Ain Shams University

Faculty of Engineering

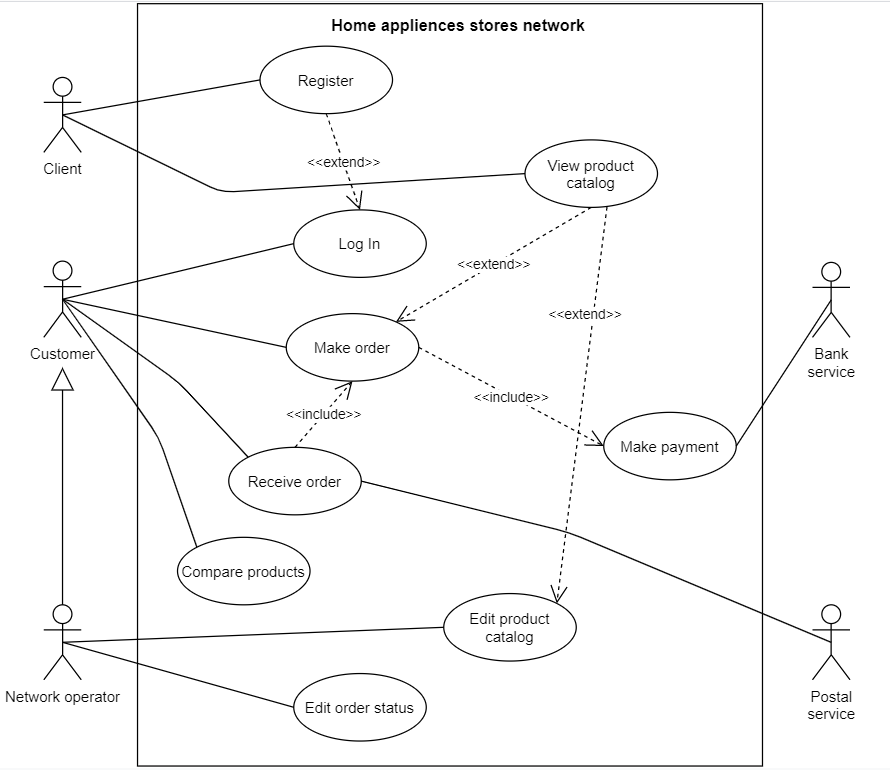
Advanced Software Engineering CSE 608

Dr Islam El-Maddah

Assignment no 1

to be handed by 15 Dec 2020

The following is a use case of home appliance store, client is a general user for this system and when he or she registers he becomes a customer.



1. Explain why there is an arrow between the two actors network operator and customer **[5 marks]**

**One of the actor is a parent class for the other so customer is a general class and network operator can do whatever the customer can do i.e. he can do the 4 case studies connected with customer in addition to edit product catalog and edit order status**

1. Explain how the customer can view product catalog **[5 marks]**

**During make order use case of customer which extends view catalog he can see the products**

1. Explain whether bank service and postal service are other software programs or humans or hardware devices. **[5 marks]**

**Bank service is a software that checks the credit cards and make payment**

**Post service is a human that takes the order to deliver it to the client**

1. Create a detailed use case description for the use case make order**[15 marks]**

**Make order need to list all logical steps and speak about pre post conditions which actors etc**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Use case name** | **Make order** |  |
| **Use case actors** | **Customer** |  |
| **extends** | **View catalog** |  |
| **includes** | **Make payment** |  |
| **Pre conditions** | **customer is logged in** |  |
| **Post conditions** | **Order detailed is saved and ready to be delivered** |  |
|  |  |  |
| **Normal course** | **Actor** | **1-Select make order** |
|  | **System** | **2-Display user interface** |
|  | **Call view catalog use case** | **3** |
|  |  |  |
|  | **Actor** | **4-Decide and select which product** |
|  | **System** | **5-System saves information** |
|  | **Actor** | **6-Add details like quantity** |
|  | **System** | **7-System saves information** |
|  | **Actor** | **8-Add payment details** |
|  | **Call make payment** | **9** |
|  | **system** | **10-Confirms and saves order data** |
|  |  |  |
| **Upnormal cases** | **actor** | **4.1 user cancels** |
|  | **system** | **4.2 Display the main window** |
|  | **actor** | **6.1 Need other products** |
|  | **system** | **6.2 Back to step 2** |
|  | **Make payment is not working** | **9.1 payment is not accepted** |
|  | **system** | **9.2 Display warning message** |

1. Create a number of user stories to replace the use case, each user story should have a value level showing how important it is and points showing how much effort is needed in it. **[15 marks]**

**User stories is replacing the use cases but more basic they do not mention user interface or dialog boxes etc**

**User story**

**Customer can make order to buy needed products**

**Value is high compared to the other user stories because the system is mainly for selling products**

**Value is higher than other user stories like amend order, view catalog, recommend product etc**

Points of effort could be measured as number of objects that need to be accessed and is it difficult or usual task

Points are 3/5 or middle as it is not difficult task

1. Create a class diagram showing the most important classes needed in this system and how they are related **[15 marks]**

**The classes will appear from the description of the use cases like the one we just had make order**

**Looks like there Order (center class) , customers, products, may have extra information like offers, supplier, seller, employee**

* **Customer may have more than one order**
* **An order could contain many products of different types**
* **Each product has price**
* **Orders should have information to know if they have been paid or not**
* **Order payment may be a class related one-one to the orders**
* **If we need to add control and boundary classes we need to have one boundary class for each actor/use case and one control class for each use case**
* **So for make order since it is linked with customer only we will have**
* **Make\_order\_dialog class which takes care of how a customer as human input and read data from the system related to making order**

**Make\_order\_controller class which takes care of what code will be there once the dialog box is opened this class/code should be separate from the dialog to make sure the operation and display tiers are not hardwired**

Make\_order\_dialogBox

Make\_order\_controller

Product

Price

stock

Product line

quanitiy

Customer

name

Order

date

Display\_catalog\_controller

Network operator

Payment

Date

credit card

Display\_catalog\_dialogBox

Solid lines are associations between the classes

dotted lines mean the class is related to the other class but not 1:1 or 1:m

say we are going to search for one product and display it