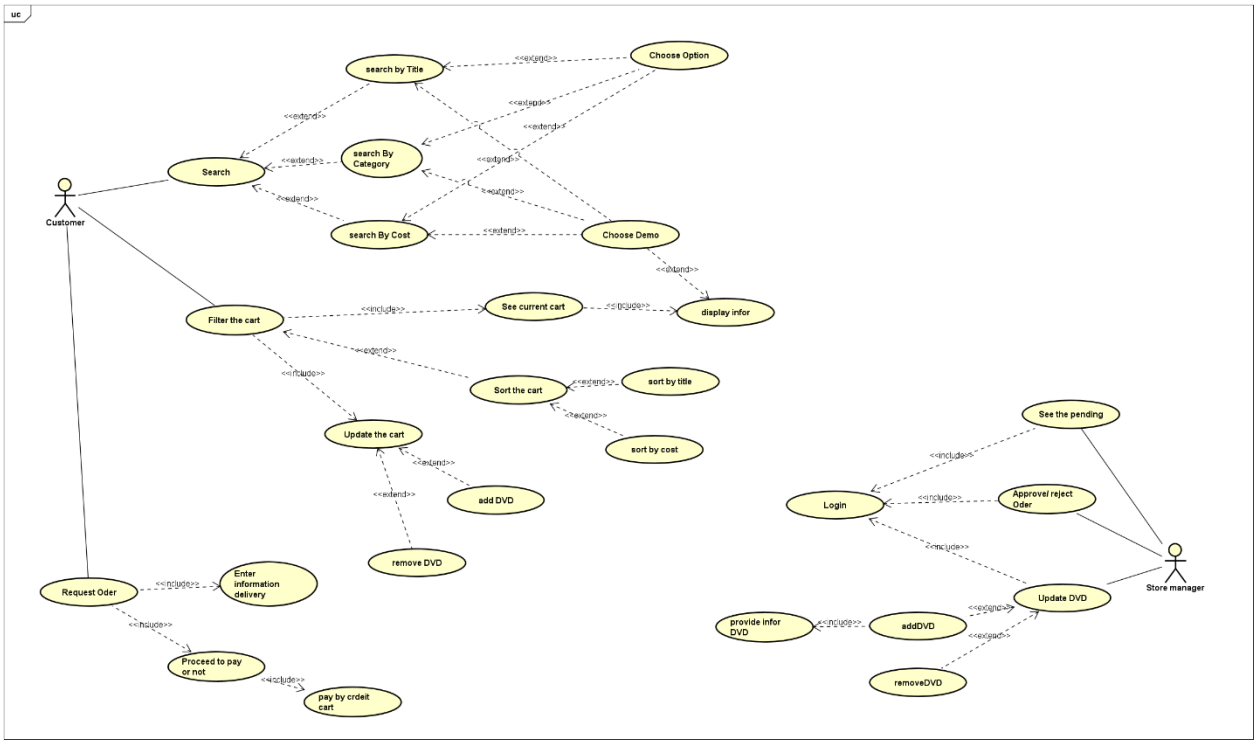
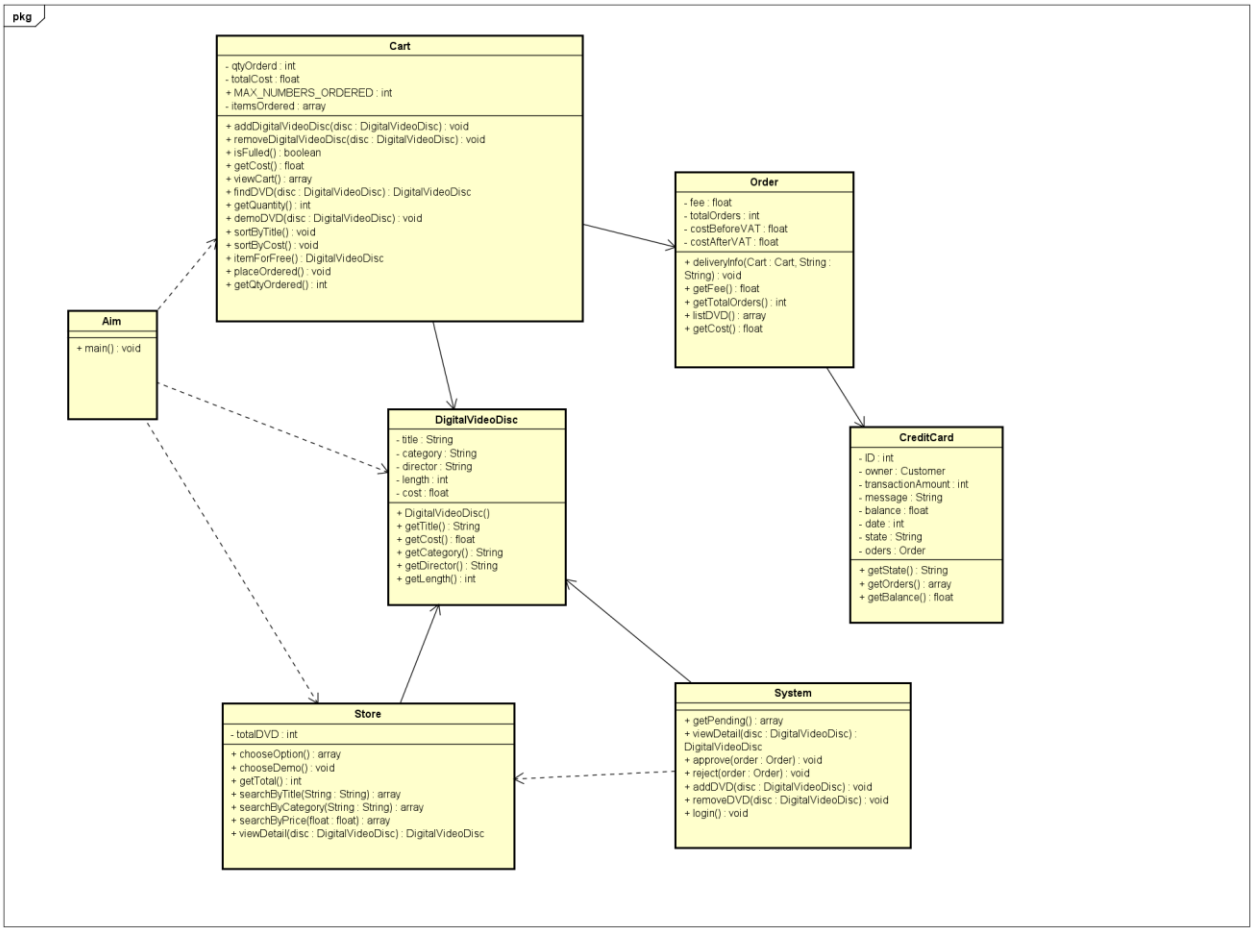
Lab 02: Problem Modeling and Encapsulation

3. Use case diagram



4. UML Class Diagram for use cases related to cart management



5. Create Aims class

```
aProject > store > Aims.java
package store;

public class Aims {
    public static void main(String[] args){
        Cart anOrder = new Cart();

        DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King" , "Animation" , "Roger Allers",87, 19.95f);
        anOrder.addDigitalVideoDisc(dvd1);

        DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star Wars" , "Science Fiction" , "George Lucas",87, 24.95f);
        anOrder.addDigitalVideoDisc(dvd2);

        DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladin" , "Animation", 18.995f);
        anOrder.addDigitalVideoDisc(dvd3);

        anOrder.removeDigitalVideoDisc(dvd1);

        System.out.println("Total Cost is:");
        System.out.println(anOrder.getTotalCost());


    }
}
```

6. Create the DigitalVideoDisc class and its attributes

```java
package store;

public class DigitalVideoDisc {
    private String title;
    private String category;
    private String director;
    private int length;
    private float cost;
```

7. Create accessors and mutators for the class DigitalVideoDisc + 8. Create Constructor method

```java
    DigitalVideoDisc(String title) {
        this.title = title;
    }

    DigitalVideoDisc(String title, String category, float cost) {
        this(title);
        this.category = category;
        this.cost = cost;
    }

    DigitalVideoDisc(String director, String category, String title, float cost) {
        this(title, category, cost);
        this.category = director;
    }

    DigitalVideoDisc(String title, String category, String diretor, int length, float cost) {
        this(diretor, category, title, cost);
        this.length = length;
    }

    public String getTitle() {
        return this.title;
    }

    public float getCost() {
```

```java

    DigitalVideoDisc(String title, String category, String diretor, int length, float cost) {
        this(diretor, category, title, cost);
        this.length = length;
    }

    public String getTitle() {
        return this.title;
    }

    public float getCost() {
        return this.cost;
    }

    public String getCategory() {
        return this.category;
    }

    public int getLength() {
        return this.length;
    }

    public String getDirector() {
        return this.director;
    }

```

Question:

- If you create a constructor method to build a DVD by title then create a constructor method to build a DVD by category. Does JAVA allow you to do this?

- Answer: No, because two constructor have same paramatter (as String type) so it not overloading

9. Create the Cart class to work with DigitalVideoDisc

```java
package store;

public class Cart {
    public static final int MAX_NUMBERS_ORDERED = 20;
    private int qtyOrdered = 0;
    private float totalCost = 0f;
    private DigitalVideoDisc itemsOrdered[] = new DigitalVideoDisc[MAX_NUMBERS_ORDERED];

    public int getQtyOrderd() {
        return this.qtyOrdered;
    }

    public boolean isFulled() {
        if (qtyOrdered == MAX_NUMBERS_ORDERED)
            return true;
        else
            return false;
    }

    public void addDigitalVideoDisc(DigitalVideoDisc disc) {
        if (isFulled())
            System.out.println(x:"The Cart is almost full");
        itemsOrdered[qtyOrdered] = disc;
        qtyOrdered++;
        System.out.println(x:"The disc has been added");
    }

    public void removeDigitalVideoDisc(DigitalVideoDisc disc) {
        int index = -1;
        for (int i = 0; i < 20; i++)
            if (itemsOrdered[i] == disc)
                index = i;
        if (index == -1) {
            System.out.println(x:"Can't find the disc");
```

```java
            System.out.println(x:"The Cart is almost full");
        itemsOrdered[qtyOrdered] = disc;
        qtyOrdered++;
        System.out.println(x:"The disc has been added");
    }

    public void removeDigitalVideoDisc(DigitalVideoDisc disc) {
        int index = -1;
        for (int i = 0; i < 20; i++)
            if (itemsOrdered[i] == disc)
                index = i;
        if (index == -1) {
            System.out.println(x:"Can't find the disc");
        } else {
            qtyOrdered--;
            for (int i = index; i < qtyOrdered; i++) {
                itemsOrdered[i] = itemsOrdered[i + 1];
            }
        }
        System.out.println(x:"The disc has been removed");
    }

    public float getTotalCost() {
        for (int i = 0; i < qtyOrdered; i++) {

                totalCost += itemsOrdered[i].getCost();
        }
        return totalCost;
    }

}
```

10.Create Carts of DigitalVideoDiscs

```java
package store;

public class Aims {
    Run | Debug
    public static void main(String[] args){
        Cart anOrder = new Cart();

        DigitalVideoDisc dvd1 = new DigitalVideoDisc(title:"The Lion King" , category:"Animation" , diretor:"Roger Allers",length:87,
        anOrder.addDigitalVideoDisc(dvd1);

        DigitalVideoDisc dvd2 = new DigitalVideoDisc(title:"Star Wars" , category:"Science Fiction" , diretor:"George Lucas",length:87
        anOrder.addDigitalVideoDisc(dvd2);

        DigitalVideoDisc dvd3 = new DigitalVideoDisc(title:"Aladin" , category:"Animation", cost:18.995f);
        anOrder.addDigitalVideoDisc(dvd3);

        anOrder.removeDigitalVideoDisc(dvd1);

        System.out.println(x:"Total Cost is:");
        System.out.println(anOrder.getTotalCost());


    }
}
```

```
18          System.out.println(x:"Total Cost is:");
19          System.out.println(anOrder.getTotalCost());
20
```

PROBLEMS OUTPUT DEBUG CONSOLE TIMELINE TERMINAL

```
The disc has been added
The disc has been added
The disc has been removed
Total Cost is:
43.945
PS E:\OOP\javaLab\Java\lab2>
```