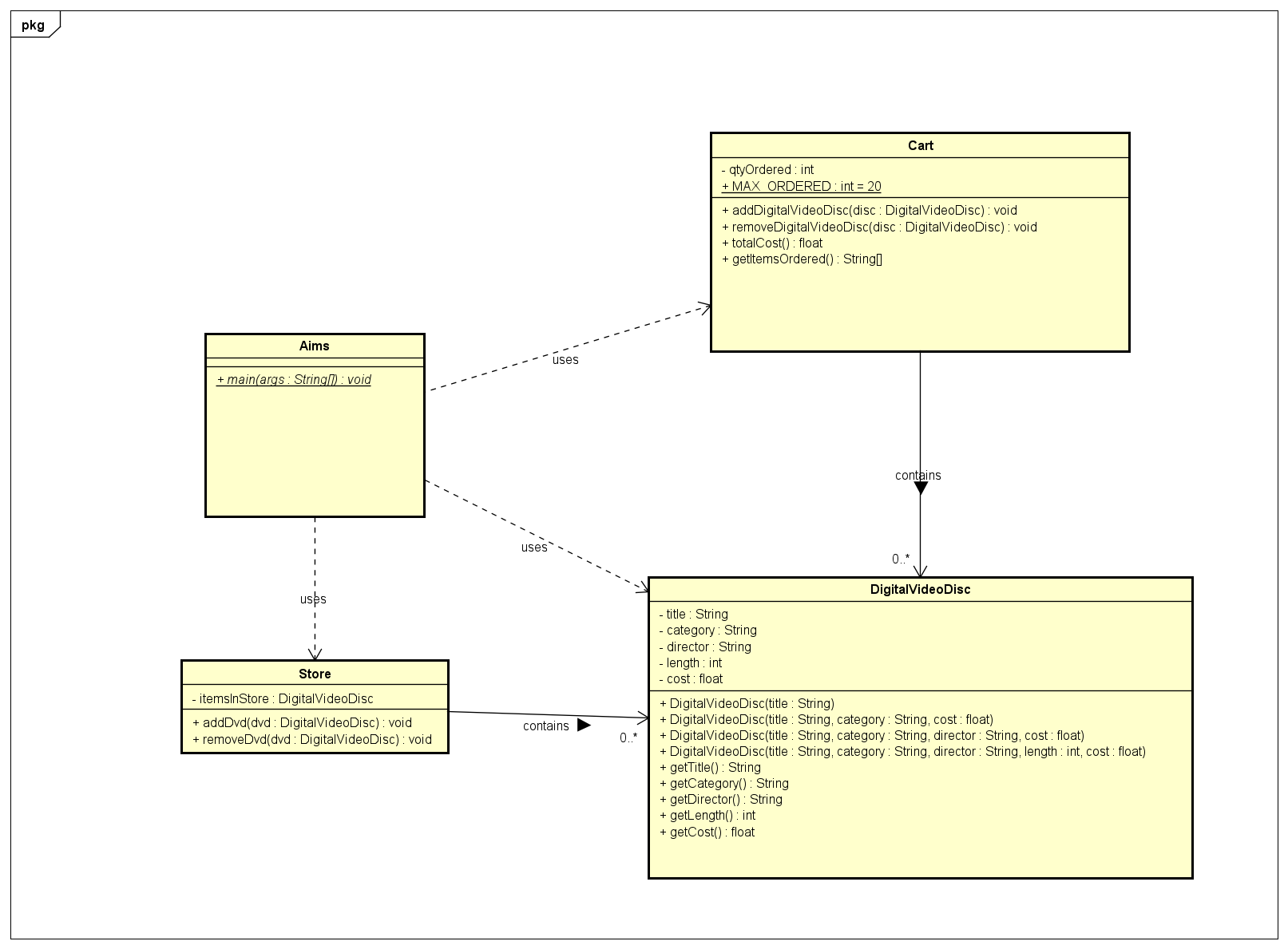
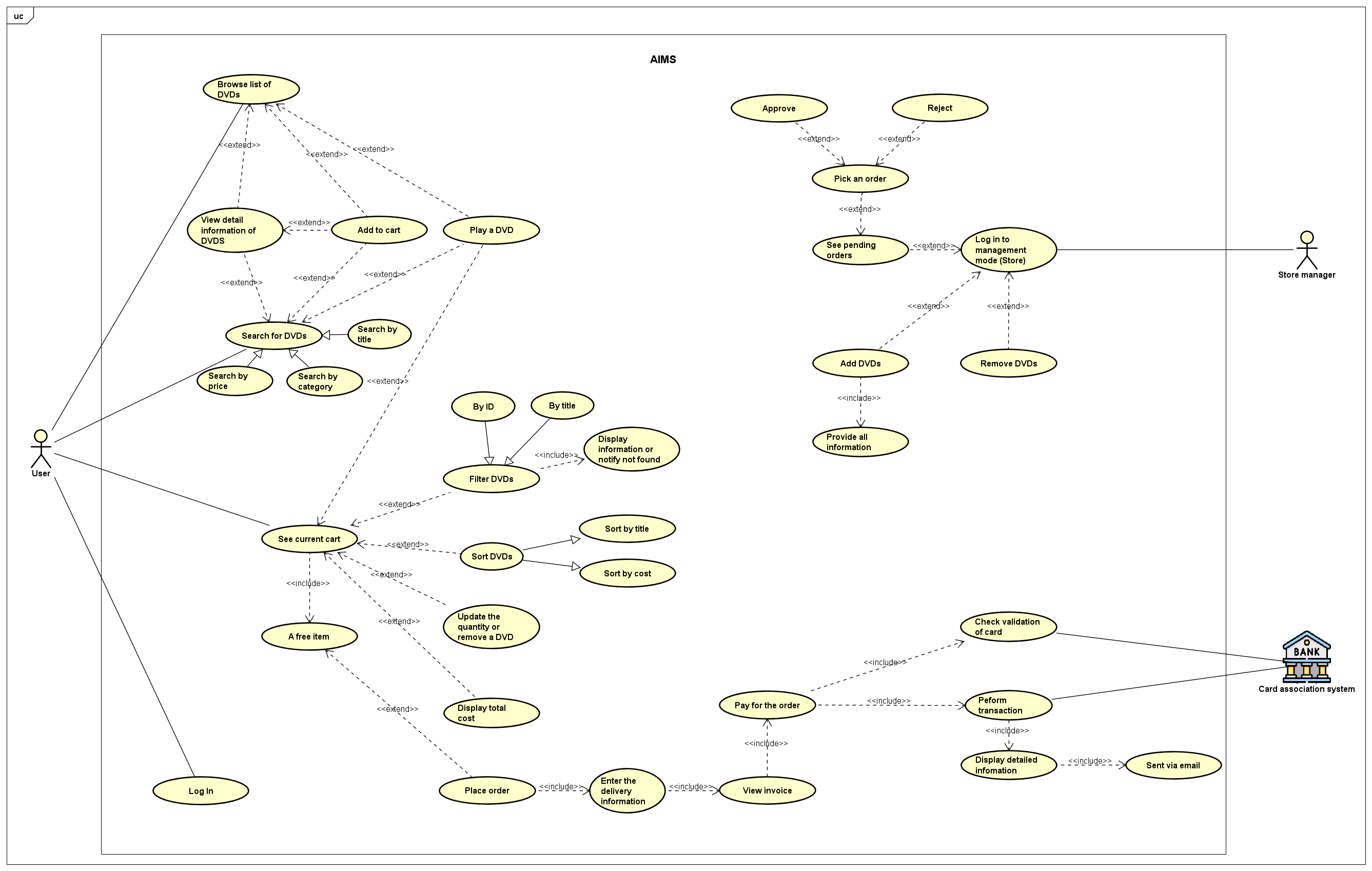
**REPORT LAB 03**

**Name: Nguyễn Vũ Thủy**

**Student ID: 20235562**

1. Update use-case diagram and class diagram





2. Working with method overloading

A screenshot of a computer

Description automatically generated

- *Try to add a method addDigitalVideoDisc which allows to pass an arbitrary number of arguments for dvd. Compare to an array parameter. What do you prefer in this case?*

=> I think I prefer using the array parameter because it allows me to easily determine how many DVDs I need to add to the cart for validation, whereas handling an arbitrary number of DVDs would be more difficult.

3. Passing parameter

A screen shot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

* ***Is JAVA a Pass by Value or a Pass by Reference programming language?***
  + Java Pass by Value. For example, if we pass an object into a method in Java ( swap(DVD dvd1, DVD dvd2) ) then in the method swap only recieve the address value point to the dvd1 Object and dvd2 object in the memory so if we try to swap by Obj tmp = dvd1; dvd1 = dvd2; dvd2 = tmp then it won't work. Because it is only dvd1 and dvd2 in the method change value for each other which does not affect original 2 objects.
* After the call of **swap(jungleDVD, cinderellaDVD)** why does the title of these two objects still remain?
  + As i said earlier, o1 and o2 are just local variables of method swap so swap value of o1 and o2 does not affect the value of original objects which are jungleDVD and cinderellaDVD. So the value of jungleDVD and cinderellaDVD still remain
* After the call of **changeTitle(jungleDVD, cinderellaDVD.getTitle())** why is the title of the JungleDVD changed?
  + In changeTitle, we have passed down the address of jungleDVD object so when we modify the title of dvd ( which is the jungleDVD object) it also changes the title of jungleDVD because both point to the same Object.

4. Debugging Java in Eclipse

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

Result:

A screenshot of a computer program

Description automatically generated

5. Classifier Member and Instance Member

A screen shot of a computer

Description automatically generated

6. Open the **Cart** class

A screen shot of a computer

Description automatically generated

- Write a **toString()** method for the **DigitalVideoDisc** class. What should be the return type of this method?

+ The method should return a String.

A computer screen shot of a program

Description automatically generated

A screenshot of a computer

Description automatically generated

7. Implement the **Store** class

A computer screen shot of a program

Description automatically generated

A screen shot of a computer program

Description automatically generated

**9. String, StringBuilder and StringBuffer**

A screenshot of a computer program

Description automatically generated

String concatenation

A screen shot of a computer

Description automatically generated

* Very slow. Take so much time to read the file\

Using stringBuffer & string Builder:

A screenshot of a computer program

Description automatically generated

A computer screen shot of a program

Description automatically generated

- It is significantly faster and more efficient.

- The main difference between StringBuffer and StringBuilder is thread safety. In most cases, I believe StringBuilder is the preferred choice unless thread safety is a concern.

- Although it still reaches the limit, this is due to the file I created being very large (around 4GB). → It is definitely an improvement over string concatenation.