[Template for weekly studio submission]

Studio 3

Name and Student Id: Ke Zhang, 29218969

Self-Evaluation {To be highlighted by Student only}:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Need Help | Work in Progress | Pass | Credit | Distinction | High Distinction |

Tutor Evaluation {To be highlighted by Tutor only}:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Incomplete | Needs improvement | Pass | Credit | Distinction | High Distinction |

Task 1.1:

Modified JPAPropertyRepositoryImpl.java file highlighting changes with comments

1. **package** fit5042.tutex.repository;
3. **import** java.util.ArrayList;
4. **import** java.util.HashSet;
5. **import** java.util.List;
6. **import** java.util.Set;
7. **import** java.util.stream.Collectors;
9. **import** javax.ejb.Stateless;
11. **import** fit5042.tutex.repository.constants.CommonInstance;
12. **import** fit5042.tutex.repository.entities.ContactPerson;
13. **import** fit5042.tutex.repository.entities.Property;
15. @Stateless
16. **public** **class** JPAPropertyRepositoryImpl **implements** PropertyRepository {
17. **private** ArrayList<Property> propertyList;
19. **public** JPAPropertyRepositoryImpl() {
20. propertyList = **new** ArrayList<Property>();
21. **this**.initialisePropertyList();
22. }
24. **public** **void** initialisePropertyList() {
25. propertyList.clear();
27. propertyList.add(CommonInstance.PROPERTY\_FIRST);
28. propertyList.add(CommonInstance.PROPERTY\_SECOND);
29. propertyList.add(CommonInstance.PROPERTY\_THIRD);
30. propertyList.add(CommonInstance.PROPERTY\_FOURTH);
31. }
33. **public** ArrayList<Property> getPropertyList() {
34. **return** propertyList;
35. }
37. **public** **void** setPropertyList(ArrayList<Property> propertyList) {
38. **this**.propertyList = propertyList;
39. }
41. **public** **void** removeProperty(**int** propertyId) {
42. **for** (Property p : propertyList) {
43. **if** (p.getPropertyId() == propertyId) {
44. propertyList.remove(p);
45. **break**;
46. }
47. }
49. }
51. **public** **void** addProperty(Property property) {
52. propertyList.add(property);
53. }
55. **public** **void** editProperty(Property property) {
56. **for** (Property p : propertyList) {
57. **int** id = property.getPropertyId();
58. **if** (p.getPropertyId() == id) {
59. propertyList.set(id, property);
60. **break**;
61. }
62. }
63. }
65. **public** **int** getPropertyId() {
66. **return** propertyList.get(propertyList.size() - 1).getPropertyId();
67. }
69. **public** Property searchPropertyById(**int** propertyId) {
70. **for** (Property p : propertyList) {
71. **if** (p.getPropertyId() == propertyId) {
72. **return** p;
73. }
74. }
75. **return** **null**;
76. }
78. **public** List<ContactPerson> getAllContactPeople() {
79. // map property list to contact person set
80. Set<ContactPerson> contactPersonSet = propertyList.stream().map(property -> property.getContactPerson()).collect(Collectors.toSet());
81. // then set convert to list (it can be one line code but here i separate it to two line because of the readability)
82. List<ContactPerson> contactPersonList = **new** ArrayList<>(contactPersonSet);
83. **return** contactPersonList;
84. }
86. **public** Set<Property> searchPropertyByContactPerson(ContactPerson contactPerson) {
87. // filter the property which contain a contact person object that has same id with argument
88. Set<Property> propertySet = propertyList.stream().filter(property -> property.getContactPerson().getConactPersonId() == contactPerson.getConactPersonId()).collect(Collectors.toSet());
89. **return** propertySet;
90. }
92. **public** List<Property> searchPropertyByBudget(**double** budget) {
93. // filter the property which price are lower or equal to budget
94. **return** propertyList.stream().filter(property -> property.getPrice() <= budget).collect(Collectors.toList());
95. }
97. }

Modified LoanManagedBean.java file highlighting changes with comments

1. /\*
2. \* To change this license header, choose License Headers in Project Properties.
3. \* To change this template file, choose Tools | Templates
4. \* and open the template in the editor.
5. \*/
6. **package** fit5042.tutex.mbeans;
8. **import** java.io.Serializable;
9. **import** javax.ejb.EJB;
10. **import** javax.faces.bean.ManagedBean;
11. **import** javax.enterprise.context.SessionScoped;
13. **import** fit5042.tutex.repository.entities.Loan;
14. **import** fit5042.tutex.calculator.MonthlyPaymentCalculator;
16. /\*\*
17. \*
18. \* @author: originally created by Eddie. The following code has been changed in
19. \* order for students to practice.
20. \*/
21. @ManagedBean(name = "loanManagedBean", eager=**true**)
22. @SessionScoped
23. **public** **class** LoanManagedBean **implements** Serializable {
25. @EJB
26. **private** MonthlyPaymentCalculator calculator;
28. **private** Loan loan;
30. **public** LoanManagedBean() {
31. **this**.loan = **new** Loan();
33. }
35. **public** Loan getLoan() {
36. **return** loan;
37. }
39. **public** **void** setLoan(Loan loan) {
40. **this**.loan = loan;
41. }
43. **public** String calculate() {
44. //You will need to modify the monthlyPayment value and set it as the monthly payment attribute value into the loan instance
45. //Please complete this method starts from here
46. // use the calculator to calculate the monthlyPayment and set it into instance
47. **double** monthlyPayment = calculator.calculate(loan.getPrinciple(), loan.getNumberOfYears(), loan.getInterestRate());
48. loan.setMonthlyPayment(monthlyPayment);
49. **return** "index";
50. }
51. }

Modified CompareProperty.java remote interface

* **package** fit5042.tutex.calculator;
* **import** javax.ejb.Remote;
* **import** fit5042.tutex.repository.entities.Property;
* @Remote
* **public** **interface** CompareProperty {
* **void** addProperty(Property property);
* **void** removeProperty(Property property);
* **int** getBestPerRoom();
* }
* Modified ***ComparePropertySessionBean.java*** implementation
* **package** fit5042.tutex.calculator;
* **import** java.util.ArrayList;
* **import** java.util.List;
* **import** javax.ejb.Stateful;
* **import** fit5042.tutex.repository.entities.Property;
* @Stateful
* **public** **class** ComparePropertySessionBean **implements** CompareProperty {
* // store properties
* **private** List<Property> propertyList;

* // constructor
* **public** ComparePropertySessionBean() {
* propertyList = **new** ArrayList<>();
* }

* @Override
* **public** **void** addProperty(Property property) {
* propertyList.add(property);
* }
* @Override
* **public** **void** removeProperty(Property property) {
* // remove property which have same property id with given argument
* Property existProperty = propertyList.stream().filter(p -> p.getPropertyId() == property.getPropertyId()).findFirst().orElse(**null**);
* propertyList.remove(existProperty);
* }
* @Override
* **public** **int** getBestPerRoom() {
* // fine the zoom with minimum value of (price / bad room)
* Property cheapestProperty = propertyList.stream().min((p1,p2)-> (p1.getPrice() / p1.getNumberOfBedrooms()) > (p2.getPrice() / p2.getNumberOfBedrooms()) ? 1 : -1).orElse(**null**);
* **return** cheapestProperty == **null** ? 0 : cheapestProperty.getPropertyId();
* }
* }

Modified index.xhtml file

* **<?xml** version='1.0' encoding='UTF-8' **?>**
* <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"**>**
* **<html** xmlns="http://www.w3.org/1999/xhtml"
* xmlns:f="http://java.sun.com/jsf/core"
* xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
* xmlns:h="http://xmlns.jcp.org/jsf/html"
* xmlns:p="http://xmlns.jcp.org/jsf/passthrough"**>**
* **<h:head>**
* **<h:outputStylesheet** library="css" name="bootstrap.min.css"**></h:outputStylesheet>**
* **<title>**#{titleController.pageTitle}**</title>**
* **</h:head>**
* **<h:body>**
* **<div** class="container"**>**
* **<div** class="navbar navbar-inverse"**>**
* **<div** class="navbar-brand"**>**#{titleController.pageTitle} -
* Property List**</div>**
* **</div>**
* **<h:form** id="MyForm"**>**
* **<table** class="table table-bordered table-striped"**>**
* <!-- For each property in properties ArrayList... -->
* **<tr>**
* **<td><h:outputLabel** id="Id" value="Id" **/></td>**
* **<td><h:outputLabel** id="Price" value="Price" **/></td>**
* **<td><h:outputLabel** id="Address" value="Address" **/></td>**
* **</tr>**
* **<ui:repeat** id="Myrepeat" value="#{propertyApplication.properties}"
* var="property" varStatus="status"**>**
* **<tr**
* onclick="document.getElementById('MyForm:principle').value = document.getElementById('MyForm:Myrepeat:#{status.index}:outPrice').textContent;
* document.getElementById('MyForm:selectedId').value = document.getElementById('MyForm:Myrepeat:#{status.index}:MainPropertyID').textContent;"**>**
* **<td><h:outputText**
* value="#{propertyApplication.properties.get(status.index).propertyId}"
* id="MainPropertyID" **/></td>**
* **<td><h:outputText** id="outPrice"
* value="#{propertyApplication.properties.get(status.index).price}" **/>**
* **</td>**
* **<td><h:outputText** id="outAddress"
* value="#{propertyApplication.properties.get(status.index).address}"**></h:outputText>**
* **</td>**
* **<td><h:link** value="View | " outcome="faces/detail.xhtml"**>**
* <!-- pass the parameter to next page, the param name is propertyID, and the value is index + 1.
* You can get the value from next page using the indexController --**>**
* **<f:param** name="propertyID" value="#{status.index + 1}" **/>**
* **</h:link>** **<h:link** value="Edit | " outcome="faces/edit.xhtml"**>**
* <!-- pass the parameter to next page, the param name is propertyID, and the value is index + 1.
* You can get the value from next page using the indexController --**>**
* **<f:param** name="propertyID" value="#{status.index + 1}" **/>**
* **</h:link>** **<h:commandLink** value="Delete"
* action="#{removeProperty.removeProperty(propertyApplication.properties.get(status.index).propertyId)}"
* disabled="#{propertyApplication.properties.get(status.index).propertyId == null}"**>**
* **</h:commandLink>** **<h:outputText** value=" " **/></td>**
* **</tr>**
* **</ui:repeat>**
* **</table>**
* **<div>**
* **<h:commandButton** id="add" value="Add" action="add.xhtml"
* class="btn btn-primary" **/>**
* **<h:commandButton** id="viewAll" value="View All"
* action="#{searchProperty.searchAll()}" class="btn btn-primary"**>**
* **<f:param** name="propertyID" value="#{status.index + 1}" **/>**
* **</h:commandButton>**
* **</div>**
* **<div>**
* **<h:inputText** value="#{searchProperty.searchByInt}" id="searchById"
* required="true"
* onchange="if (document.getElementById('MyForm:searchById').value.trim() == '') {document.getElementById('MyForm:searchById').value = 0;} " **/>**
* **<h:commandButton** id="searchId" value="Search by Id"
* action="#{searchProperty.searchPropertyById(searchProperty.searchByInt)}"
* class="btn btn-primary"**>**
* **<f:param** name="propertyID" value="#{status.index + 1}" **/>**
* **</h:commandButton>**
* **<h:inputText** value="#{searchProperty.searchByDouble}"
* id="searchByBudget" required="true"
* onchange="if (document.getElementById('MyForm:searchByBudget').value.trim() == '') {document.getElementById('MyForm:searchByBudget').value = 0.0;} " **/>**
* **<h:commandButton** id="searchBudget" value="Search by Budget"
* action="#{searchProperty.searchPropertyByBudget(searchProperty.searchByDouble)}"
* class="btn btn-primary"**>**
* **<f:param** name="propertyID" value="#{status.index + 1}" **/>**
* **</h:commandButton>**

* **<h:selectOneMenu** id="contact"
* value="#{searchProperty.contactPersonId}"**>**
* **<f:selectItem** itemValue="#{null}" itemLabel="--select--" **/>**
* **<f:selectItems** value="#{propertyManagedBean.allContactPeople}"
* var="c" itemLabel="#{c.name}" itemValue="#{c.conactPersonId}" **/>**
* **</h:selectOneMenu>**
* **<h:commandButton** id="searchByContact" value="Search by Contact"
* action="#{searchProperty.searchPropertyByContactPersonId(searchProperty.contactPersonId)}"
* class="btn btn-primary"**>**
* **<f:param** name="propertyID" value="#{status.index + 1}" **/>**
* **</h:commandButton>**
* **</div>**
* **<div>**
* **<br** **/>**
* **<hr** style="border-color: lightgrey;" **/>**
* **<table>**
* **<tr>**
* **<td** width="500"**><b>**Loan Monthly Payment Calculation**</b></td>**
* **<td><b>**Comparing Properties**</b></td>**
* **</tr>**
* **</table>**
* **</div>**
* **<br** **/>**
* **<br** **/>**
* **<table>**
* **<tr>**
* **<td** width="180"**>**Principle(click on property):**</td>**
* **<td** class="#{!principle.valid ? 'has-error' : 'none'}" width="320"**>**
* **<h:inputText** id="principle"
* value="#{loanManagedBean.loan.principle}" required="true"
* requiredMessage="Please enter the principle!"
* validatorMessage="Principle must be greater than or equal to 1000"
* p:placeholder="1000.00"**>**
* **<f:converter** converterId="javax.faces.Double" **/>**
* **<f:validateLongRange** minimum="1000" **/>**
* **<h:message** for="principle"
* class="principle ? 'label label-warning' : 'none'}" **/>**
* **</h:inputText>**
* **</td>**
* **<td>**Selected property id:**</td>**
* **<td><h:inputText** id="selectedId"
* value="#{propertyManagedBean.selectedId}"**></h:inputText></td>**
* **</tr>**
* **<tr>**
* **</tr>**
* **<tr>**
* **<td>**Interest Rate:**</td>**
* **<td><h:selectOneListbox**
* value="#{loanManagedBean.loan.interestRate}" size="3"**>**
* **<f:selectItem** itemValue="0.01" itemLabel="0.01" **/>**
* **<f:selectItem** itemValue="0.02" itemLabel="0.02" **/>**
* **<f:selectItem** itemValue="0.03" itemLabel="0.03" **/>**
* **<f:selectItem** itemValue="0.04" itemLabel="0.04" **/>**
* **<f:selectItem** itemValue="0.05" itemLabel="0.05" **/>**
* **<f:selectItem** itemValue="0.06" itemLabel="0.06" **/>**
* **<f:selectItem** itemValue="0.07" itemLabel="0.07" **/>**
* **</h:selectOneListbox></td>**
* **<td></td>**
* **<td><h:commandButton** id="addCompare" value="AddCompare"
* action="#{propertyManagedBean.addPropertyID()}" **/>**
* **<h:commandButton** id="removeCompare" value="RemoveCompare"
* action="#{propertyManagedBean.removePropertyID()}" **/></td>**
* **</tr>**
* **<tr>**
* **<td>**Number of Years:**</td>**
* **<td** class="#{!numberOfYears.valid ? 'has-error' : 'none'}"**>**
* **<h:inputText** id="numberOfYears"
* value="#{loanManagedBean.loan.numberOfYears}" required="true"
* requiredMessage="Please enter the number of year!"
* validatorMessage="The number of year must be greater than or equal to 1"
* p:placeholder="1"**>**
* **<f:converter** converterId="javax.faces.Integer" **/>**
* **</h:inputText></td>**
* **<td></td>**
* **<td><h:commandButton** id="bestPerRoom" value="BestPerRoom"
* action="#{propertyManagedBean.bestPerRoom()}" **/>**    **<h:outputLabel**
* id="bestPropertyId" value="#{propertyManagedBean.bestPerRoom}" **/>**
* **</td>**
* **</tr>**
* **<tr>**
* **<td>**Monthly Payment:**</td>**
* **<td** colspan="3"**><h:outputLabel** id="monthlyPayment"
* value="#{loanManagedBean.loan.monthlyPayment}"**>**
* **<f:convertNumber** currencyCode="AUD" type="currency" **/>**
* **</h:outputLabel></td>**
* **</tr>**
* **<tr>**
* **<td** colspan="4"**><h:commandButton** id="calculate"
* value="Calculate" action="#{loanManagedBean.calculate()}" **/></td>**
* **</tr>**
* **</table>**
* **</h:form>**
* **</div>**
* **</h:body>**
* **</html>**

A screenshot of a computer

Description automatically generated

Edit screen

A screenshot of a computer

Description automatically generated

Screen shot for searchPropertyByBudget(double budget) method

A screenshot of a computer

Description automatically generated

Screen shot for getAllContactPeople method and searchPropertyByContactPerson(ContactPerson contactPerson)

A screenshot of a computer

Description automatically generated

Loan Monthly Payment Calculation(error input)

A screenshot of a computer

Description automatically generated Loan Monthly Payment Calculation(correct input)

A screenshot of a computer

Description automatically generated

Comparing Properties(add properties with id 1, 2, 4)

Link to code repository

https://github.com/zake111000/FIT5042

References

https://www.tutorialspoint.com/ejb/ejb\_annotations.html

# Declaration

I declare that this eFolio tasks and the linked code are my individual work. I have not copied from any other student’s work or from any other source except where due acknowledgment is made explicitly in the text and code, nor has any part of this submission been written for me by another person.

Signature of student: Ke Zhang\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_