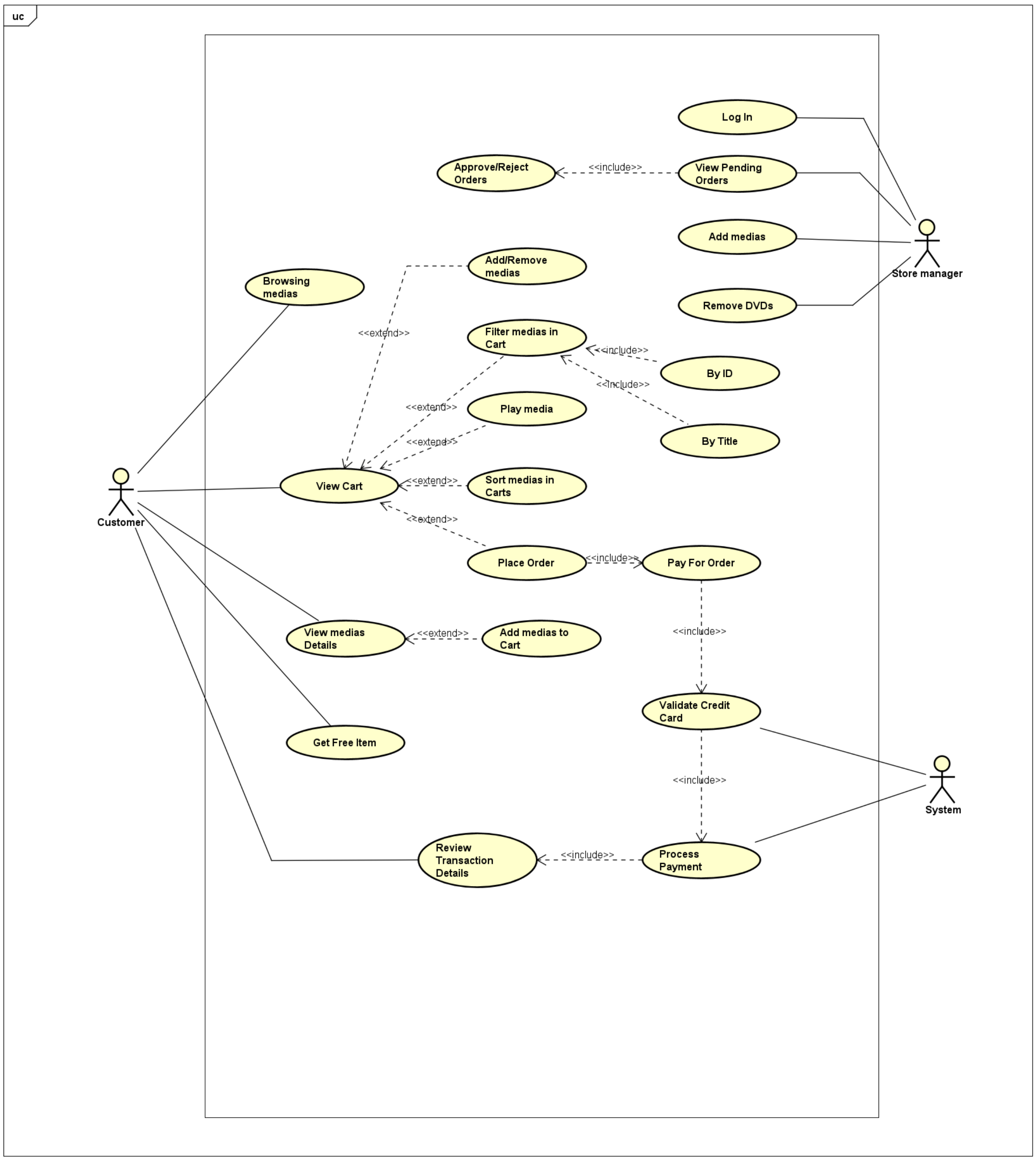
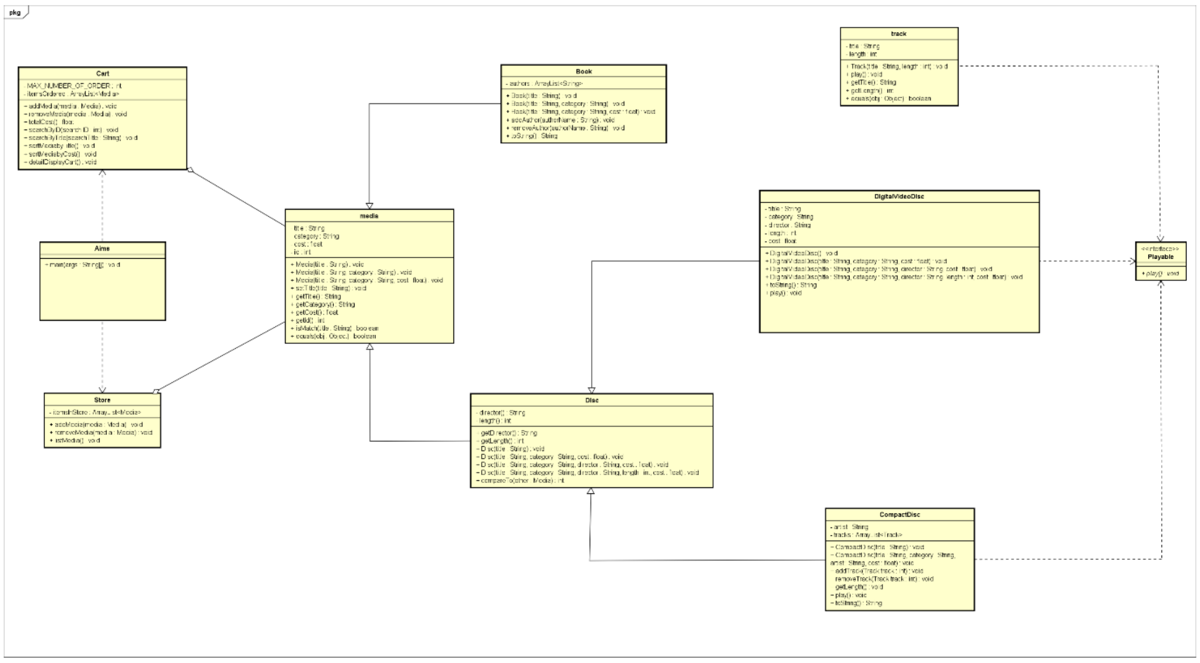
Lê Nhật Hoàng 20235498

1. Usecase diagram



2. Class diagram



3.

Question: Which classes are aggregates of other classes? Checking all constructors of whole classes if they initialize for their parts?

Class Store aggregates class Media.

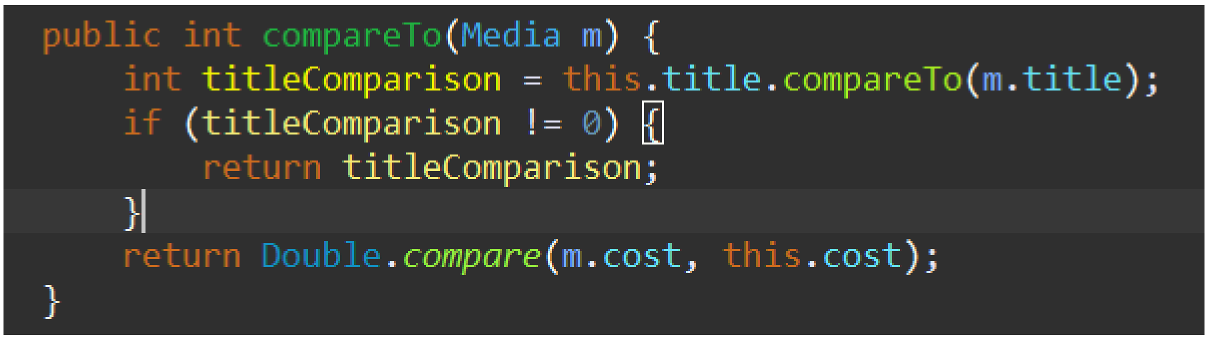
Class Cart aggregates class Media.

Class CompactDisc aggregates class Track.

Question: Alternatively, to compare items in the cart, instead of using Comparator, we can use the Comparable interface and override the compareTo()method. You can refer to the Java docs to see the information of this interface. Suppose we are taking this Comparable interface approach, what class should implement the Comparable interface?

The Media class should implement the Comparable interface

In those classes, how should you implement the compareTo()method to reflect the ordering that we want?



```
public int compareTo(Media m) {
    int titleComparison = this.title.compareTo(m.title);
    if (titleComparison != 0) {
        return titleComparison;
    }
    return Double.compare(m.cost, this.cost);
}
```

Question: Can we have two ordering rules of the item (by title then cost and by cost then title) if we use this Comparable interface approach?

We cannot because the Comparable interface allows only one natural ordering for a class.

Question: Suppose the DVDs has a different ordering rule from the other media types, that is by title, then decreasing length, then cost. How would you modify your code to allow this?

We can write method compareTo() in class DigitalVideoDisc to make it overridden