i);
            result = true;
            return result;
        }
    }
    return result;
}
```

```java
@Override
public boolean updateCartItem(CartItem cartItem) {
    CartItem old = null;
    boolean result = false;
    if (cartItem != null) {
        for (int i = 0; i < cart.size(); i++) {
            old = cart.get(i);
            if (old.getItemId().equals(cartItem.getItemId())) {
                //If it is repeated, it will be accumulated.
                // cart.remove(i);
                //cart.add(cartItem);
                old.setItemId(cartItem.getItemId());
                old.setDescription(cartItem.getDescription());
                old.setUnitPrice(cartItem.getUnitPrice());
                old.setQuantity(cartItem.getQuantity());


                result = true;

                return result;
            }
        }
        return result;
    }
    return result;
}
```

ShopCartBeanRemote.java

```java
ArrayList<CartItem> getCart();

public void remove();

public boolean add(CartItem cartItem);

public boolean addCartItem(CartItem cartItem);

public boolean deleteCartItem(String itemId);

public boolean updateCartItem(CartItem cartItem);
```

Task 3

```
CartItem item1 = new CartItem("000001", "Intel Core i7 CPU", 349.99, 2);
CartItem item2 = new CartItem("000002", "Intel SSD 512GB", 299.99, 3);
appClient.addCart(item1);
appClient.displayCart();
appClient.addCart(item2);
appClient.displayCart();
CartItem item3 = new CartItem("000001", "Intel Core i7 CPU", 349.99, 2);
CartItem item4 = new CartItem("000003", "Intel SSD 128GB", 149.99, 1);
appClient.addCartItem(item3);
appClient.displayCart();
appClient.addCartItem(item4);
appClient.displayCart();
String id1 = "000002";
String id2 = "000006";
appClient.deleteCartItem(id1);
appClient.displayCart();
appClient.deleteCartItem(id2);
appClient.displayCart();
CartItem item5 = new CartItem("000009", "Intel Core i7 CPU", 349.99, 2);
CartItem item6 = new CartItem("000003", "Intel SSD 1TB", 429.99, 5);
appClient.updateCartItem(item5);
appClient.displayCart();
appClient.updateCartItem(item6);
appClient.displayCart();
```

```
    public void addCartItem(CartItem cartItem) {
        System.out.println("Adding item " + cartItem.getDescription() + " to cart");
        if (shopCart.addCartItem(cartItem)) {
            System.out.println("Your order of " + cartItem.getQuantity()
                    + " " + cartItem.getDescription() + " has been added. ");
        } else {
            System.out.println("Sorry, your order of " + cartItem.getQuantity() + " "
                    + cartItem.getDescription() + " cannot be added due to low stock. ");
        }
    }

    public void deleteCartItem(String itemId) {

        if (shopCart.deleteCartItem(itemId)) {
            System.out.println(itemId + " has been deleted from the cart");
        } else {
            System.out.println(itemId + " couldn't been found in the cart");
        }

    }

    public void updateCartItem(CartItem cartItem) {
        if (shopCart.updateCartItem(cartItem)) {
            System.out.println(cartItem.getItemId() + " has been updated from the cart");
        } else {

            System.out.println(cartItem.getItemId() + " couldn't been updated from the cart");
        }
    }
```

Task 4

```
The shopping cart is empty!
Adding item Intel Core i7 CPU to cart
Your order of 2 Intel Core i7 CPU has been added,
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349,99      Quantity: 2     Sub-Total: 699,98
---
Total price: 699,98
----End of Shopping Cart---
Adding item Intel SSD 512GB to cart
Your order of 3 Intel SSD 512GB has been added,
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349,99      Quantity: 2     Sub-Total: 699,98
Item: Intel SSD 512GB   Unit Price: 299,99      Quantity: 3     Sub-Total: 899,97
---
Total price: 1599,95
----End of Shopping Cart---
Adding item Intel Core i7 CPU to cart
Your order of 2 Intel Core i7 CPU has been added,
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349,99      Quantity: 4     Sub-Total: 1399,96
Item: Intel SSD 512GB   Unit Price: 299,99      Quantity: 3     Sub-Total: 899,97
---
Total price: 2299,9300000000003
----End of Shopping Cart---
Adding item Intel SSD 128GB to cart
Your order of 1 Intel SSD 128GB has been added,
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349,99      Quantity: 4     Sub-Total: 1399,96
Item: Intel SSD 512GB   Unit Price: 299,99      Quantity: 3     Sub-Total: 899,97
Item: Intel SSD 128GB   Unit Price: 149,99      Quantity: 1     Sub-Total: 149,99
---
Total price: 2449,92
----End of Shopping Cart---
000002 has been deleted from the cart
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349,99      Quantity: 4     Sub-Total: 1399,96
Item: Intel SSD 128GB   Unit Price: 149,99      Quantity: 1     Sub-Total: 149,99
---
Total price: 1549,95
----End of Shopping Cart---
000006 couldn't been found in the cart
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349,99      Quantity: 4     Sub-Total: 1399,96
Item: Intel SSD 128GB   Unit Price: 149,99      Quantity: 1     Sub-Total: 149,99
---
```

```
Total price: 1549,95
----End of Shopping Cart---
000006 couldn't been found in the cart
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349,99      Quantity: 4     Sub-Total: 1399,96
Item: Intel SSD 128GB   Unit Price: 149,99      Quantity: 1     Sub-Total: 149,99
---
Total price: 1549,95
----End of Shopping Cart---
000009 couldn't been updated from the cart
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349,99      Quantity: 4     Sub-Total: 1399,96
Item: Intel SSD 128GB   Unit Price: 149,99      Quantity: 1     Sub-Total: 149,99
---
Total price: 1549,95
----End of Shopping Cart---
000003 has been updated from the cart
Your shopping cart has the following items:
Item: Intel Core i7 CPU Unit Price: 349,99      Quantity: 4     Sub-Total: 1399,96
Item: Intel SSD 1TB     Unit Price: 429,99      Quantity: 5     Sub-Total: 2149,95
---
Total price: 3549,91
----End of Shopping Cart---
run:
BUILD SUCCESSFUL (total time: 27 seconds)
```

Task 5

Database Table

1. PRODUCT – productid(PK), productDescription, price(PK), quantity, etc
2. ORDER – orderid(PK), productid(FK), price(FK), quantity, sum, address, customerid(FK), shipping, etc
3. CUSTOMER – customerid(PK), address, phone, email, etc
4. PAYMENT – paymentid, cardNo, cardType, cardCompany, customerid(FK), etc

Entity Class

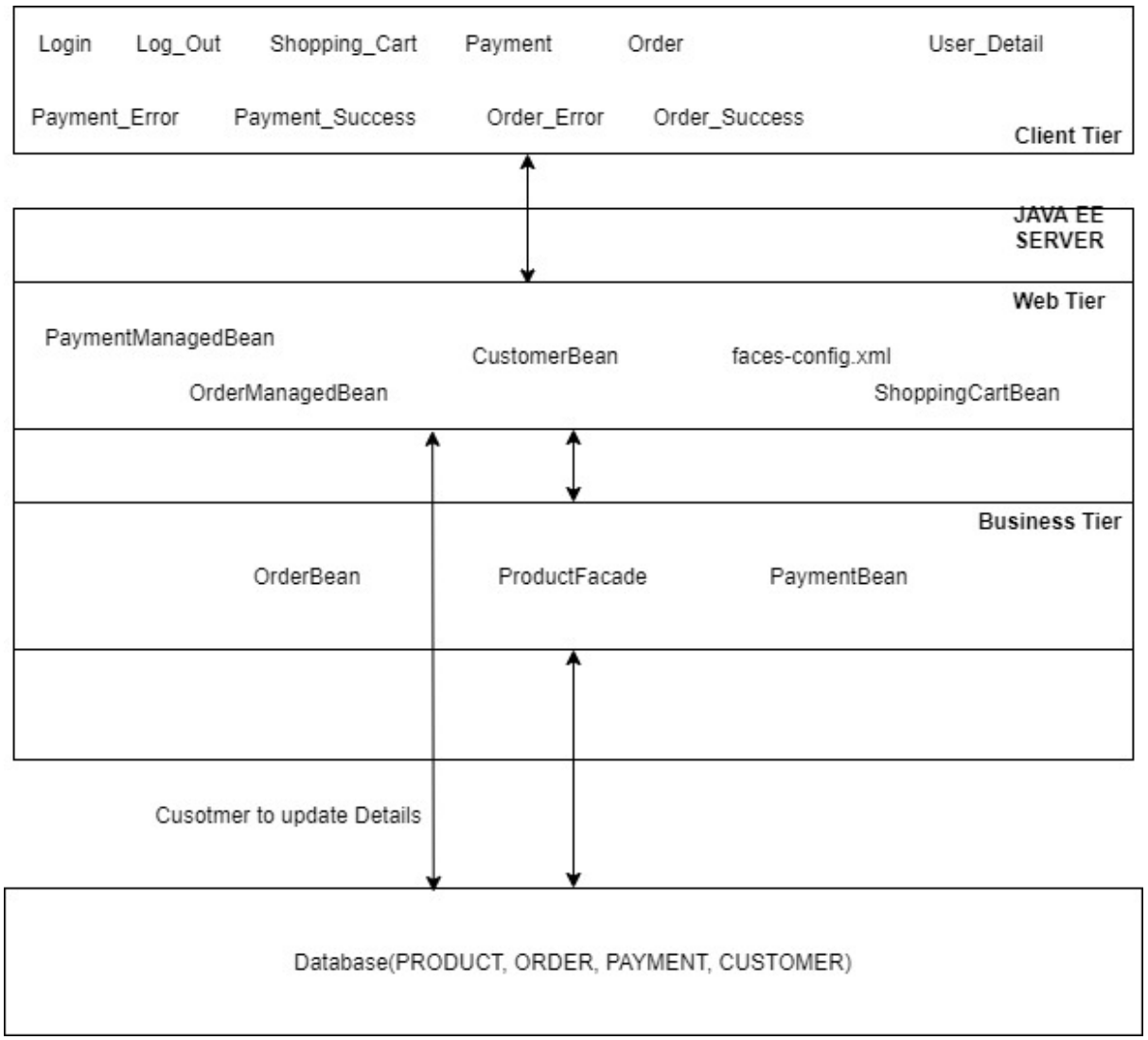
1. Product
2. Order
3. Customer
4. Payment

Session Bean

1. ShoppingCartBean – Stateful

2. ProductFacade – Stateless
3. CustomerBean – Stateful
4. PaymentBean – Stateless
5. PaymentManagedBean – Stateless
6. OrderBean – Stateless
7. OrderManagedBean - Stateless

5.1



5.2

Client Tier

1. Login – Log in Page
2. Log_Out – Log out page
3. Shopping_Cart – Checking shopping cart
4. Payment – User to decide/enter the payment details
5. Order – User can check their order
6. User_Detail – User can update their personal details
7. Payment_Error – When payment info is wrong, this page will be loaded
8. Payment_Success – when payment is successful, this message will come up and add details to Order page
9. Order_Error – when user making order of products, if any error occurs, this page will be loaded
10. Order_Success – when order is successful, this page will show the order details before making a payment

Java EE Server

1. PaymentManagedBean – Get users' payment details from users and check the validation and deliver to PaymentBean which is linked to Database(PAYMENT).
2. OrderManagedBean – Get order details from users such as address, shipping, etc and check the validation
3. CustomerBean – users/customers must log in to order the products and this bean is used to perform log in process. Moreover, only this bean is linked to database without passing business tier so that users/customers can update their personal details.
4. Faces-config.xml – this will guide whether the order or payment is successful or failed.
5. ShoppingCartBean – Allow user to add the product into the cart

Business Tier

1. OrderBean – it gets order details from OrderManagedBean and update the details to database.

2. ProductFacade – it gets information (product id, product quantity) from ShoppingCartBean and update(substract) the PRODUCT database.
3. PaymentBean – it gets payment details from PaymentManagedBean and update the details to database.

Database

It contains 4 database table which are PRODUCT, CUSTOMER, ORDER and PAYMENT. All the information will be stored in the databases.