DSD2 Coursework

Andrew Muir

“I declare that all work submitted for this coursework is the work of Andrew Muir alone unless stated otherwise.”

Acceptance testing

|  |  |  |
| --- | --- | --- |
| **ID** | **Description** | **Completed (Y/N)** |
| 1 | Methods to add a new product, register a new customer, and, create a customer visit with one or more products. | Y |
| 2 | Methods to list all customers, all products and all customer visits | Y |
| 3 | A method to retrieve a specified customer visit’s details. | Y |
| 4 | A method to update a specified customer visit by adding products. | Y |
| 5 | A method to delete a specified customer – customer visits should be deleted | Y |
| 6 | A method to show all customer visit and product details for a specified customer. | Y |
| 7 | A method to list products viewed on any customer visit and which customers viewed them. | N |
| 8 | After you have implemented the above methods modify the Customer Visit class in such a way that all customer visits also have a Calendar instance variable that records when a customer visit is created. Re-test the previously created methods. | Y |
| Enhancement | A Gold Customer will, additionally, have a discount property, implement this. | Y |

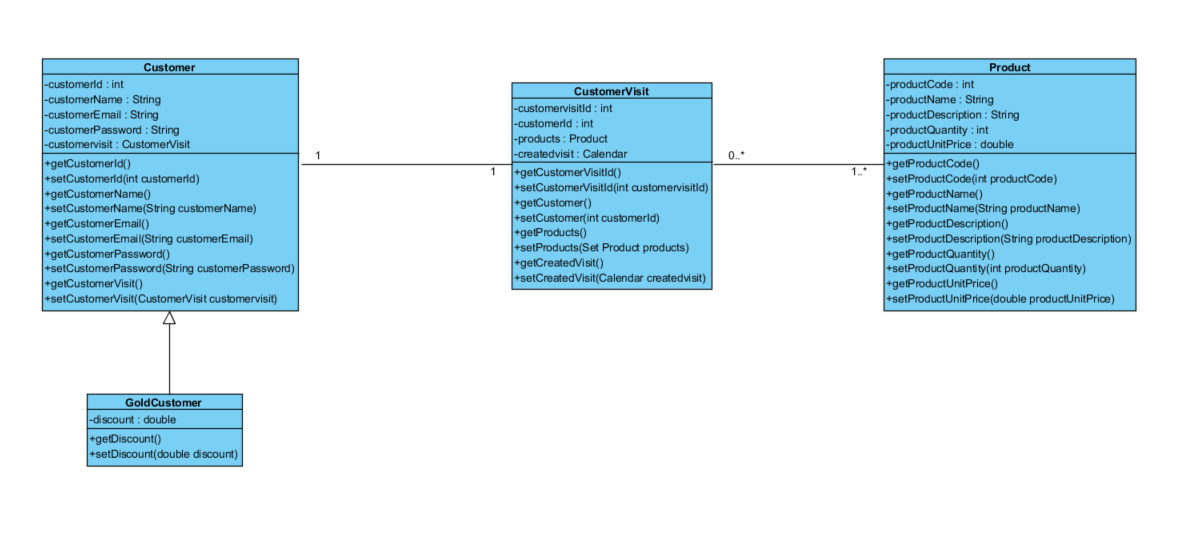
Initial Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Title** | **Description** | **Result** | **Comments** |
| 1 | Register customer | Testing the register customer method with hard coded data | Successfully adds all of the specified customers, customer visits and products | No code to handle duplication, could become a problem |
| 2 | List all customers | Should produce a list of all customers | All customers are displayed | No code to hide passwords |
| 3 | List all products | Should produce a list of all products | All products are displayed | Could be better with code to add in the currency of the unit price |
| 4 | List all customer visits | Should produce a list of all customer visits | All customer visits are displayed | Could be better with code to hide the empty parenthesis of a customer visit without a product |
| 5 | Retrieve specified customer visit details | Should return a single customer visit result with the id of 1 | Customer visit is displayed with the correct id |  |
| 6 | Delete customer | Specified customer should be deleted with the customer visit object being updated also | Customer was deleted but customer visit did not update |  |
| 7 | Update customer visit | Customer visit should be updated with products | Customer visit successfully updated |  |
| 8 | Products viewed on customer visits | List products viewed on any customer visit and which customers viewed them. | Unable to figure this out |  |

Final testing

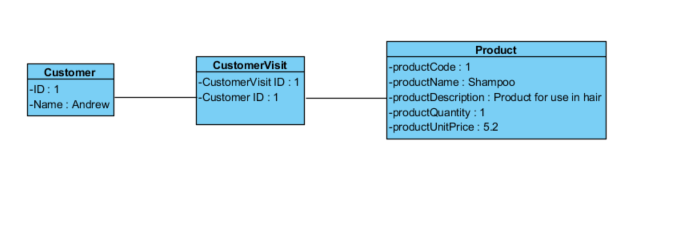
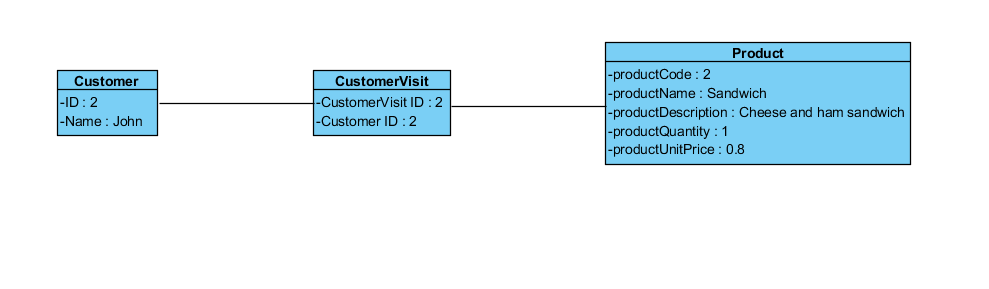
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Title** | **Description** | **Result** | **Comments** |
| 1 | Register customer | Testing the register customer method with hard coded data | Successfully adds all of the specified customers, customer visits and products | No code to handle duplication, could become a problem |
| 2 | List all customers | Should produce a list of all customers | All customers are displayed | No code to hide passwords |
| 3 | List all products | Should produce a list of all products | All products are displayed | Could be better with code to add in the currency of the unit price |
| 4 | List all customer visits | Should produce a list of all customer visits | All customer visits are displayed | Could be better with code to hide the empty parenthesis of a customer visit without a product |
| 5 | Retrieve specified customer visit details | Should return a single customer visit result with the id of 1 | Customer visit is displayed with the correct id |  |
| 6 | Delete customer | Specified customer should be deleted with the customer visit object being updated also | Customer was deleted and customer visit updated | Fixed this after initial testing |
| 7 | Update customer visit | Customer visit should be updated with products | Customer visit successfully updated |  |
| 8 | Products viewed on customer visits | List products viewed on any customer visit and which customers viewed them. | Unable to figure this out |  |
| 9 | Register customer with calendar | Add in calendar property to Customer Visit class and test if it works | Successfully added in calendar |  |
| 10 | Implement gold customer property | Add in gold customer functionality with discount property | Successfully tested, with a new class added to inherit Customer attributes and operations |  |

Class diagram



For the solution, I decided on using four classes to represent the system. As the gold customer only has an additional discount property to add it seemed best to separate it as a different class, inheriting the rest of the attributes and operations from the Customer class. The Customer and Customer Visit classes have a one to one relationship as a customer can only have one visit at a time. A customer visit can have one to many products. Additionally, product can have either zero or many customer visits.

Object graph



An object graph visualises how the system is represented during the active operation of the program. As you can see above, the Customer named Andrew interacts with the Customer Visit object, as opposed to interacting with the Product object itself. The Customer Visit object has a Customer Visit Id of 1, and a Customer Id matching that of the Customer Id in the Customer object. This then goes on to show the details of the Product object that the Customer Visit interacts with during the running of the application. In this case, it is Shampoo that the Customer was looking at, with a description, quantity and unit price. In the second example the Customer named John looks at a Sandwich.