Coursework

260CT Software Engineering

Name: Haseong Kim

Student ID: 8355421

Contents

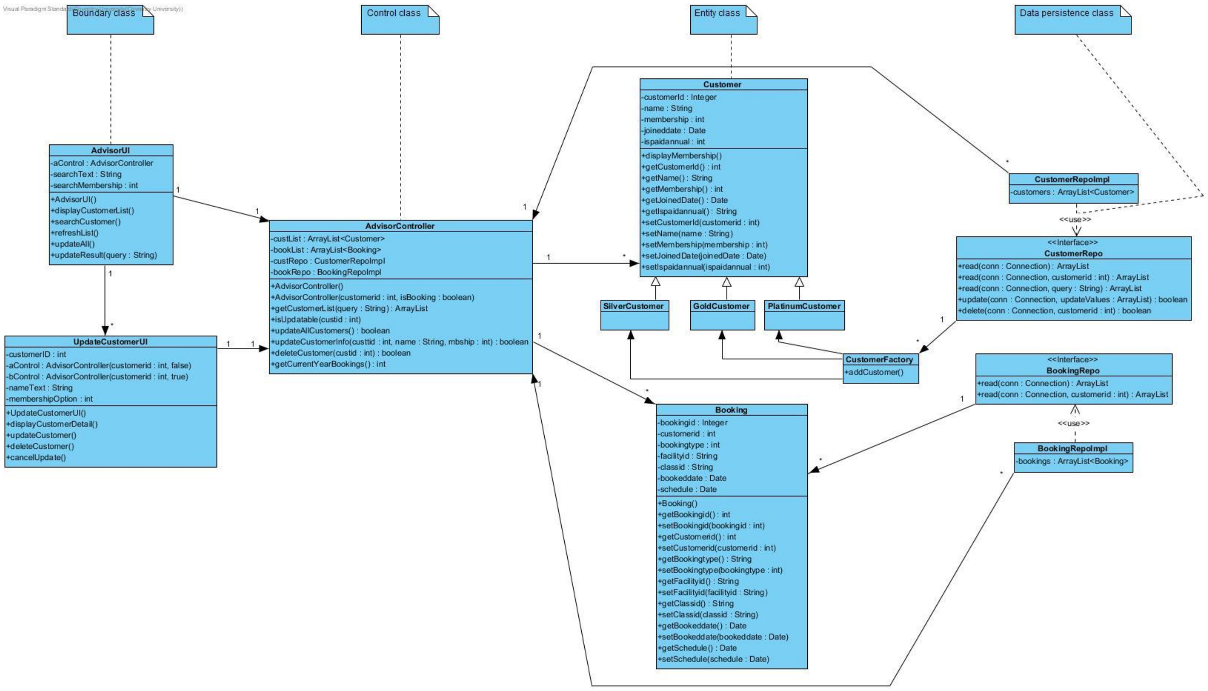
Task 1 ……… 2

Task 2 ……….... 4

References …….…. 18

Task 1

Four-layered architectural design model


Boundary class
Control class
Entity class
Data persistence class
AdvisorUI
-aControl : AdvisorController
-searchText : String
-searchMembership : int
+AdvisorUI()
+displayCustomerList()
+searchCustomer()
+refreshList()
+updateAll()
+updateResult(query : String)
UpdateCustomerUI
-customerID : int
-aControl : AdvisorController(customerid : int, false)
-bControl : AdvisorController(customerid : int, true)
-nameText : String
-membershipOption : int
+UpdateCustomerUI()
+displayCustomerDetail()
+updateCustomer()
+deleteCustomer()
+cancelUpdate()
AdvisorController
-custList : ArrayList<Customer>
-bookList : ArrayList<Booking>
-custRepo : CustomerRepoImpl
-bookRepo : BookingRepoImpl
+AdvisorController()
+AdvisorController(customerid : int, isBooking : boolean)
+getCustomerList(query : String) : ArrayList
+isUpdatable(custid : int)
+updateAllCustomers() : boolean
+updateCustomerInfo(custtid : int, name : String, mbship : int) : boolean
+deleteCustomer(custid : int) : boolean
+getCurrentYearBookings() : int
Customer
-customerId : Integer
-name : String
-membership : int
-joineddate : Date
-ispaidannual : int
+displayMembership()
+getCustomerId() : int
+getName() : String
+getMembership() : int
+getJoinedDate() : Date
+getIspaidannual() : String
+setCustomerId(customerid : int)
+setName(name : String)
+setMembership(membership : int)
+setJoinedDate(joinedDate : Date)
+setIspaidannual(ispaidannual : int)
SilverCustomer
GoldCustomer
PlatinumCustomer
CustomerFactory
+addCustomer()
Booking
-bookingid : Integer
-customerid : int
-bookingtype : int
-facilityid : String
-classid : String
-bookeddate : Date
-schedule : Date
+Booking()
+getBookingid() : int
+setBookingid(bookingid : int)
+getCustomerid() : int
+setCustomerid(customerid : int)
+getBookingtype() : String
+setBookingtype(bookingtype : int)
+getFacilityid() : String
+setFacilityid(facilityid : String)
+getClassid() : String
+setClassid(classid : String)
+getBookeddate() : Date
+setBookeddate(bookeddate : Date)
+getSchedule() : Date
+setSchedule(schedule : Date)
CustomerRepoImpl
-customers : ArrayList<Customer>
<<use>>
<<Interface>>
CustomerRepo
+read(conn : Connection) : ArrayList
+read(conn : Connection, customerid : int) : ArrayList
+read(conn : Connection, query : String) : ArrayList
+update(conn : Connection, updateValues : ArrayList) : boolean
+delete(conn : Connection, customerid : int) : boolean
<<Interface>>
BookingRepo
+read(conn : Connection) : ArrayList
+read(conn : Connection, customerid : int) : ArrayList
<<use>>
BookingRepoImpl
-bookings : ArrayList<Booking>

The chosen functionality is membership management. For the details of the four layers, there are user interface layer, application and business logic layer, domain data layer and data persistence layer. The user interface layer that has boundary classes to graphically handle input and output with the clients has two user interfaces; AdvisorUI, and UpdateCustomerUI. In the AdvisorUI, customers are listed with search form in order to select specific customer to update the information. When selected the specific customer, the UpdateCustomerUI is opened to check the customer details with booking lists and change name and membership. The application and business logic layer that is control classes to coordinate the application logic for each use case *(Hedley, 2019)* has AdvisorController. The AdvisorController provides getting customer and booking lists, updating customer details and deleting customer. The domain data layer that has data objects contained in data classes *(Hedley, 2019)* and stores data from the data persistence class as an array list has two classes; Customer and Booking. The two classes allow to store data from the database table. The data persistence layer that has data persistence classes to access the underlying database and handle data from the database has two classes with two extended classes; CustomerRepo with CustomerRepoImpl and BookingRepo with BookingRepoImpl. They are similar roles to AdvisorController and two entities classes but they have actual parts to read, update and delete data from the database after connected to the database.

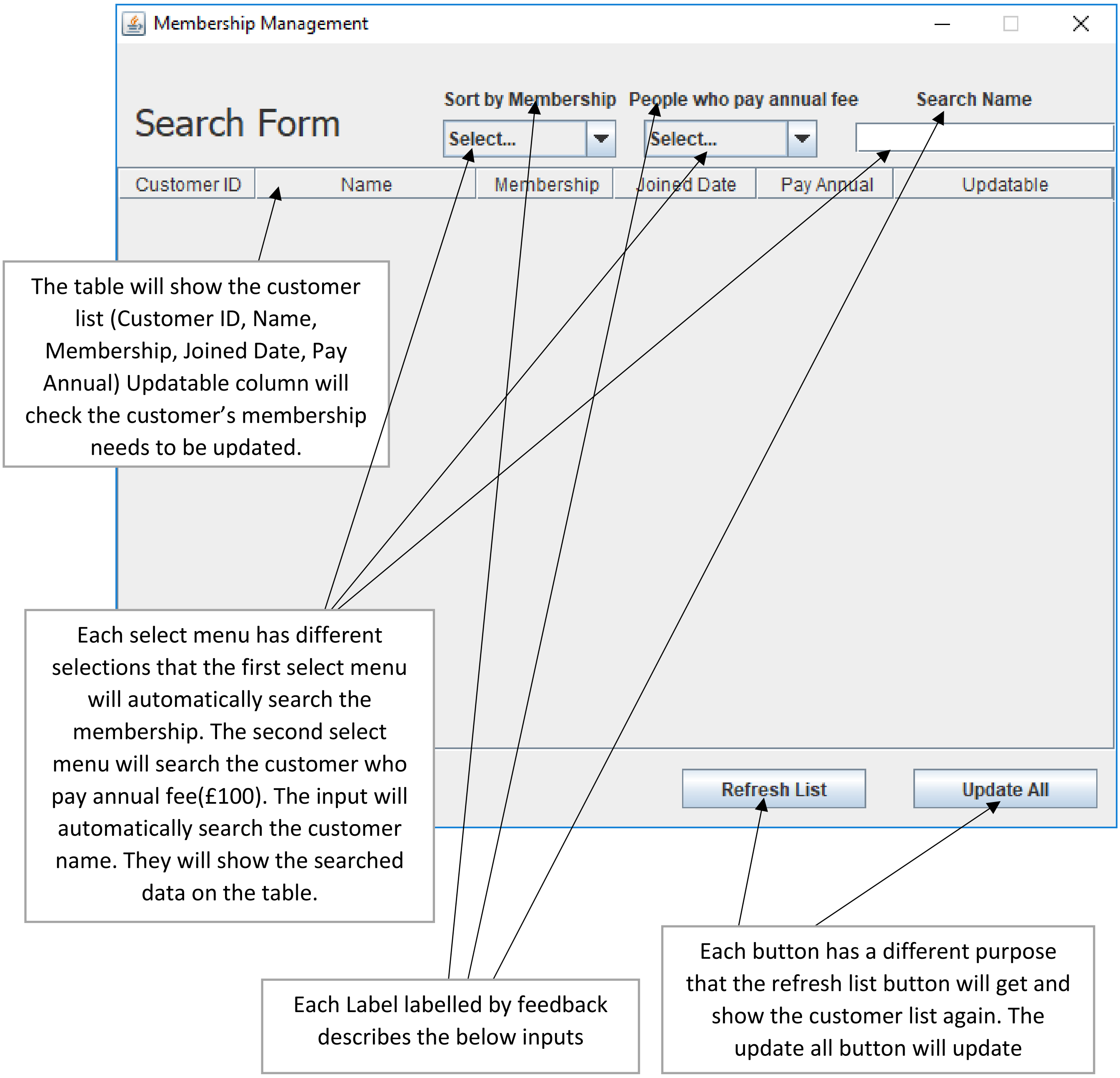
For the GRASP pattern, Façade controller pattern is applied; one class represents the "system" or business and controls everything in that system. *(Hedley, 2019)* The reason is that the system is a complex system that would have an external communication layer over the existing system that is not compatible with the system. The control class has responsibility to show customer list, details and booking list, to update the customer information that the client sends the input form, included name and membership, and to delete the customer data.

For the GoF pattern, Factory method pattern is applied to the entity class; instantiates the appropriate subclass based on information supplied by the client or extracted from the current state. *(Hedley, 2019)* Firstly, the factory class works to add Customer details by using Customer Class. When displayMembership function is called, then show the membership as a string because the membership is stored as an integer at the database. By separating to display membership as a string and an integer, the useless codes are deleted, for example, distinguish the membership as an integer and show it as a string, using switch statement. The booking entity class can't be linked to GoldCustomer because the class is necessary to update a membership from silver to gold and if the class is linked only to GoldCustomer, then silver can't be upgraded or platinum can't be downgraded to gold. Also, PlatinumCustomer can't take ispaidannual column from the database because if it takes the column, then silver or gold can't be upgraded to platinum. The reason why they are happened is that if the class is linked and the PlatinumCustomer takes the column, only the customers who got gold or platinum membership can be seen the booking lists or the column but other memberships can't be seen them.

Task 2

First prototype user interface design

AdvisorUI.java



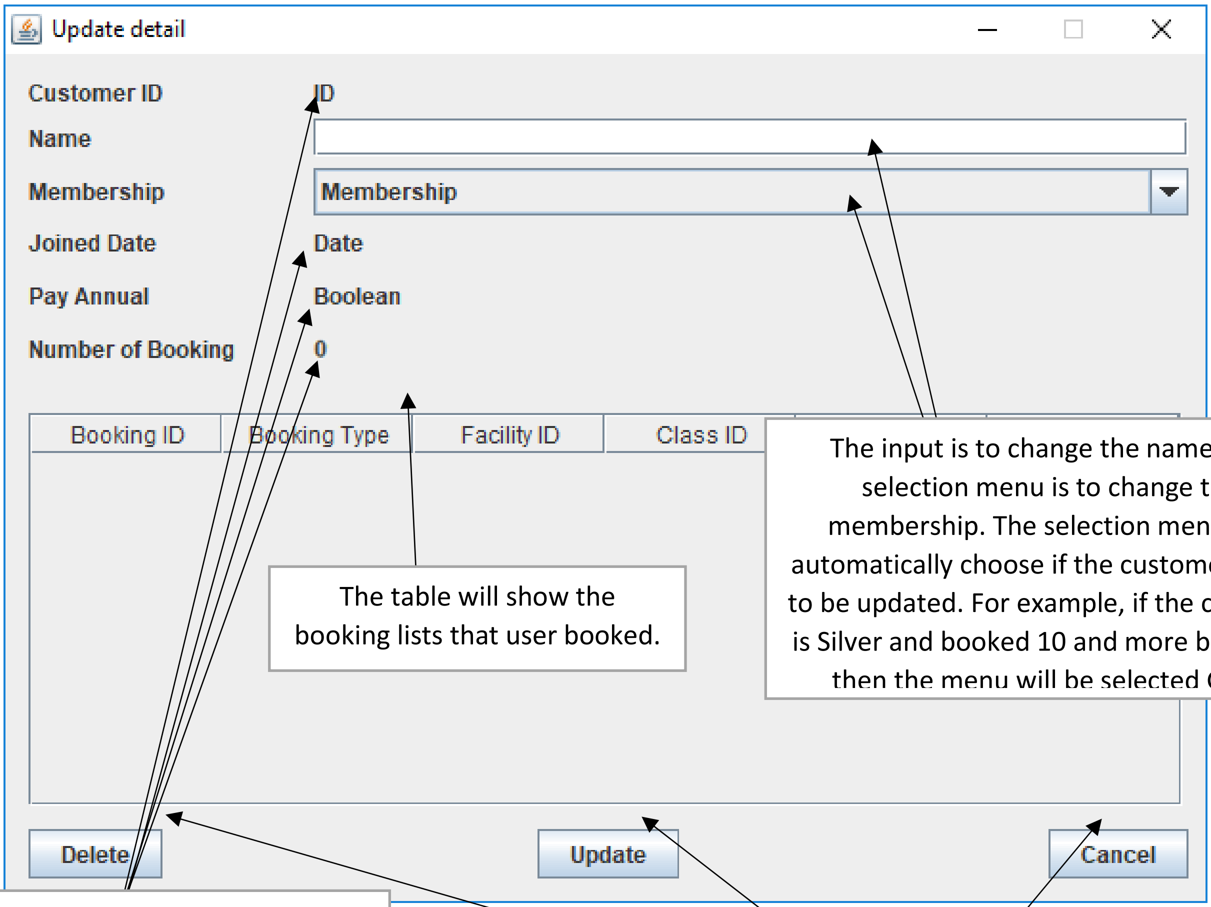
The table will show the customer list (Customer ID, Name, Membership, Joined Date, Pay Annual) Updatable column will check the customer's membership needs to be updated.

Each select menu has different selections that the first select menu will automatically search the membership. The second select menu will search the customer who pay annual fee(£100). The input will automatically search the customer name. They will show the searched data on the table.

Each Label labelled by feedback describes the below inputs

Each button has a different purpose that the refresh list button will get and show the customer list again. The update all button will update

UpdateCustomerUI.java



The input is to change the name. The selection menu is to change the membership. The selection menu will automatically choose if the customer needs to be updated. For example, if the customer is Silver and booked 10 and more bookings, then the menu will be selected Gold.

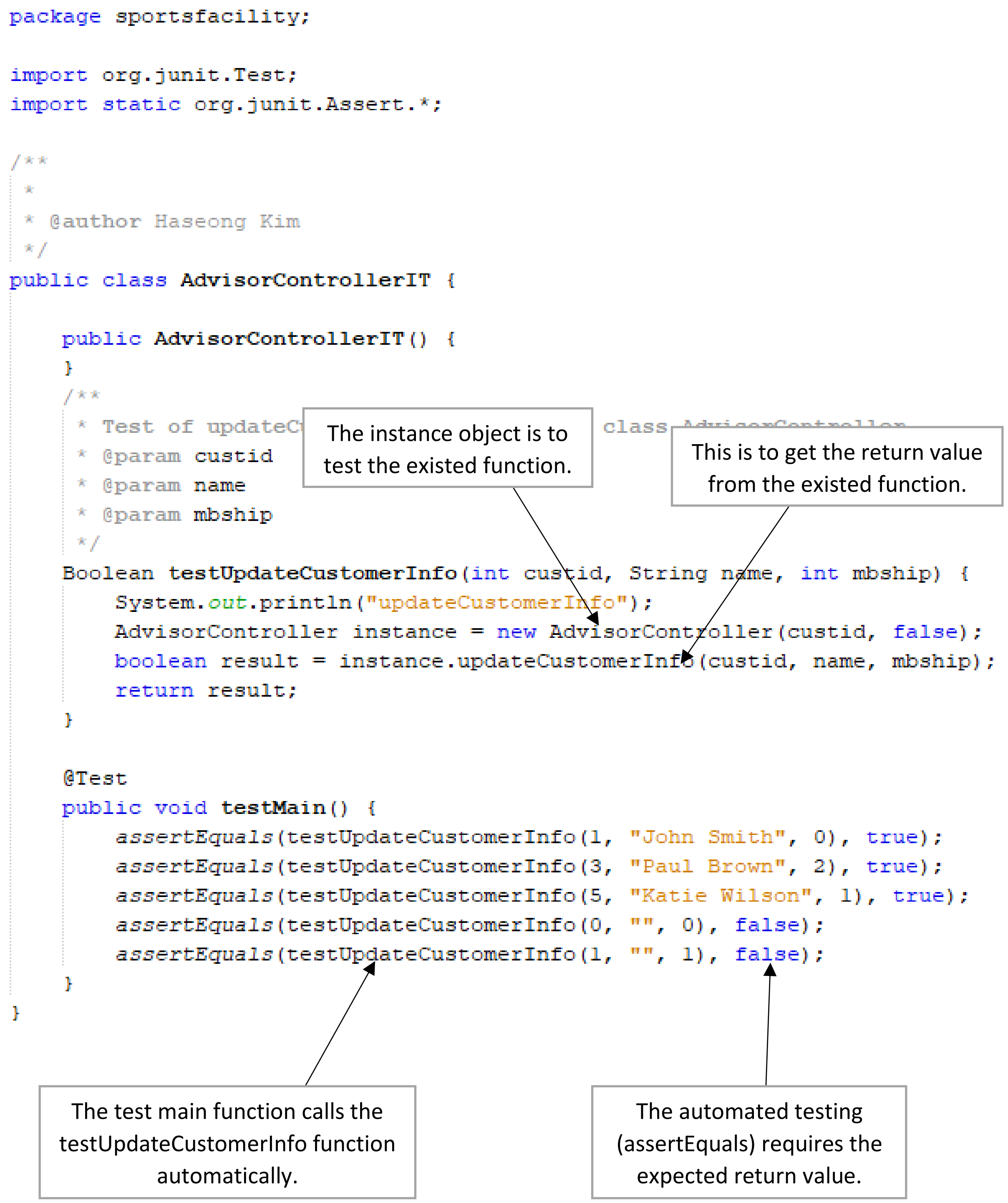
The table will show the booking lists that user booked.

Each Label will be replaced by the customer information selected from Advisor table. ID will show the customer ID, Date will show the date user joined, Boolean will show true or false if the customer pays a fee of £100 a year, and zero will be replaced how many bookings the customer booked this year with total bookings.

The first design before getting feedback was Update, Delete, Cancel. After getting feedback, Update and Delete buttons were changed.

Each button has a different purpose that the delete button will remove the selected customer information from the database. The update button will update the customer information with the entered Name and the selected Membership. The cancel button will just close the form.

Automated unit testing



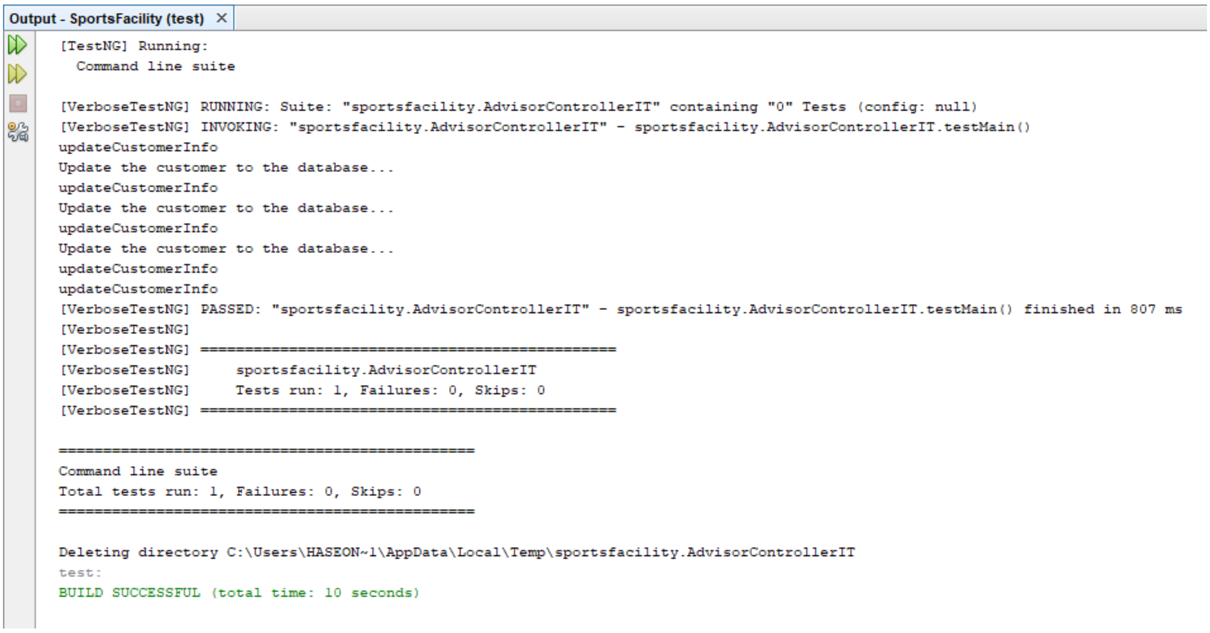
```
package sportsfacility;

import org.junit.Test;
import static org.junit.Assert.*;

/**
 *
 * @author Haseong Kim
 */
public class AdvisorControllerIT {

    public AdvisorControllerIT() {
    }
    /**
     * Test of updateC                    class AdvisorController
     * @param custid
     * @param name
     * @param mbship
     */
    Boolean testUpdateCustomerInfo(int custid, String name, int mbship) {
        System.out.println("updateCustomerInfo");
        AdvisorController instance = new AdvisorController(custid, false);
        boolean result = instance.updateCustomerInfo(custid, name, mbship);
        return result;
    }

    @Test
    public void testMain() {
        assertEquals(testUpdateCustomerInfo(1, "John Smith", 0), true);
        assertEquals(testUpdateCustomerInfo(3, "Paul Brown", 2), true);
        assertEquals(testUpdateCustomerInfo(5, "Katie Wilson", 1), true);
        assertEquals(testUpdateCustomerInfo(0, "", 0), false);
        assertEquals(testUpdateCustomerInfo(1, "", 1), false);
    }
}
```

The instance object is to test the existed function.

This is to get the return value from the existed function.

The test main function calls the testUpdateCustomerInfo function automatically.

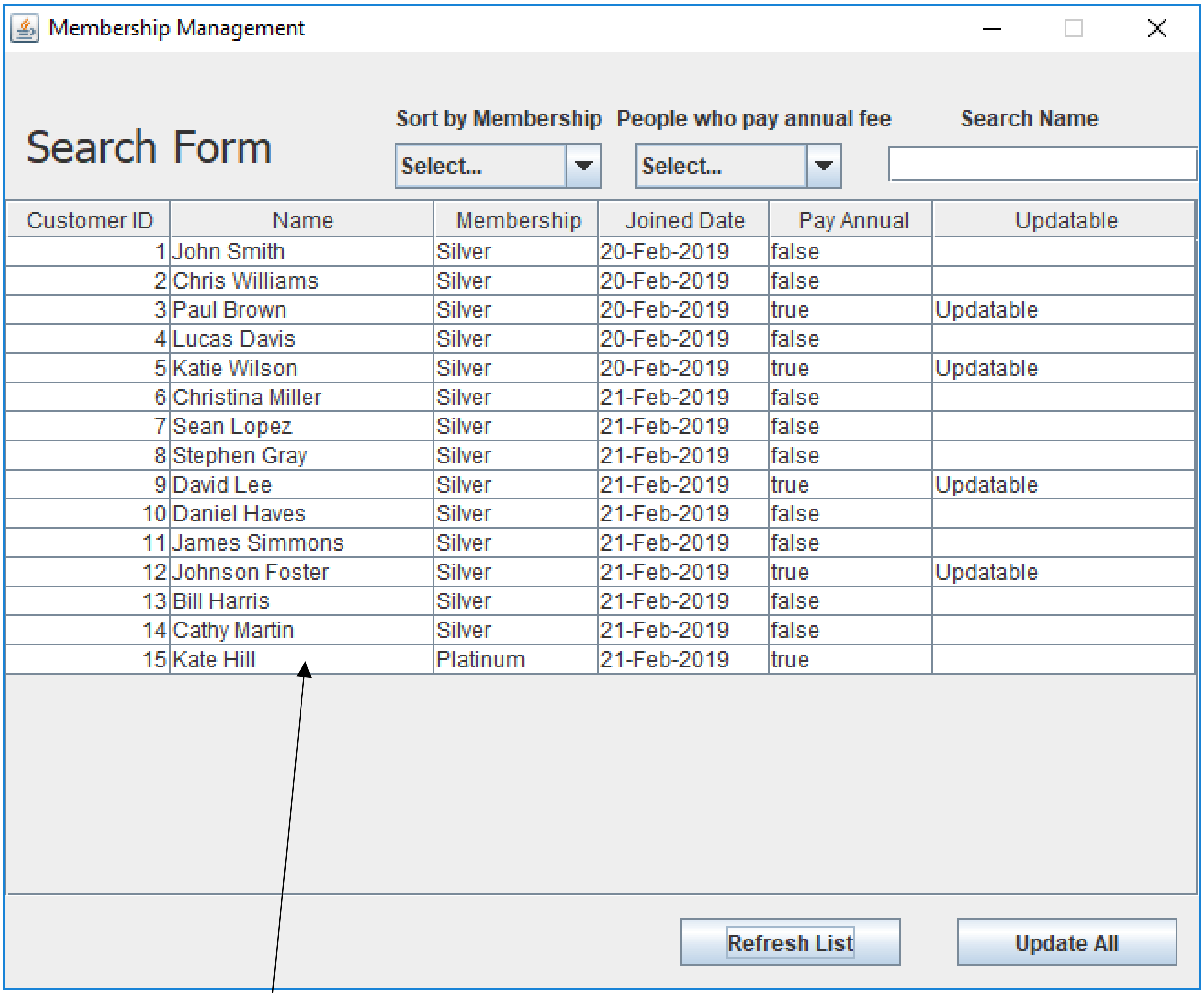
The automated testing (assertEquals) requires the expected return value.



```
 [TestNG] Running:
   Command line suite

 [VerboseTestNG] RUNNING: Suite: "sportsfacility.AdvisorControllerIT" containing "0" Tests (config: null)
 [VerboseTestNG] INVOKING: "sportsfacility.AdvisorControllerIT" - sportsfacility.AdvisorControllerIT.testMain()
updateCustomerInfo
Update the customer to the database...
updateCustomerInfo
Update the customer to the database...
updateCustomerInfo
Update the customer to the database...
updateCustomerInfo
updateCustomerInfo
 [VerboseTestNG] PASSED: "sportsfacility.AdvisorControllerIT" - sportsfacility.AdvisorControllerIT.testMain() finished in 807 ms
 [VerboseTestNG]
 [VerboseTestNG] ===============================================
 [VerboseTestNG]     sportsfacility.AdvisorControllerIT
 [VerboseTestNG]     Tests run: 1, Failures: 0, Skips: 0
 [VerboseTestNG] ===============================================

===============================================
Command line suite
Total tests run: 1, Failures: 0, Skips: 0
===============================================

Deleting directory C:\Users\HASEON~1\AppData\Local\Temp\sportsfacility.AdvisorControllerIT
test:
BUILD SUCCESSFUL (total time: 10 seconds)
```

The key method of the functionality is updating customer details to change membership or name. The testing is constructed with the automated unit testing, called "JUnit", in Netbeans. The testUpdateCustomerInfo takes three parameters; CustomerID, Name, Membership. The existed function called updateCustomerInfo also takes the three parameters and return Boolean that will be false when there is nothing to be changed, the customer id is 0 or the input name is empty and true when there is some changes. The first test at the 38th line in the code is that calling the test function returns true because the name who the customer id is 1 was "Smith John". The second test at the 39th line in the code is that calling the test function returns true because the membership with that the customer id is 3 was 0. The third test at the 40th line in the code is that calling the test function returns true because the membership with that the customer id is 5 was 0. The fourth test at the 41th line in the code is that calling the test function returns false because the input customer id is 0 and the input name is empty. The third test at the 42th line in the code is that calling the test function returns false because the input name is empty and the customer who the id is 1 does not have 10 or more bookings (Gold Membership).

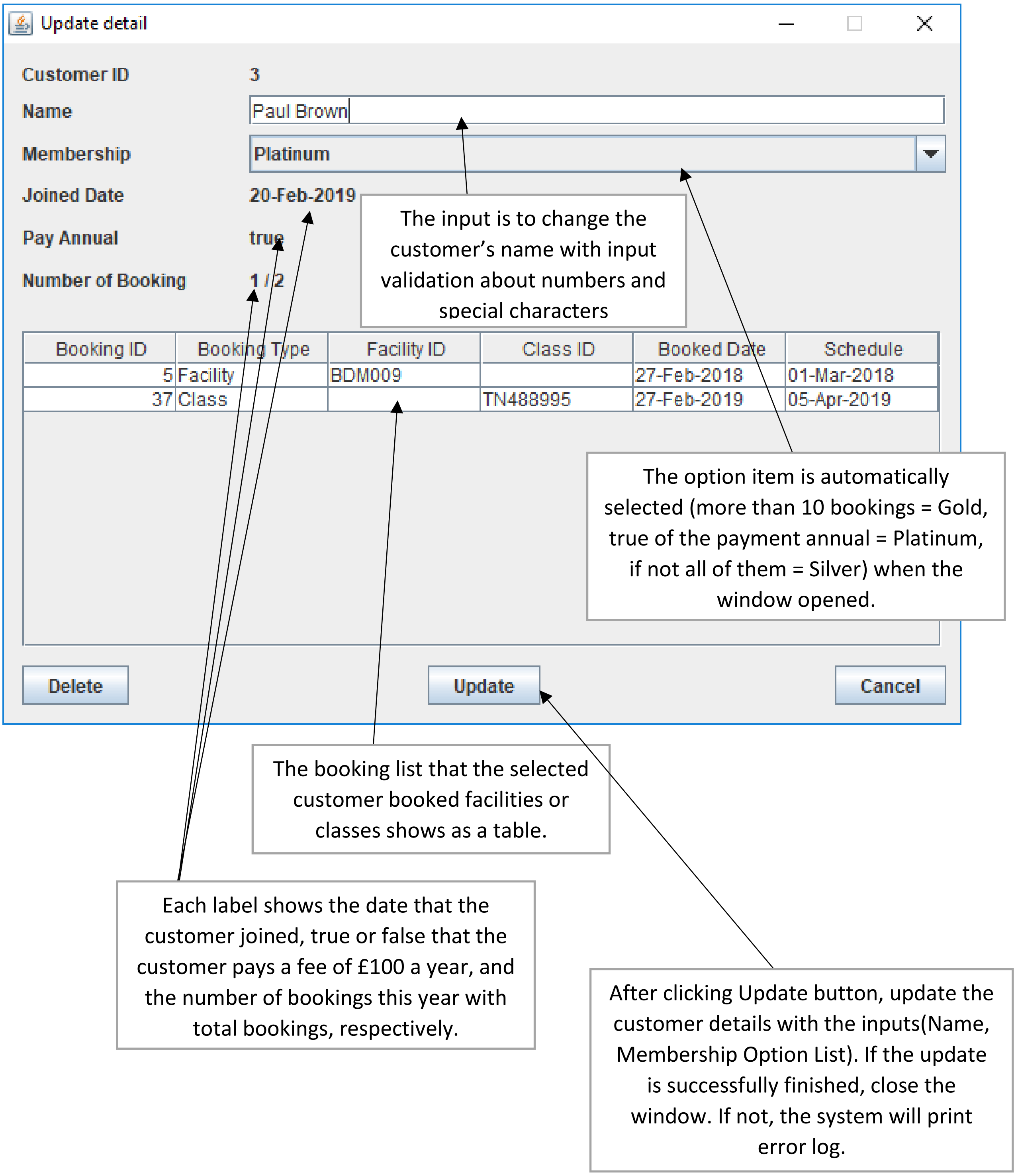
Final version prototype system



Customer ID	Name	Membership	Joined Date	Pay Annual	Updatable
1	John Smith	Silver	20-Feb-2019	false	
2	Chris Williams	Silver	20-Feb-2019	false	
3	Paul Brown	Silver	20-Feb-2019	true	Updatable
4	Lucas Davis	Silver	20-Feb-2019	false	
5	Katie Wilson	Silver	20-Feb-2019	true	Updatable
6	Christina Miller	Silver	21-Feb-2019	false	
7	Sean Lopez	Silver	21-Feb-2019	false	
8	Stephen Gray	Silver	21-Feb-2019	false	
9	David Lee	Silver	21-Feb-2019	true	Updatable
10	Daniel Haves	Silver	21-Feb-2019	false	
11	James Simmons	Silver	21-Feb-2019	false	
12	Johnson Foster	Silver	21-Feb-2019	true	Updatable
13	Bill Harris	Silver	21-Feb-2019	false	
14	Cathy Martin	Silver	21-Feb-2019	false	
15	Kate Hill	Platinum	21-Feb-2019	true	

The customer list that is not updated shows as a table. The Updatable column checks whether the customer needs to be updated to get a higher or lower membership than the current membership.

If the manager double-clicks any row in the table, then opens the window that can change some values of the customer details.



Customer ID	3
Name	Paul Brown
Membership	Platinum
Joined Date	20-Feb-2019
Pay Annual	true
Number of Booking	1 / 2

Booking ID	Booking Type	Facility ID	Class ID	Booked Date	Schedule
5	Facility	BDM009		27-Feb-2018	01-Mar-2018
37	Class		TN488995	27-Feb-2019	05-Apr-2019

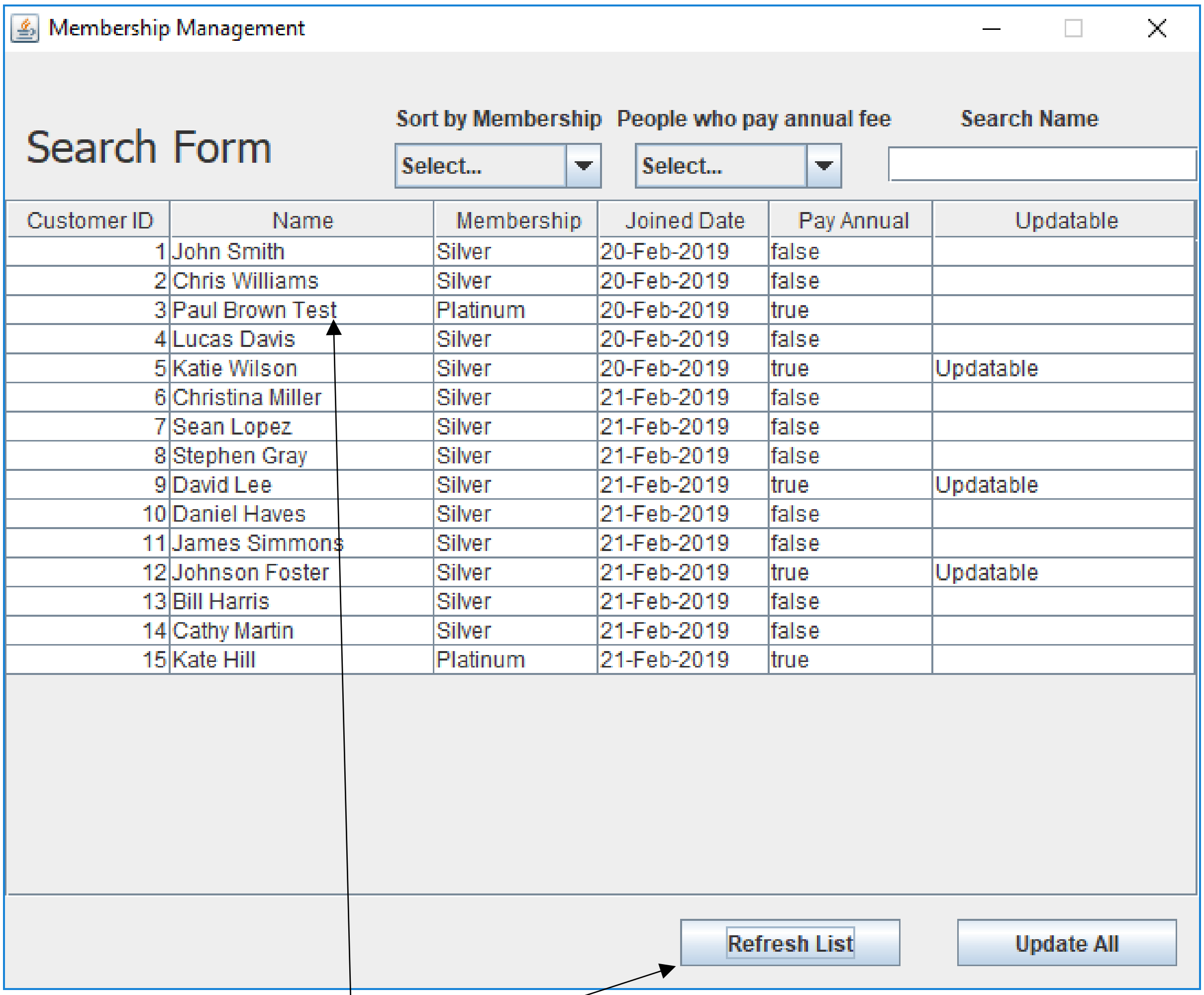
The input is to change the customer's name with input validation about numbers and special characters

The option item is automatically selected (more than 10 bookings = Gold, true of the payment annual = Platinum, if not all of them = Silver) when the window opened.

The booking list that the selected customer booked facilities or classes shows as a table.

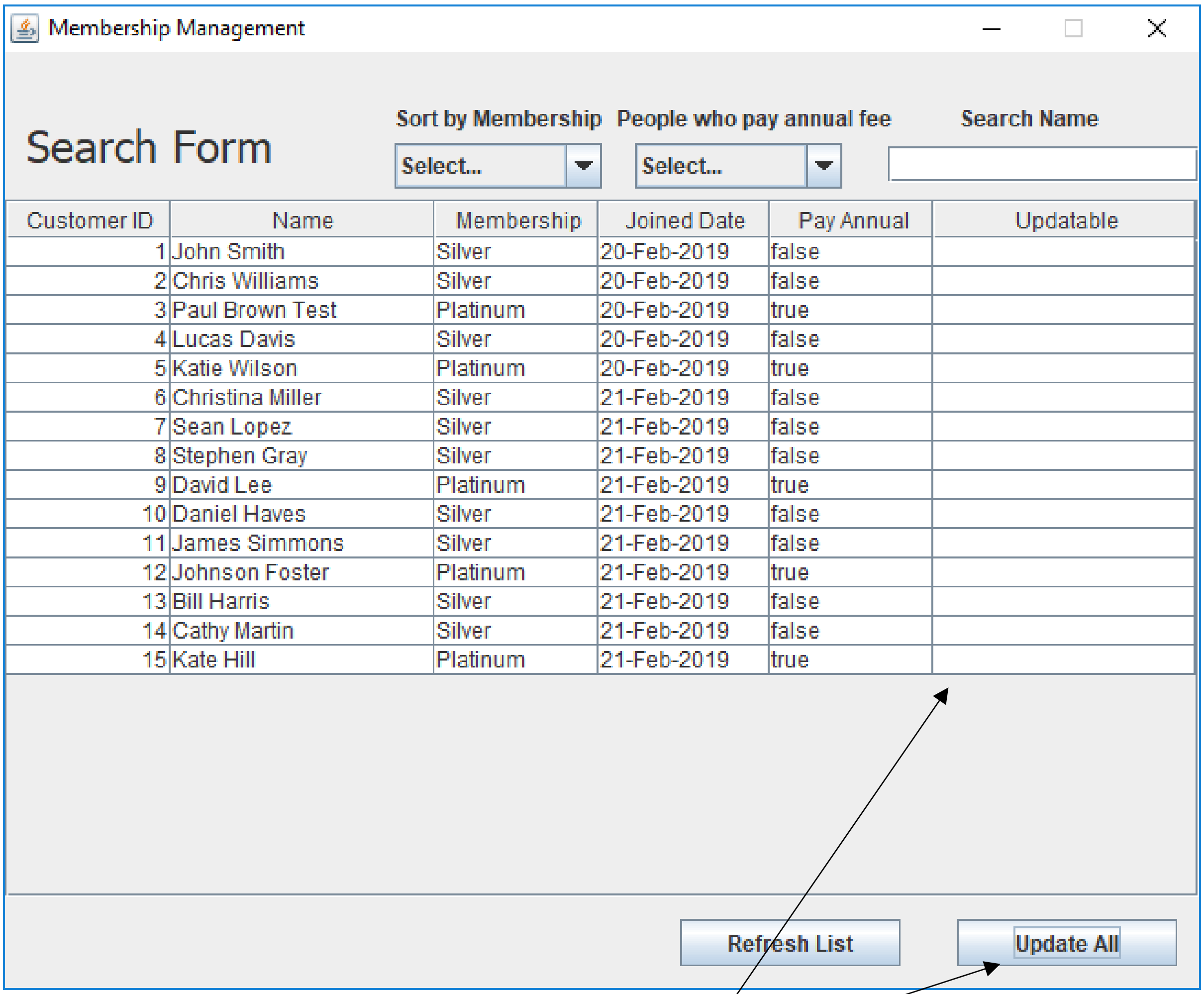
Each label shows the date that the customer joined, true or false that the customer pays a fee of £100 a year, and the number of bookings this year with total bookings, respectively.

After clicking Update button, update the customer details with the inputs(Name, Membership Option List). If the update is successfully finished, close the window. If not, the system will print error log.



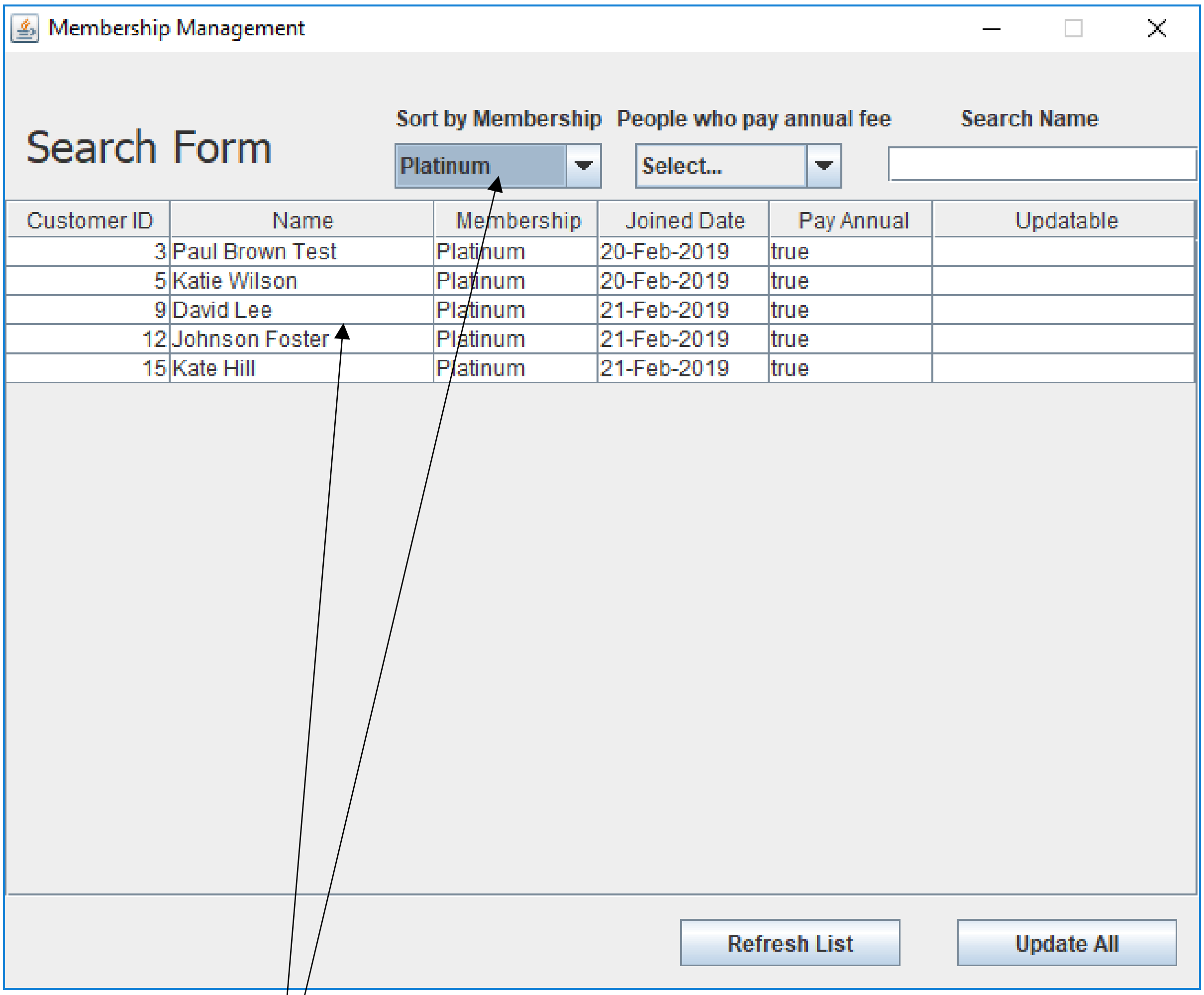
Customer ID	Name	Membership	Joined Date	Pay Annual	Updatable
1	John Smith	Silver	20-Feb-2019	false	
2	Chris Williams	Silver	20-Feb-2019	false	
3	Paul Brown Test	Platinum	20-Feb-2019	true	
4	Lucas Davis	Silver	20-Feb-2019	false	
5	Katie Wilson	Silver	20-Feb-2019	true	Updatable
6	Christina Miller	Silver	21-Feb-2019	false	
7	Sean Lopez	Silver	21-Feb-2019	false	
8	Stephen Gray	Silver	21-Feb-2019	false	
9	David Lee	Silver	21-Feb-2019	true	Updatable
10	Daniel Haves	Silver	21-Feb-2019	false	
11	James Simmons	Silver	21-Feb-2019	false	
12	Johnson Foster	Silver	21-Feb-2019	true	Updatable
13	Bill Harris	Silver	21-Feb-2019	false	
14	Cathy Martin	Silver	21-Feb-2019	false	
15	Kate Hill	Platinum	21-Feb-2019	true	

After finishing the update previous, click Refresh List button. The list reshows updated list as shown the updated customer previously.



Customer ID	Name	Membership	Joined Date	Pay Annual	Updatable
1	John Smith	Silver	20-Feb-2019	false	
2	Chris Williams	Silver	20-Feb-2019	false	
3	Paul Brown Test	Platinum	20-Feb-2019	true	
4	Lucas Davis	Silver	20-Feb-2019	false	
5	Katie Wilson	Platinum	20-Feb-2019	true	
6	Christina Miller	Silver	21-Feb-2019	false	
7	Sean Lopez	Silver	21-Feb-2019	false	
8	Stephen Gray	Silver	21-Feb-2019	false	
9	David Lee	Platinum	21-Feb-2019	true	
10	Daniel Haves	Silver	21-Feb-2019	false	
11	James Simmons	Silver	21-Feb-2019	false	
12	Johnson Foster	Platinum	21-Feb-2019	true	
13	Bill Harris	Silver	21-Feb-2019	false	
14	Cathy Martin	Silver	21-Feb-2019	false	
15	Kate Hill	Platinum	21-Feb-2019	true	

After clicking Update All button, the customers who need to be updated for getting a higher or lower membership. For example, if the customer is Gold, has 19 bookings in total and has 9 bookings this year, then the customer gets Silver membership.



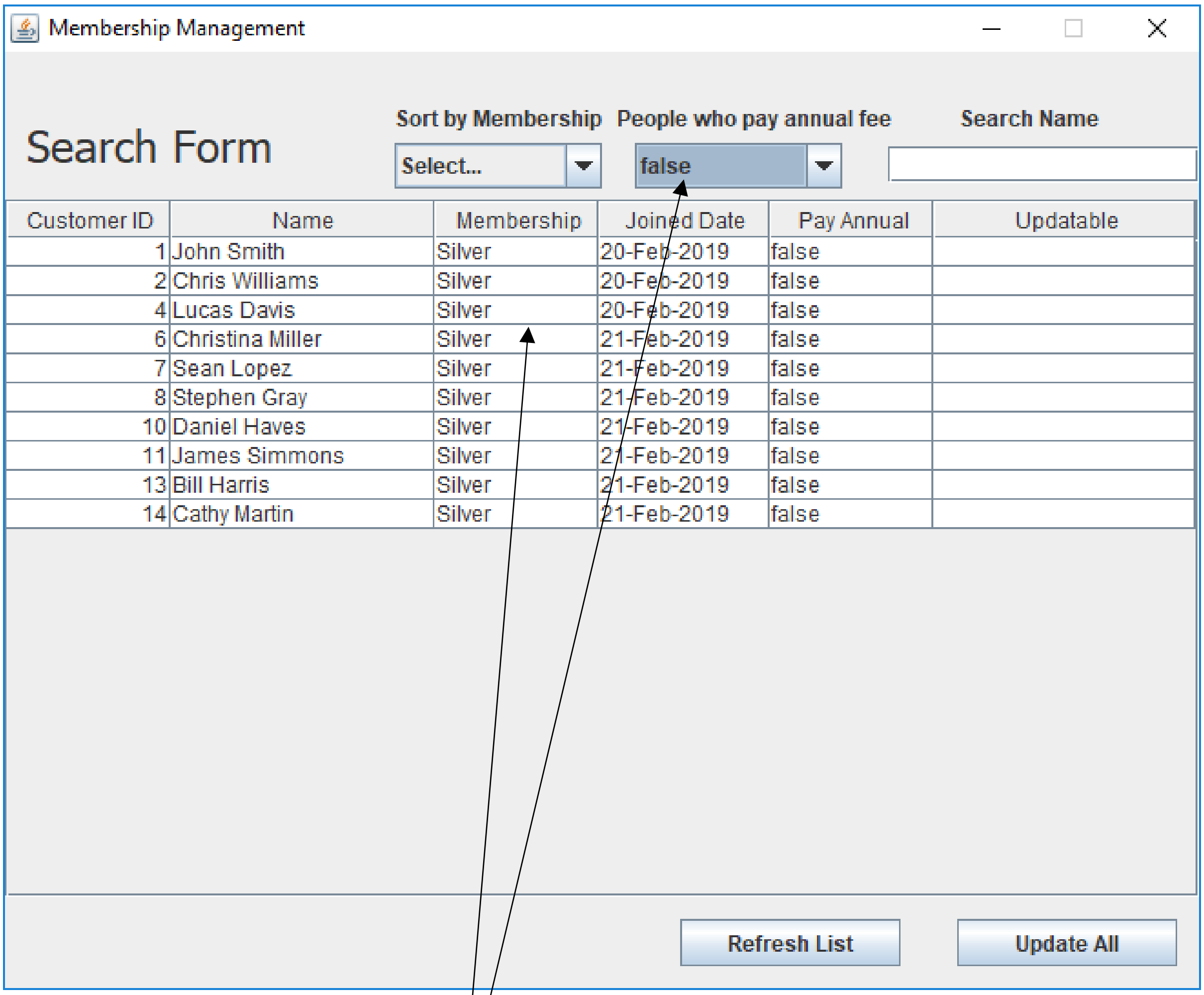
Sort by Membership: Platinum

People who pay annual fee: Select...

Search Name:

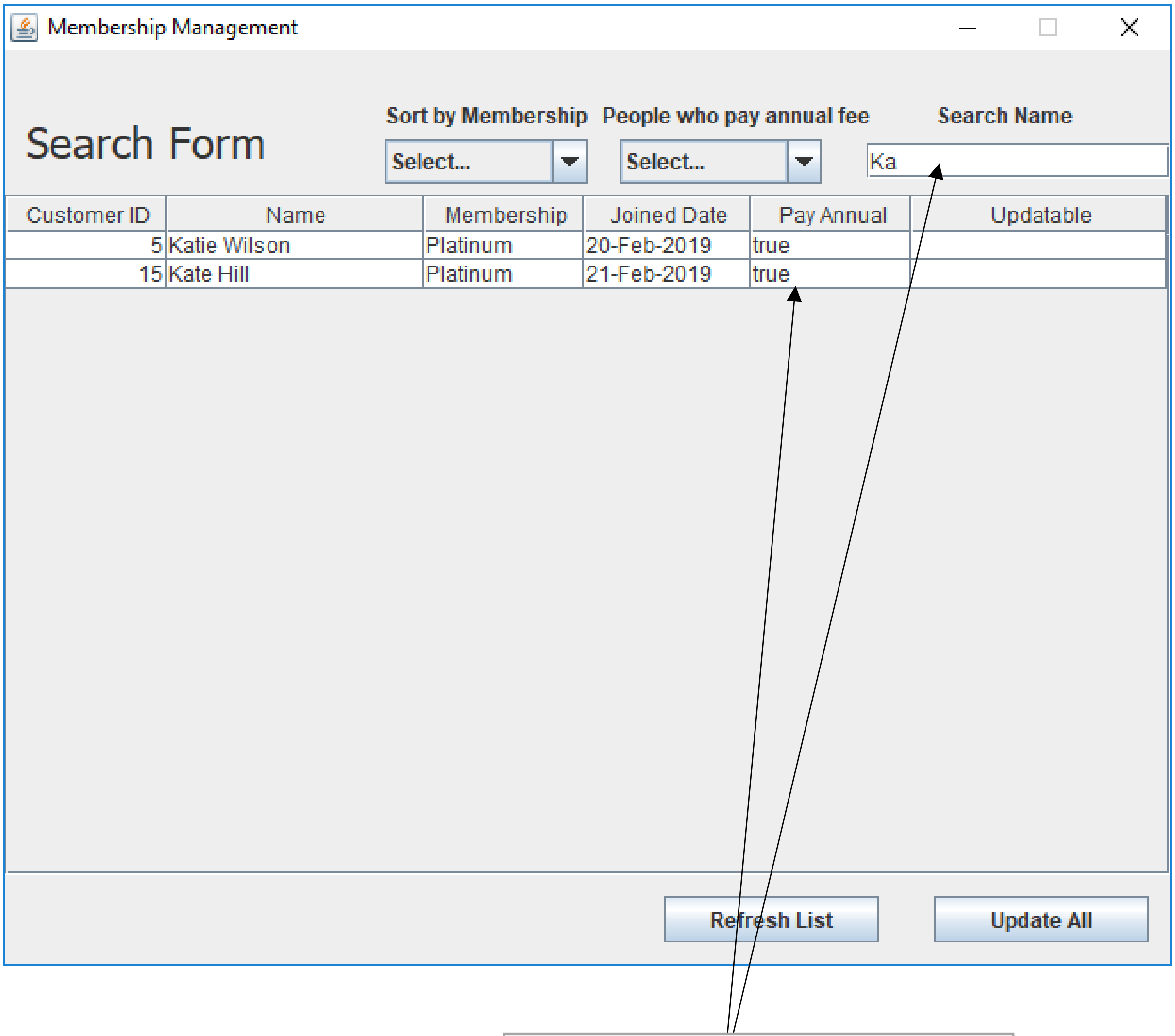
Customer ID	Name	Membership	Joined Date	Pay Annual	Updatable
3	Paul Brown Test	Platinum	20-Feb-2019	true	
5	Katie Wilson	Platinum	20-Feb-2019	true	
9	David Lee	Platinum	21-Feb-2019	true	
12	Johnson Foster	Platinum	21-Feb-2019	true	
15	Kate Hill	Platinum	21-Feb-2019	true	

If select any membership option, then automatically search the selected membership from the database and show the result list to the table.



Customer ID	Name	Membership	Joined Date	Pay Annual	Updatable
1	John Smith	Silver	20-Feb-2019	false	
2	Chris Williams	Silver	20-Feb-2019	false	
4	Lucas Davis	Silver	20-Feb-2019	false	
6	Christina Miller	Silver	21-Feb-2019	false	
7	Sean Lopez	Silver	21-Feb-2019	false	
8	Stephen Gray	Silver	21-Feb-2019	false	
10	Daniel Haves	Silver	21-Feb-2019	false	
11	James Simmons	Silver	21-Feb-2019	false	
13	Bill Harris	Silver	21-Feb-2019	false	
14	Cathy Martin	Silver	21-Feb-2019	false	

If select true or false option, then automatically search customers who pay a fee of £100 a year from the database and show the result list to the table.



Search Form

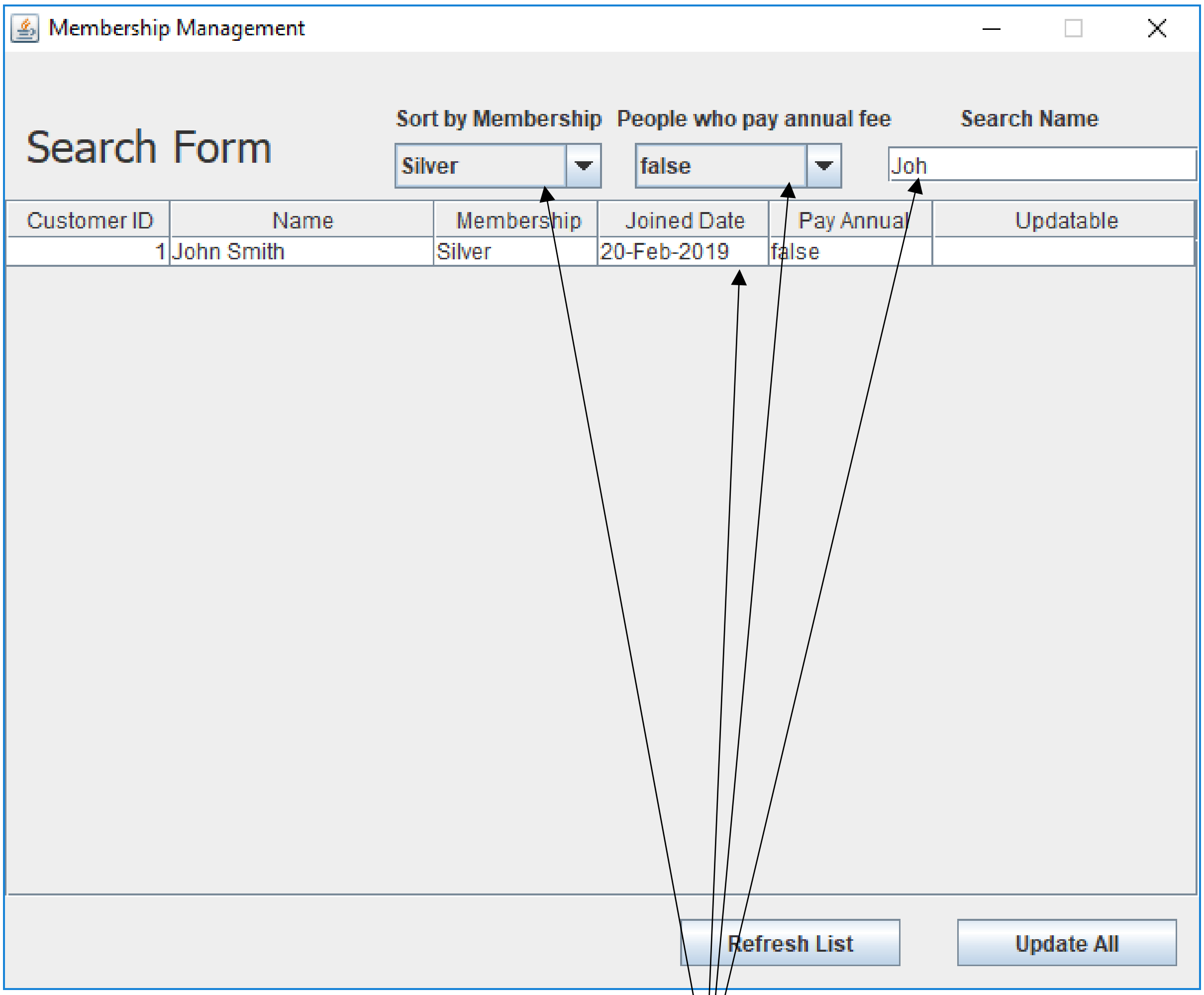
Sort by Membership: Select...

People who pay annual fee: Select...

Search Name: Ka

Customer ID	Name	Membership	Joined Date	Pay Annual	Updatable
5	Katie Wilson	Platinum	20-Feb-2019	true	
15	Kate Hill	Platinum	21-Feb-2019	true	

If type any text in the input, then automatically search customers' name with the typed letters from the database and show the result list to the



Search Form

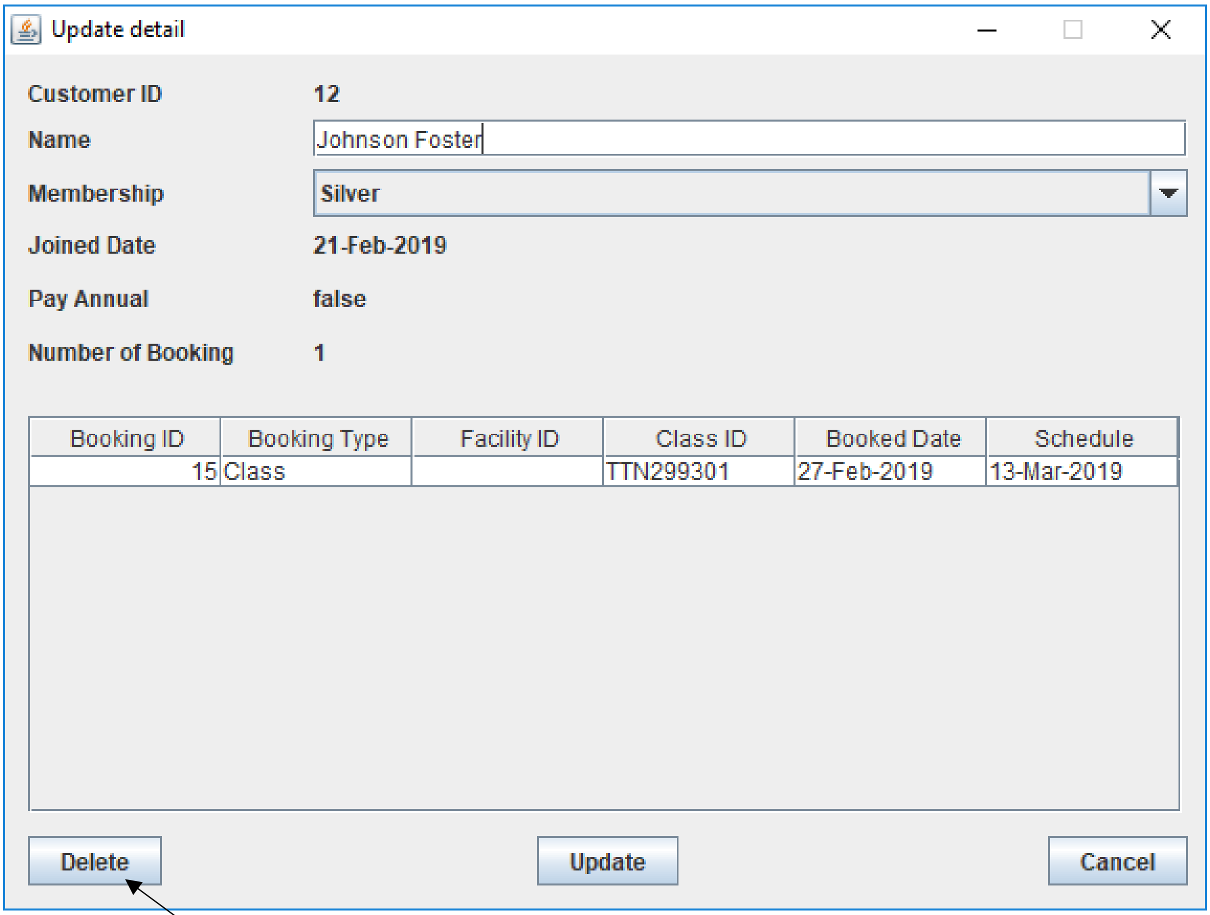
Sort by Membership: Silver

People who pay annual fee: false

Search Name: Joh

Customer ID	Name	Membership	Joined Date	Pay Annual	Updatable
1	John Smith	Silver	20-Feb-2019	false	

The search function allows to search multiple options as shown the searched customer with the selected options and typed text.

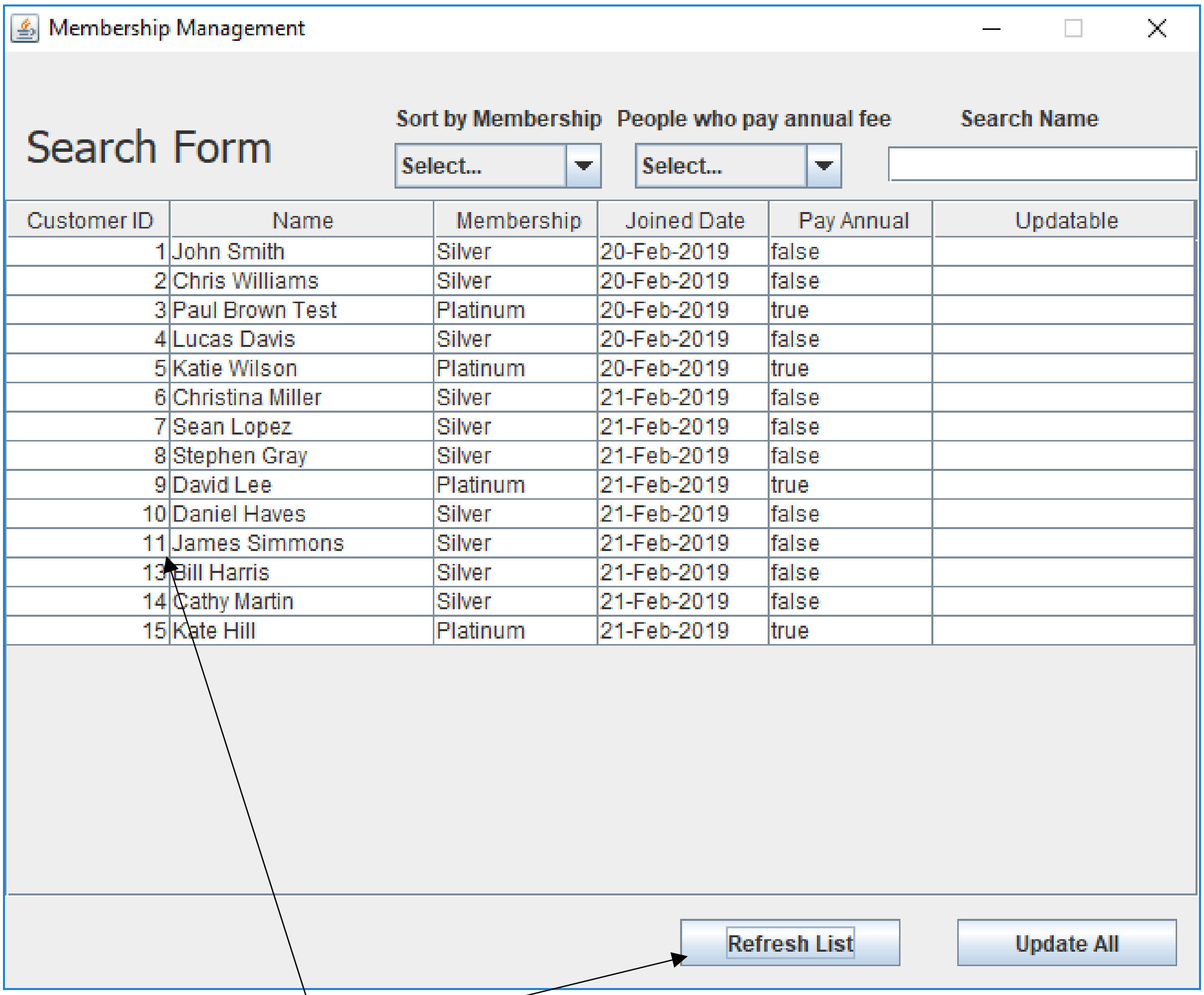


Update detail

Customer ID	12
Name	Johnson Foster
Membership	Silver
Joined Date	21-Feb-2019
Pay Annual	false
Number of Booking	1

Booking ID	Booking Type	Facility ID	Class ID	Booked Date	Schedule
15	Class		TTN299301	27-Feb-2019	13-Mar-2019

After clicking Delete button, remove the customer information from the database. If it is successful, then close the window. If not, the system will print error log.



Search Form

Customer ID	Name	Membership	Joined Date	Pay Annual	Updatable
1	John Smith	Silver	20-Feb-2019	false	
2	Chris Williams	Silver	20-Feb-2019	false	
3	Paul Brown Test	Platinum	20-Feb-2019	true	
4	Lucas Davis	Silver	20-Feb-2019	false	
5	Katie Wilson	Platinum	20-Feb-2019	true	
6	Christina Miller	Silver	21-Feb-2019	false	
7	Sean Lopez	Silver	21-Feb-2019	false	
8	Stephen Gray	Silver	21-Feb-2019	false	
9	David Lee	Platinum	21-Feb-2019	true	
10	Daniel Haves	Silver	21-Feb-2019	false	
11	James Simmons	Silver	21-Feb-2019	false	
13	Bill Harris	Silver	21-Feb-2019	false	
14	Cathy Martin	Silver	21-Feb-2019	false	
15	Kate Hill	Platinum	21-Feb-2019	true	

After clicking Refresh List button, can't see the deleted customer.

References

Hedley, Y. (2019). *Week 3: Software Architecture Design*.

Hedley, Y. (2019). *Week 6: GRASP Patterns 1*.

Hedley, Y. (2019). *Week 7: GRASP Patterns 2 and GoF Patterns 1*.