Suppose you have to develop a **Loyalty Account application** for Walmart. Every time you buy something at Walmart, you get loyalty points. If you have enough loyalty points you can exchange them for cash or you can buy certain products with them. The Loyalty Account application keeps track of the number of loyalty points every customer has. This Loyalty Account application for Walmart should support the following functionality:

```
createAccount(String accountNumber, String customerName, String
customerPhone, String customerEmail)
addLoyaltyPoints(String accountNumber, int numberOfPoints)
subtractLoyaltyPoints(String accountNumber, int numberOfPoints)
getAccount(String accountNumber)
deleteAccount(String accountNumber)
```

The Loyalty Account application for Walmart has the following additional requirements:

- All data is stored in the database
- The system keeps track of the history of all additions and subtractions of points on an account including the date, time and number of points.
- The method getAccount() should return all account information including the history of all additions and subtractions of points on this account.
- The system supports basic, loyal and premium customers. Basic customers get the normal number of points they earn, but loyal customers get 1 extra point for every 10 points they earn. Premium customers get 1 extra point for every 3 points they earn. It should be easy to change this algorithm.
- Every time points are added or subtracted, the customer gets an email with all account details. It should be easy to notify the customer in other ways like SMS or WhatsApp messages.

a. Draw the class diagram of the loyalty Account application for Walmart.

   Draw the solution of this question in **StarUML** and upload your solution as a **JPG picture** (only JPG is allowed) as solution to this question

   Make sure you **add all necessary UML elements** (attributes, multiplicity, etc.) to communicate the important parts of your design. **Use the best practices** we learned in this course. **Do NOT add extra complexity** that is not needed.

When you finished the Loyalty Account application for Walmart, you are asked to develop a Loyalty Account application for MovieNight, a large movie theater that wants to give loyalty points to their customers. You discover that the requirements for this system are almost the same as the requirements for the loyalty Account application for Walmart. The requirements that are different are:

- The system does not supports basic, loyal and premium customers, but the number of points you get for seeing a movie depends on the total of points you have at the moment. If you have less than 100 points, you get 2 points for every movie you watch. If you have more than 100 points, but less than 1000 points you get 3 points for every movie you watch. If you have 1000 points or more, you get 6 points for every movie you watch.

- Every time points are added or subtracted, the customer gets a WhatsApp message with the updated total of points
- Every time points are added or subtracted, this is written to a log file.

The lead architect in your company decides to develop a Loyalty Account **framework** that can be used to develop Loyalty Account applications for different types of customers.

The framework supports the following functionality:

```
createAccount(String accountNumber, String customerName, String customerPhone, String customerEmail)
addLoyaltyPoints(String accountNumber, int numberOfPoints)
subtractLoyaltyPoints(String accountNumber, int numberOfPoints)
getAccount(String accountNumber)
deleteAccount(String accountNumber)
```

The framework has the following requirements:

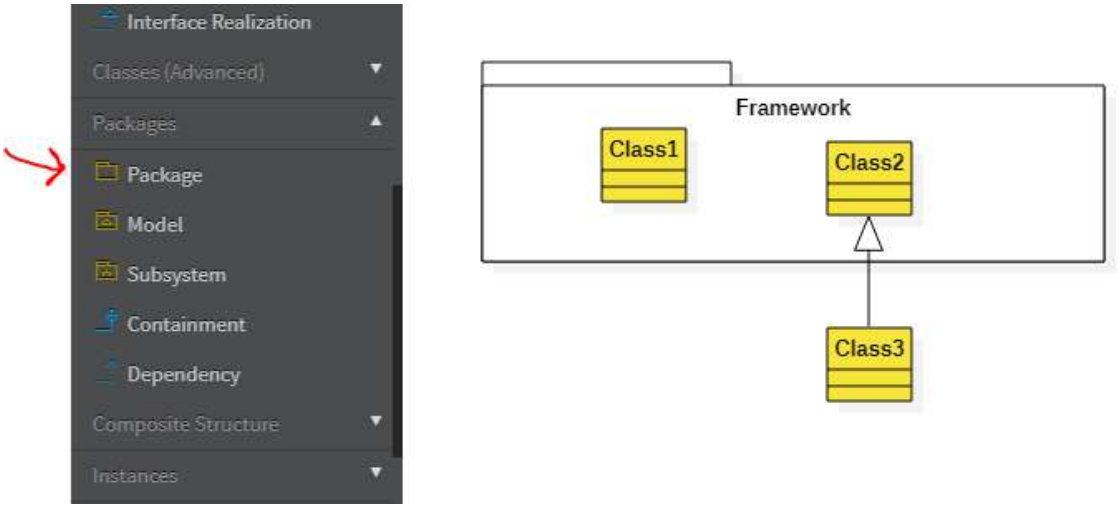
- All data is stored in the database
- The system keeps track of the history of all additions and subtractions of points on an account including the date, time and number of points.
- The method getAccount() should return all account information including the history of all additions and subtractions of points on this account.
- The framework should support all kind of different account types where the number of points you get differ for every account type.
- The framework should support both email and sms notification messages to the customer when points are added or subtracted.

b. Draw the class diagram of the loyalty Account **framework**.

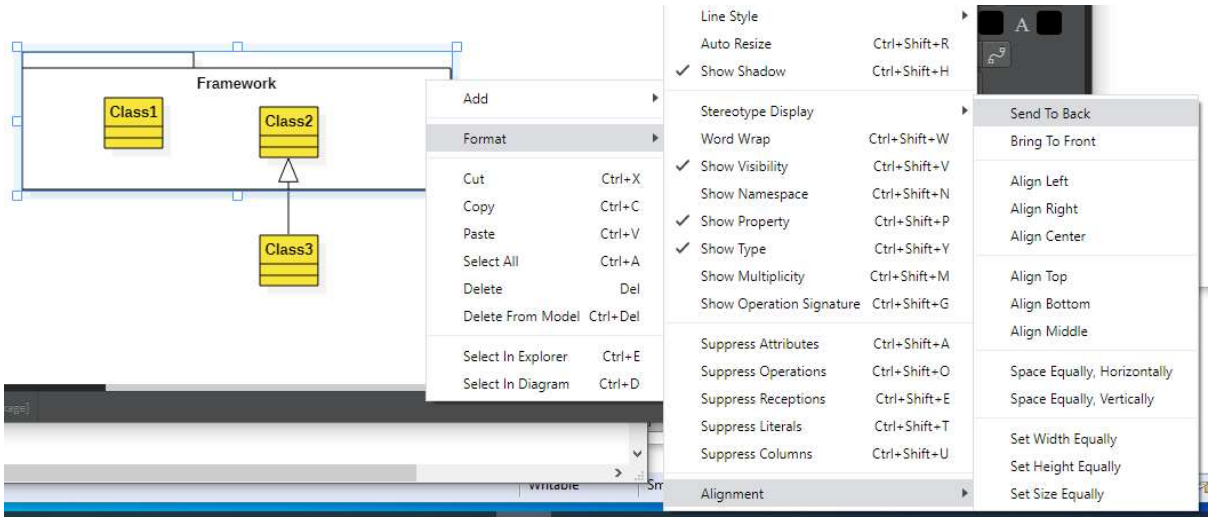
   Draw the solution of this question in StarUML and upload your solution as a **JPG picture** (only JPG is allowed) as solution to this question

   Make sure you **add all necessary UML elements** (attributes, multiplicity, etc.) to communicate the important parts of your design. **Use the best practices** we learned in this course. **Do NOT add extra complexity** that is not needed.

c. Now we want to implement the Loyalty Account application for MovieNight using the Loyalty Account framework of part b. In the class diagram of part b, add the classes that are needed to implement the Loyalty Account application for MovieNight. Clearly show which classes are inside the framework, and which classes are outside the framework. You can do this by placing a package around the framework classes.



If you place a package over the existing classes, the package goes on top of the existing classes. To bring the package to the back, right click the package and select **Format->Alignment->Send to Back.**



Upload your solution as a **JPG picture** (only JPG is allowed) as solution to this question

d. Using the classes of the Loyalty Account application for MovieNight of part c, draw the sequence diagram of the scenario below. **Before the scenario starts, assume the particular customer has already 998 points in her account**.

Scenario:

1. The customer watches a movie so we call addLoyaltyPoints() for her account.
2. The customer watches another movie so we call addLoyaltyPoints() again

Upload your solution as a **JPG picture** (only JPG is allowed) as solution to this question