

BÁO CÁO THỰC HÀNH LAB 5

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

[link](#)

Mục lục

1. Đề bài.....	1
2. Yêu cầu bài toán.....	1
3. Use case diagram	2
4. Class diagram	3
5. Mã nguồn của chương trình	3
6. Demo chương trình.....	79

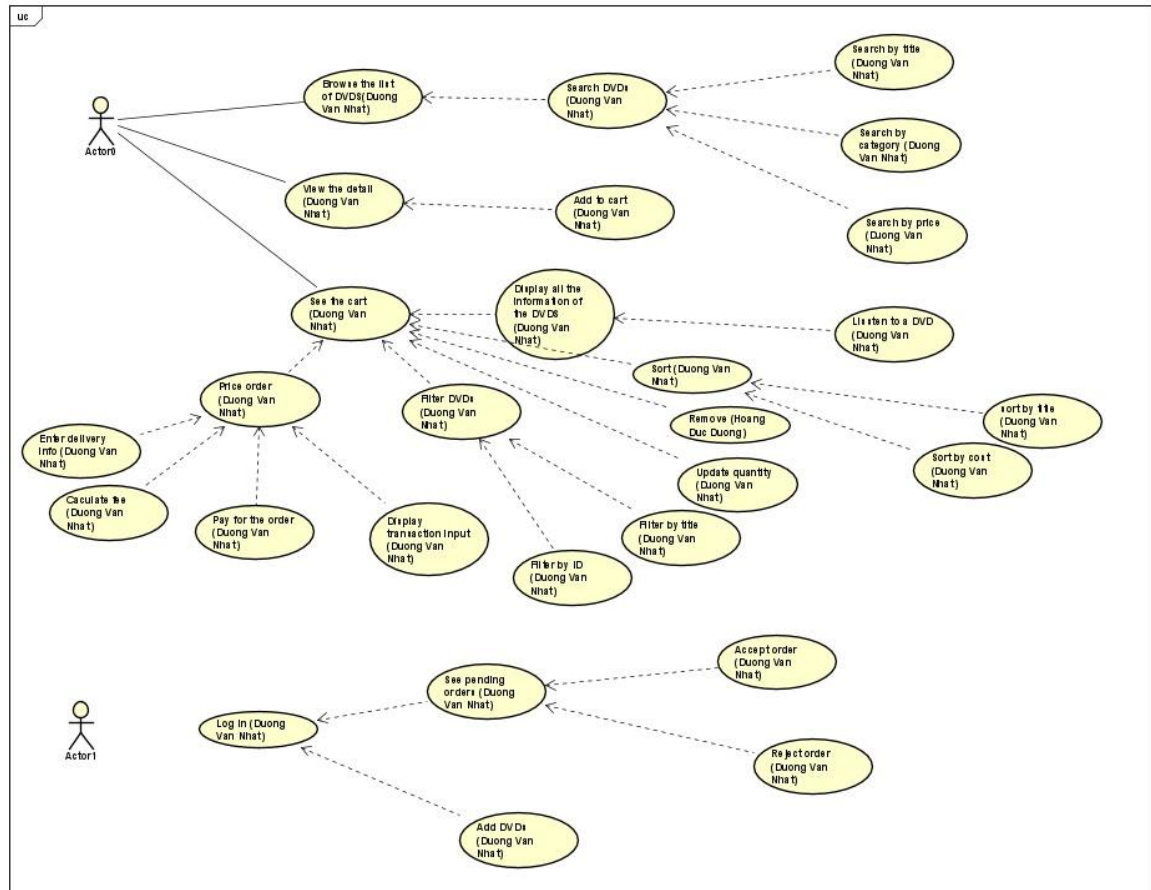
1. Đề bài

Thiết kế hệ thống mới cho dự án AIMS

2. Yêu cầu bài toán

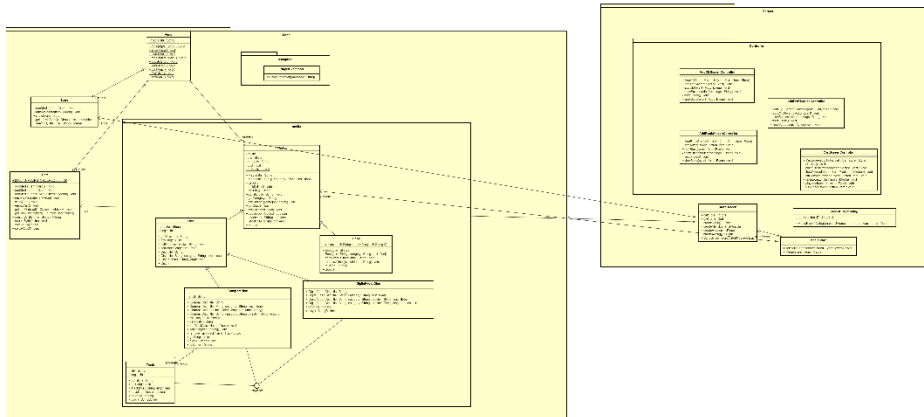
- Đối với người mua hàng
 - Duyệt danh sách các DVD có sẵn trong cửa hàng
 - Tìm kiếm DVD theo: tiêu đề, danh mục, giá cả
 - Xem thông tin chi tiết của 1 DVD
 - Thêm DVD vào giỏ hàng
 - Xem giỏ hàng
 - Sắp xếp DVD trong giỏ hàng theo tiêu đề hoặc chi phí
 - Cập nhật số lượng DVD trong giỏ hàng
 - Đặt hàng:
 - Đối với người quản lý cửa hàng
 - Đăng nhập, kiểm tra quyền
 - Xem danh sách các đơn hàng đang chờ xử lý
 - Xem chi tiết 1 đơn hàng và xử lý nó (đồng ý từ chối đặt hàng) Thêm(xóa) DVD mới vào(ra) danh sách sản phẩm

3. Use case diagram



4.

5. Class diagram



6. Mã nguồn của chương trình

A. hust.soict.hedspi.aims

a) Aims.jav

```
package hust.soict.hedspi.aims;

import hust.soict.hedspi.aims.exception.PlayerException;
import hust.soict.hedspi.aims.media.*;
import hust.soict.hedspi.aims.screen.StoreScreen;

import java.util.Scanner;
public class Aims {
    private static String mediaTitle;
    public static Scanner scanner = new Scanner(System.in);
    public static Store aStore;
    public static Cart aCart = new Cart();
    private static Media foundMedia;
    // Khai báo các biến static
    public static void main(String[] args)
    {
        aStore = new Store();
        DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King",
            "Animation", "Roger Allers", 87, 19.95f );
        aStore.addMedia(dvd1);
    }
}
```

```

        DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star Wars",
            "Science Fiction", "George Lucas", 87, 24.95f );
        aStore.addMedia(dvd2);
        DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladin",
            "Animation", 18.99f );
        aStore.addMedia(dvd3);
        CompactDisc cd1 = new CompactDisc("Disco Elysium", "High Classic",12.3f);
        CompactDisc cd2 = new CompactDisc("Divine dive", "Blue", 16f);
        aStore.addMedia(cd1);
        aStore.addMedia(cd2);
        Book book1 = new Book("Introduction to Demons summoning", "Occult",12f);
        Book book2 = new Book("High and low", "shitpost", 120f);
        aStore.addMedia(book1);
        aStore.addMedia(book2);
        // Tạo các media với các loại khác nhau và thêm vào store
        StoreScreen Screen = new StoreScreen(aStore,aCart);
        // Khởi tạo giao diện cho người dùng với StoreScreen
    }

    public static void showMenu() {
        //Hàm cho thấy menu
        System.out.println("AIMS: ");
        System.out.println("-----");
        System.out.println("1. View store");
        System.out.println("2. Update store");
        System.out.println("3. See current cart");
        System.out.println("0. Exit");
        System.out.println("-----");
        System.out.println("Please choose a number: 0-1-2-3");
        //In ra các lựa chọn
        int choice = scanner.nextInt();
        //Lấy lựa chọn
        switch (choice){
            case 1: storeMenu(); break;
            case 2: updateStore(); break;
            case 3: cartMenu(); break;
            //Với các lựa chọn hợp lệ chạy hàm tương ứng
            case 0: return;
            //Với 0 kết thúc giao diện
            default:
                System.out.println("Error, choose again!");
                //Không báo lỗi và quay lại giao diện
                showMenu();
        }
    }

```

```

}

public static void storeMenu() {
    //Hàm cho thấy có gì trong store
    aStore.showStore();
    //In ra những gì đang có trong store
    System.out.println("Options: ");
    System.out.println("-----");
    System.out.println("1. See a media's details");
    System.out.println("2. Add a media to cart");
    System.out.println("3. Play a media");
    System.out.println("4. See current cart");
    System.out.println("0. Back");
    System.out.println("-----");
    System.out.println("Please choose a number: 0-1-2-3-4");
    //In ra các lựa chọn
    int choice = scanner.nextInt();
    //Lấy lựa chọn
    switch (choice){
        case 0: showMenu(); break;
        case 1:
            System.out.println("Enter media's title: ");
            scanner.nextLine();
            mediaTitle = scanner.nextLine();
            foundMedia = aStore.searchByTitle(mediaTitle);
            if (foundMedia != null) {
                mediaDetailsMenu();
            }
            //Trong lựa chọn 1, lấy tên media rồi tìm nó trong store, nếu tìm
            thấy vào mediaDetailsMenu
            //Xong quay trở lại giao diện
            storeMenu();
            break;
        case 2:
            System.out.println("Enter media's title");
            scanner.nextLine();
            String mediaTitleAdd = scanner.nextLine();
            foundMedia = aStore.searchByTitle(mediaTitleAdd);
            //Lấy tên media rồi tìm nó trong store và lưu kết quả vào biến
            foundMedia
            if (foundMedia != null) {
                aCart.addMedia(foundMedia);
                aStore.removeMedia(foundMedia);
                System.out.println("Media added to cart successfully");
                //Nếu có kết quả thì thêm vào giỏ hàng và bỏ khỏi store

```

```

        } else {
            System.out.println("Media not found in the store");
            //Không thì báo không tìm thấy
        }

        storeMenu();
        break;
    case 3:
        System.out.println("Enter media's title");
        scanner.nextLine();
        String mediaTitlePlay = scanner.nextLine();
        foundMedia = aStore.searchByTitle(mediaTitlePlay);
        //Lấy tên media rồi tìm nó trong store và lưu kết quả vào biến
foundMedia
        if (foundMedia instanceof CompactDisc) {
            CompactDisc cd = (CompactDisc) foundMedia;
            try {
                // Call the play() method on a Media object
                cd.play();
            } catch (PlayerException e) {
                // Handle the PlayerException
                System.err.println("Error playing media: " +
e.getMessage());

                e.printStackTrace(); // Print the stack trace
                // You can display a dialog box to the user with the
content of the exception here
            }
        } else if (foundMedia instanceof DigitalVideoDisc) {
            DigitalVideoDisc dvd = (DigitalVideoDisc) foundMedia;
            try {
                // Call the play() method on a Media object
                dvd.play();
            } catch (PlayerException e) {
                // Handle the PlayerException
                System.err.println("Error playing media: " +
e.getMessage());

                e.printStackTrace(); // Print the stack trace
                // You can display a dialog box to the user with the
content of the exception here
            }
        }
        storeMenu();
        break;
    case 4:
        cartMenu();

```

```

        break;
    default:
        System.out.println("Error, choose again!");
        storeMenu();
    }
}

public static void mediaDetailsMenu() {
    //method cho xem thông tin của media đã chọn
    int check = 0;
    System.out.println("Options: ");
    System.out.println("-----");
    System.out.println("1. Add to cart");
    if (foundMedia instanceof CompactDisc || foundMedia instanceof
DigitalVideoDisc) {
        System.out.println("2. Play");
        check = 1;
        //Kiểm tra xem media xem có phải loại play được không để in ra lựa
chọn
    }
    System.out.println("0. Back");
    System.out.println("-----");
    System.out.println("Please choose a number: 0-1-(2)");
    int detailMenuChoice = scanner.nextInt();
    switch (detailMenuChoice){
    case 1:
        aCart.addMedia(foundMedia);
        aStore.removeMedia(foundMedia);
        System.out.println("Media added to cart successfully");
        //Thêm media lưu trong biến foundMedia từ trước vào cart rồi bỏ khỏi
store
        storeMenu();
        break;
    case 2:
        if (check == 0)
        {
            System.out.println("Error, choose again!");
            mediaDetailsMenu();
            //Nếu media không phải loại play được thì không có lựa chọn 2, coi
nó như lựa chọn lỗi
        }
        else
        {
            if (foundMedia instanceof CompactDisc) {
                CompactDisc cd = (CompactDisc) foundMedia;

```

```

        try {
            // Call the play() method on a Media object
            cd.play();
        } catch (PlayerException e) {
            // Handle the PlayerException
            System.err.println("Error playing media: " + e.getMessage());
            e.printStackTrace(); // Print the stack trace
            // You can display a dialog box to the user with the content
of the exception here
        }
    } else if (foundMedia instanceof DigitalVideoDisc) {
        DigitalVideoDisc dvd = (DigitalVideoDisc) foundMedia;
        try {
            // Call the play() method on a Media object
            dvd.play();
        } catch (PlayerException e) {
            // Handle the PlayerException
            System.err.println("Error playing media: " + e.getMessage());
            e.printStackTrace(); // Print the stack trace
            // You can display a dialog box to the user with the content
of the exception here
        }
    }
    //Nếu là loại play được thì ép kiểu rồi dùng hàm play tương ứng
}
break;
case 0:
    storeMenu();
    break;
default:
    System.out.println("Error, choose again!");
    mediaDetailsMenu();
}
}

public static void updateStore(){
    //Method cho cập nhật cửa hàng
    Media item;
    // In các lựa chọn
    System.out.println("Options: ");
    System.out.println("-----");
    System.out.println("1. Add new media.");
    System.out.println("2. Remove media.");
    System.out.println("0. Back");
    System.out.println("-----");
    System.out.println("Please choose a number 0-1-2");
}

```



```

    int updateStoreChoice = scanner.nextInt();
    switch (updateStoreChoice){
        case 0: showMenu(); break;
        case 1:
            AddMedia();
            break;
        case 2:
            System.out.println("Removed media");
            scanner.nextLine();
            String removeFromStore = scanner.nextLine();
            //Đọc tên media
            item = aStore.searchByTitle(removeFromStore);
            //Tìm trong cửa hàng media đó
            if (item != null) {
                System.out.println("Media removed successfully");
                aStore.removeMedia(item);
                //nếu có thì loại bỏ và báo
            } else {
                System.out.println("Can't remove");
                //Nếu không báo lỗi
            }
            updateStore();
            break;
        default:
            System.out.println("Error, choose again!");
            updateStore();
    }
}

public static void AddMedia(){
    //Method cho việc thêm media trong store
    String Title;
    //In ra các lựa chọn
    System.out.println("Options: ");
    System.out.println("-----");
    System.out.println("1. Add a book.");
    System.out.println("2. Add a DVD.");
    System.out.println("3. Add a CD.");
    System.out.println("0. Back");
    System.out.println("-----");
    System.out.println("Please choose a number 0-1-2-3");
    int updateStoreChoice = scanner.nextInt();
    scanner.nextLine();
    switch (updateStoreChoice){
        case 0: updateStore(); break;

```

```
        case 1:
            System.out.println("Title");
            Title = scanner.nextLine();
            //Lấy tên của book
            Book book = new Book(Title);
            //Tạo một instance mới rồi thêm vào store
            aStore.addMedia(book);
            updateStore();
            break;
        case 2:
            System.out.println("Title");
            Title = scanner.nextLine();
            //Lấy tên
            DigitalVideoDisc dvd = new DigitalVideoDisc(Title);
            aStore.addMedia(dvd);
            //Tạo một instance mới rồi thêm vào store
            updateStore();
            break;
        case 3:
            System.out.println("Title");
            Title = scanner.nextLine();
            //Lấy tên
            CompactDisc cd = new CompactDisc(Title);
            aStore.addMedia(cd);
            updateStore();
            break;
        default:
            System.out.println("Error, choose again!");
            updateStore();
    }
}

public static void cartMenu() {
    //Method cho xem trong cart có gì
    Media item;
    aCart.showCart();
    //In ra những gì có trong Cart
    System.out.println("Options: ");
    System.out.println("-----");
    System.out.println("1. Filter media in cart");
    System.out.println("2. Sort media in cart");
    System.out.println("3. Remove media from cart");
    System.out.println("4. Play a media");
    System.out.println("5. Place order");
}
```

```
System.out.println("0. Back");
System.out.println("-----");
System.out.println("Please choose a number: 0-1-2-3-4-5");
int cartMenuChoice = scanner.nextInt();
switch (cartMenuChoice){
    case 1:
        filterMenu();
        break;
    case 2:
        sortMenu();
        break;
    case 3:
        System.out.println("Enter media's title: ");
        scanner.nextLine();
        String removeMediaTitle = scanner.nextLine();
        item = aCart.searchByTitle(removeMediaTitle);
        //Lấy tên Media muốn xóa rồi tìm nó
        if (item != null) {
            System.out.println("Media removed successfully");
            aCart.removeMedia(item);
        } else {
            System.out.println("Can't removed");
        }
        //Tìm được thì xóa không thì báo lỗi
        cartMenu();
        break;
    case 4:
        System.out.println("Enter media's title");
        scanner.nextLine();
        String mediaTitlePlay = scanner.nextLine();
        item = aCart.searchByTitle(mediaTitlePlay);
        //Lấy tên rồi tìm media để play
        if (foundMedia instanceof CompactDisc) {
            CompactDisc cd = (CompactDisc) foundMedia;
            try {
                // Call the play() method on a Media object
                cd.play();
            } catch (PlayerException e) {
                // Handle the PlayerException
                System.err.println("Error playing media: " + e.getMessage());
                e.printStackTrace(); // Print the stack trace
                // You can display a dialog box to the user with the content
                // of the exception here
            }
        } else if (foundMedia instanceof DigitalVideoDisc) {
```

```

        DigitalVideoDisc dvd = (DigitalVideoDisc) foundMedia;
        try {
            // Call the play() method on a Media object
            dvd.play();
        } catch (PlayerException e) {
            // Handle the PlayerException
            System.err.println("Error playing media: " + e.getMessage());
            e.printStackTrace(); // Print the stack trace
            // You can display a dialog box to the user with the content
of the exception here
        }
    }

    else {
        System.out.println("Media not found in the store");
    }

    //Nếu play được thì play không thì báo lỗi
    cartMenu();
    break;
case 5:
    System.out.println("An order is created");
    cartMenu();
    break;
case 0:
    storeMenu();
    break;
default:
    System.out.println("Error, choose again");
    cartMenu();
}
}

public static void filterMenu(){
    System.out.println("Options: ");
    System.out.println("-----");
    System.out.println("1. By ID");
    System.out.println("2. By Title");
    System.out.println("0. Back");
    System.out.println("-----");
    System.out.println("Please choose a number: 0-1-2");
    int filterChoice = scanner.nextInt();
    switch (filterChoice){
        case 1:
            System.out.println("Enter media's ID: ");
            int filterID = scanner.nextInt();
            //Lấy Id rồi tìm các id khớp

```

```
        aCart.searchById(filterID);
        filterMenu();
        break;
    case 2:
        System.out.println("Enter media's title: ");
        scanner.nextLine();
        String filterTitle = scanner.nextLine();
        //Lấy title rồi tìm các title khớp
        aCart.searchByTitle(filterTitle);
        filterMenu();
        break;
    case 0:
        cartMenu();
        break;
    default:
        System.out.println("Error, choose again");
        filterMenu();
    }
}

public static void sortMenu(){
    System.out.println("Options: ");
    System.out.println("-----");
    System.out.println("1. By Title");
    System.out.println("2. By Cost");
    System.out.println("0. Back");
    System.out.println("-----");
    System.out.println("Please choose a number: 0-1-2");
    int sortChoice = scanner.nextInt();
    switch (sortChoice){
        case 1:
            aCart.sortByTitle();
            aCart.showCart();
            //Sort bằng title rồi show lại cart
            sortMenu();
            break;
        case 2:
            aCart.sortByCost();
            aCart.showCart();
            //Sortbycost rồi show lại cart
            sortMenu();
            break;
        case 0:
            cartMenu();
            break;
    }
}
```

```
        default:
            System.out.println("Error, choose again");
            sortMenu();
        }
    }
}
```

b) Cart.java

```
package hust.soict.hedspi.aims;
import java.util.ArrayList;
import java.util.List;
import hust.soict.hedspi.aims.media.*;
import javafx.beans.property.FloatProperty;
import javafx.beans.property.SimpleFloatProperty;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
public class Cart {

    public static final int MAX_NUMBERS_ORDERED = 20;
    private ObservableList<Media> itemOrdered =
        FXCollections.observableArrayList();
    //Sử dụng observableList để tự động cập nhật cho giao diện
    private SimpleFloatProperty totalCostProperty = new SimpleFloatProperty(0);
    // Sử dụng SimpleFloatProperty để tự động cập nhật tổng giá trị của hàng
    trong cart
    // Sử dụng list để tiện cho việc remove
    public void addMedia (Media item)
    { // Method cho việc thêm media
        if ( itemOrdered.size() == MAX_NUMBERS_ORDERED)
        { //Nếu kích thước của list = tối đa thì gửi thông báo
            throw new CartException("The cart is full. Cannot add more items.");
        }
        else
        {
            // Nếu không thì thêm media vào cart và thông báo thêm thành công
            itemOrdered.add(item);
            totalCostProperty.set(item.getCost()+totalCostProperty.get());
            //Mỗi khi thêm media thì tăng totalcostproperty lên
            System.out.println("The item has been added");
        }
    }
}
```

```
public void addMedia (Media ... item)
{
    // Method cho việc thêm disc với arg tùy ý
    if ( itemOrdered.size() == MAX_NUMBERS_ORDERED)
    {
        //Nếu kích thước của list = tối đa thì gửi thông báo
        throw new CartException("The cart is full. Cannot add more items.");
    }
    else
    {
        if ( itemOrdered.size() + item.length == MAX_NUMBERS_ORDERED)
        {
            // Nếu cart không đủ chỗ để thêm list thì gửi thông báo
            throw new CartException("There are not enough spots in the
cart");
        }
        else
        {
            for (Media it : item) {
                itemOrdered.add(it);
                totalCostProperty.set(it.getCost()+totalCostProperty.get());
                // Dùng vòng lặp và thêm từ từ item từ list vào
                //Mỗi khi thêm media thì tăng totalcostproperty lên
            }
            System.out.println("The list of items has been added
successfully");
        }
    }
}

public void addMedia (Media item1, Media item2 )
{
    // Method cho việc thêm media với 2 arg
    if ( itemOrdered.size() + 2 > MAX_NUMBERS_ORDERED)
    {
        //Nếu cart không đủ chỗ để chứa thì gửi thông báo thì gửi thông báo
        throw new CartException("The cart is full. Cannot add more items.");
    }
    else
    {
        // Nếu không thì thêm media vào cart và thông báo thêm thành công
        itemOrdered.add(item1);
        itemOrdered.add(item2);
        totalCostProperty.set(item1.getCost()+totalCostProperty.get());
        totalCostProperty.set(item2.getCost()+totalCostProperty.get());
        //Mỗi khi thêm media thì tăng totalcostproperty lên
        System.out.println("Both items have been added");
    }
}
```

```
public void removeMedia (Media item)
{
    // Method cho việc loại bỏ media
    if (itemOrdered.isEmpty())
    {
        //Nếu list trống thì thông báo cart trống
        throw new CartException("The cart is empty. Cannot remove
items.");
    }
    if ( itemOrdered.remove(item))
    {
        // Nếu loại bỏ thành công thì gửi thông báo
        System.out.println("The item is removed successfully");
        totalCostProperty.set(totalCostProperty.get() - item.getCost());
        //Mỗi khi loại bỏ media thì giảm totalcostproperty xuống
    }
    else
    {
        // Nếu không loại bỏ thành công thì gửi thông báo item không có
        trong cart
        throw new CartException("The item is not in the cart.");
    }
}

public void showCart ()
{
    // Method để liệt kê những gì có trong cart và chi phí của chúng
    int i = 1;
    System.out.println("*****CART*****");
    for (Media item : itemOrdered) {
        System.out.println(i + ".item - " + item.toString());
        i++;
    }
    // Duyệt lần lượt qua cart và in kết quả
    System.out.print("Total Cost: ");
    System.out.println(totalCostProperty.get());
    System.out.println("*****");
;
}

public void emptyCart() {
    //Method để làm rỗng giỏ hàng cho khi place order
    itemOrdered.clear();
    totalCostProperty.set(0);
    //total cost về 0
    System.out.println("The cart has been emptied.");
}
```



```
}

public ObservableList<Media> getItemOrdered() {
    return itemOrdered;
}
//Getter để lấy xem trong cart có gì

public SimpleFloatProperty getTotalCostProperty() {
    return totalCostProperty;
}

//Getter để trả lại tổng giá trị

public Media searchByTitle(String Title)
{ // Method để tìm bằng title
    int count = 1;
    Media result = null;
    for (Media item : itemOrdered) {
        if (item.isMatch(Title))
            { // Duyệt lần lượt qua cart và kiểm tra xem title của media có
chứa xâu dùng để tra không
                System.out.println(count + ".item -" + item.toString());
                result = item;
                count ++;
                // nếu có thì in ra và tăng count
            }
    }
    if (count == 1)
    { //Nếu count không đổi thì in ra không tìm được
        System.out.println("No results found");
        return null;
        //trả về null khi không tìm thấy
    }
    return result;
    //trả về kết quả cuối
}

public void searchById(int Id)
{
    int count = 1;
    for (Media item : itemOrdered) {
        if (Id == item.getId())
```

```

        // Duyệt tuần tự qua cart và kiểm tra xem id có trùng không
        System.out.println(count + ".item -" + item.toString());
        count ++;
        //Nếu có thì in ra và tăng count rồi break vì id là duy nhất
        break;
    }

}

if (count == 1)
{
    // nếu count không đổi thì in ra là không tìm được
    System.out.println("No results found");
}

}

public void sortByTitle()
{
    java.util.Collections.sort(itemOrdered, Media.COMPARE_BY_TITLE_COST);
}

public void sortByCost()
{
    java.util.Collections.sort(itemOrdered, Media.COMPARE_BY_COST_TITLE);
}

public class CartException extends RuntimeException {
    public CartException(String message) {
        super(message);
    }
}

}

```

c) Store.java

```

package hust.soict.hedspi.aims;

import java.util.ArrayList;
import java.util.List;

import hust.soict.hedspi.aims.media.Media;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;

```

```
public class Store {
    private ObservableList<Media> itemsInStore =
        FXCollections.observableArrayList();
    //Sử dụng observableList để tự động cập nhập giao diện
    public void addMedia (Media item)
    {    // Method cho việc thêm item
        //Thêm item vào store và thông báo thêm thành công
        itemsInStore.add(item);
        System.out.println("The item has been added");
    }

    public void removeMedia (Media item)
    {    // Method cho việc loại bỏ media
        if (itemsInStore.isEmpty())
        {    //Nếu list trống thì thông báo shop trống
            throw new StoreException("The shop is empty. Cannot remove
items.");
        }
        if (itemsInStore.remove(item))
        {    // Nếu loại bỏ thành công thì gửi thông báo
            System.out.println("The item is removed successfully");
        }
        else
        {    // Nếu không loại bỏ thành công thì gửi thông báo disc không có
trong shop
            throw new StoreException("The item is not in the shop.");
        }
    }

    public void showStore()
    {    // Method để liệt kê những gì có trong store và chi phí của chúng
        int i = 1;
        System.out.println("*****Store*****");
;
        for (Media item : itemsInStore) {
            System.out.println(i + ".item - " + item.toString());
            i++;
            // Duyệt lần lượt qua Store và in kết quả
        }
        System.out.println("*****");
;
    }
}
```

```

public ObservableList<Media> getItemsInStore() {
    return itemsInStore;
}
//Getter trả lại những gì đang có trong cửa hàng

public Media searchByTitle(String Title)
{ // Method để tìm bằng title
    int count = 1;
    Media result = null;
    for (Media item : itemsInStore) {
        if (item.isMatch(Title))
            {// Duyệt lần lượt qua shop và kiểm tra xem title của media có
chứa xâu dùng để tra không
                System.out.println("Item -" + item.toString());
                result = item;
                count ++;
                // nếu có thì in ra và tăng count
            }
    }
    if (count == 1) {
        throw new StoreException("No results found for the given title.");
    }
    return result;
    //trả về kết quả cuối thỏa mãn
}

public class StoreException extends RuntimeException {
    public StoreException(String message) {
        super(message);
    }
}

}

}

```

B. hust.soict.hedspi.aims.media

a) Media.java

b) package hust.soict.hedspi.aims.media;

c)

d) import java.util.Comparator;

e)

f) public abstract class Media {

```
g)     private int id;
h)     private String title;
i)     private String category;
j)     private float cost;
k)     private static int nbMedia = 0;
l)     //Khởi tạo các thuộc tính của Media
m)     public static final Comparator<Media> COMPARE_BY_TITLE_COST =
n)         new MediaComparatorByTitleCost();
o)     public static final Comparator<Media> COMPARE_BY_COST_TITLE =
p)         new MediaComparatorByCostTitle();
q)     //Các thuộc tính comparator
r)     public Media(String title) {
s)         super();
t)         this.title = title;
u)         setId(++nbMedia);
v)     }
w)
x)     public Media(String title, String category, float cost) {
y)         super();
z)
aa)         if (cost < 0) {
bb)             throw new IllegalArgumentException("Cost cannot be
negative.");
cc)         }
dd)
ee)         this.title = title;
ff)         this.category = category;
gg)         this.cost = cost;
hh)         setId(++nbMedia);
ii)     }
jj)     //Các constructor
kk)     public int getId() {
ll)         return id;
mm)     }
nn)
oo)     public void setId(int id) {
pp)         this.id = id;
qq)     }
rr)
ss)     public String getTitle() {
tt)         return title;
uu)     }
vv)
ww)     public void setTitle(String title) {
xx)         this.title = title;
```

```

yy)      }
zz)
aaa)      public String getCategory() {
bbb)          return category;
ccc)      }
ddd)
eee)      public void setCategory(String category) {
fff)          this.category = category;
ggg)      }
hhh)
iii)      public float getCost() {
jjj)          return cost;
kkk)      }
lll)
mmm)      public void setCost(float cost) {
nnn)          this.cost = cost;
ooo)      }
ppp)      // Các getters và setters
qqq)      public boolean equals(Object obj) {
rrr)          //Kiểm tra xem object có phải null hoặc không phải là
1 instance của media
sss)          if (obj == null || !(obj instanceof Media)) {
ttt)              return false;
uuu)          }
vvv)
www)          // ép kiểu object thành media
xxx)          Media otherMedia = (Media) obj;
yyy)
zzz)          try {
aaaa)              // kiểm tra xem title có phải là equal
bbbb)                  return title.equals(otherMedia.title);
cccc)          } catch (NullPointerException e) {
dddd)              // Xử lý NullPointerException
eeee)                  return false;
ffff)          }
gggg)      }
hhhh)
iiii)      public boolean isMatch(String tit) {
jjjj)          //Method để kiểm tra trong title có chứa xâu tit không
(chuyển về lowercase)
kkkk)          return
getTitle().toLowerCase().contains(tit.toLowerCase());
llll)      }
mmmm)
nnnn)      public boolean isMatchID(int id) {

```

```

oooo)          //Method để kiểm tra id có trùng ko
pppp)          return (getId() == id);
qqqq)          }
rrrr)
ssss)          public Media() {
tttt)
uuuu)          }
vvvv)
wwww) }
xxxx)

```

b) Book.java

```

package hust.soict.hedspi.aims.media;
import java.util.ArrayList;
import java.util.List;

public class Book extends Media {
    private List<String> authors = new ArrayList<String>();
    public Book(String title) {
        super(title);
    }
    public Book(String title, String category, float cost) {
        super(title,category,cost);
    }
    //Các constructor cho class
    public void addAuthor(String authorName)
    { // Method cho việc thêm authorName
        if ( authors.contains(authorName))
        { //Kiểm tra xem tên tác giả có trong list chưa
            throw new BookException("The author name is already presented.");
        }
        else
        {
            // Nếu không thì thêm vào và thông báo thêm thành công
            authors.add(authorName);
            System.out.println("Author's name has been added successfully");
        }
    }

    public void removeAuthor(String authorName)
    { // Method cho việc xóa authorName
        if ( authors.contains(authorName))
        { //Kiểm tra xem tên tác giả có trong list chưa

```

```

        authors.remove(authorName);
        throw new BookException("This author's name is not presented, can't
remove it");
    }
    else
    {
        // Nếu không thì báo lỗi
        System.out.println("This author's name is not presented, can't remove
it");
    }
}

public String toString()
{
    // Phương thức chuyển các thông tin của Book thành string rồi trả lại
    String cd;
    cd = getTitle() + " - " + getCategory() + " - " + getCost();
    return cd;
}

public Book() {

}

public class BookException extends RuntimeException {
    // class exception
    public BookException(String message) {
        super(message);
    }
}
}

```

c) Disc.java

```

package hust.soict.hedspi.aims.media;

public class Disc extends Media {
    private String director;
    private int length;
}

```



```
//Các thuộc tính của Disc
public String getDirector() {
    return director;
}
public int getLength() {
    return length;
}
//Các getter
public void setDirector(String director) {
    this.director = director;
}
public void setLength(int length) {
    this.length = length;
}
//Các setter
public Disc(String title) {
    super(title);
}

public Disc(String title, String category, float cost) {
    super(title,category,cost);
}

public Disc(String director, int length) {
    super();
    this.director = director;
    this.length = length;
}

public Disc() {
    super();
}
//Các constructor
}
```

d) Track.java

```
package hust.soict.hedspi.aims.media;

import hust.soict.hedspi.aims.exception.PlayerException;

public class Track implements Playable {
    private String title;
```

```
private int length;

//Các thuộc tính của track
public String getTitle() {
    return title;
}

public int getLength() {
    return length;
}

//Các getter
public Track(String title, int length) {
    super();
    this.title = title;
    this.length = length;
}
//Constructor cho track
public boolean equals(Track o)
{//Override equals, nếu cùng title và length thì trả về true
    if (title.equals(o.title))
    {
        if (length == o.length )
        {
            return true;
        }
        else
        {
            return false;
        }
    }
    else
    {
        return false;
    }
}

public String toString()
{// Phương thức chuyển các thông tin của DVD thành string rồi trả lại
    String track;
    track = getTitle() + " - " +getLength() ;
    return track;
}
```

```

        public StringBuilder play() throws PlayerException {
            // Method play for the Playable interface sử dụng string builder cho
            // tiện
            StringBuilder result = new StringBuilder();
            if (this.getLength() > 0) {
                result.append("Playing Track:");
            }.append(getTitle()).append("\nTrack length: ").append(getLength());
            return result;
        } else {
            throw new PlayerException("ERROR: DVD length is non-positive!");
        }
    }
}

```

e) CompactDisc.java

```

package hust.soict.hedspi.aims.media;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

import hust.soict.hedspi.aims.exception.PlayerException;

public class CompactDisc extends Disc implements Playable {
    private String artist;
    private List<Track> tracks = new ArrayList<Track>();
    //khởi tạo các thuộc tính của dvd
    public CompactDisc(String title) {
        super(title);
    }

    public CompactDisc(String title, String category, float cost) {
        super(title, category, cost);
    }

    public CompactDisc(String director, int length, String artist) {
        super(director, length);
        this.artist = artist;
    }
}

```

```
    public CompactDisc(String title, String category, String director, int
length, float cost, String artist) {
        super(title, category, cost);
        setDirector(director);
        setLength(length);
        setArtist(artist);
    }

    //Các constructor cho CD
    public String getArtist() {
        return artist;
    }
    // Getter để lấy Artist

    public void addTrack(Track trackName)
    {    // Method cho việc thêm track
        if ( tracks.contains(trackName))
            { //Kiểm tra xem track có trong list chưa
                throw new CompactDiscException("The track is already presented.");
            }
        else
        {
            // Nếu không thì thêm vào và thông báo thêm thành công
            tracks.add(trackName);
            System.out.println("Track has been added successfully");
        }
    }

    public void setArtist(String artist) {
        this.artist = artist;
    }

    public void removeTrack(Track trackName)
    {    // Method cho việc xóa track
        if ( tracks.contains(trackName))
            { //Kiểm tra xem tên có trong list chưa
                tracks.remove(trackName);
                System.out.println("The track has been removed successfully");
                return;
            }
        else
        {
```

```
        // Nếu không thì báo
        throw new CompactDiscException("This track is not presented, can't
remove it");
    }
}

public int getLength ()
{    //Method tính length
    int total =0;
    for (Track track : tracks) {
        total += track.getLength();
    }
    // Duyệt tuần tự qua tracks và tính tổng thời gian
    return total;
}

public StringBuilder play() throws PlayerException {
    // method thực hiện interface play() có bắt exception, sử dụng string
builder cho tiện
    StringBuilder result = new StringBuilder();
    result.append("Total tracks: ").append(tracks.size()).append("\n");
    result.append("Total runtime: ").append(getLength()).append("\n");

    if (this.getLength() > 0) {
        // Nếu thời gian > 0 thì không có lỗi
        Iterator<Track> iter = tracks.iterator();
        Track nextTrack;
        while (iter.hasNext()) {
            nextTrack = iter.next();
            try {
                result.append(nextTrack.play()).append("\n");
                // Duyệt tuần tự và chơi từng tracks
            } catch (PlayerException e) {
                // Xử lý exception tạo bởi Track
                System.err.println("Error playing track: " + e.getMessage());
                throw e;
            }
        }
        return result;
    } else {
        throw new PlayerException("ERROR: CD length is non-positive!");
    }
}
```

```

    public String toString()
    { // Phương thức chuyển các thông tin của CD thành string rồi trả lại
      String cd;
      cd = getTitle() + " - " + getCategory() + " - " + getDirector() + "
- " + getArtist() + " - " + getLength();
      return cd;
    }

    public class CompactDiscException extends RuntimeException {
      //method exception tổng quát
      public CompactDiscException(String message) {
        super(message);
      }
    }
}

```

f) DigitalVideoDisc.java

```

package hust.soict.hedspi.aims.media;

import hust.soict.hedspi.aims.exception.PlayerException;

public class DigitalVideoDisc extends Disc implements Playable {

    // Khởi tạo thuộc tính của DigitalVideoDisc
    public DigitalVideoDisc(String title) {
        super(title);
    }

    public DigitalVideoDisc(String title, String category, float cost) {
        super(title, category, cost);
    }

    public DigitalVideoDisc(String title, String category, String director, float
cost) {
        super(title, category, cost);
        setDirector(director);
    }

    public DigitalVideoDisc(String title, String category, String director, int
length, float cost) {
        super(title, category, cost);
        setDirector(director);
        setLength(length);
    }
}

```

```

    }
    //Các phương thức constructor cho DVD, dùng super tới Media
    public String toString() {
        // Phương thức chuyển các thông tin của DVD thành string rồi trả lại
        String dvd = getTitle() + " - " + getCategory() + " - " +
            getDirector() + " - " + getLength() + " - " + getCost();
        return dvd;
    }

    public StringBuilder play() throws PlayerException {
        // Method play cho Playable interface
        StringBuilder result = new StringBuilder();
        if (this.getLength() > 0) {
            //Nếu length > thì không có lỗi play như thường
            result.append("Playing DVD: ").append(getTitle()).append("\nDVD
length: ").append(getLength());
            return result;
        } else {
            throw new PlayerException("ERROR: DVD length is non-positive!");
        }
    }
}
}

```

g) MediaComparatorByCostTitle.java

```

package hust.soict.hedspi.aims.media;

import java.util.Comparator;

public class MediaComparatorByCostTitle implements Comparator<Media> {
    //Class Comparator theo cost rồi đến title
    private Comparator<Media> costComparator =
        Comparator.comparingDouble(Media::getCost);
    private Comparator<Media> titleComparator =
        Comparator.comparing(Media::getTitle);
    // Dùng hàm comparing theo cost và title
    private Comparator<Media> compoundComparator =
        costComparator.thenComparing(titleComparator);
    //Dùng thenComparing để compare theo cost rồi đến title

    public int compare(Media media1, Media media2) {
        return compoundComparator.compare(media1, media2);
    }
}

```

h) MediaComparatorByTitleCost.java

```

package hust.soict.hedspi.aims.media;

import java.util.Comparator;

public class MediaComparatorByTitleCost implements Comparator<Media> {
    //Class Comparator theo title rồi đến cost
    private Comparator<Media> costComparator =
        Comparator.comparingDouble(Media::getCost);
    private Comparator<Media> titleComparator =
        Comparator.comparing(Media::getTitle);
    // Dùng hàm comparing theo cost và title
    private Comparator<Media> compoundComparator =
        titleComparator.thenComparing(costComparator);
    //Dùng thenComparing để compare theo cost rồi đến title

    public int compare(Media media1, Media media2) {
        return compoundComparator.compare(media1, media2);
    }
}

```

i) Playable.java

```

package hust.soict.hedspi.aims.media;

import hust.soict.hedspi.aims.exception.PlayerException;

public interface Playable {
    StringBuilder play() throws PlayerException;
    //interface Playable
}

```

C. hust.soict.hedspi.screen
 a) StoreScreen.java

```

package hust.soict.hedspi.aims.screen;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;

import javax.swing.*;

import hust.soict.hedspi.aims.Cart;
import hust.soict.hedspi.aims.Store;
import hust.soict.hedspi.aims.media.Media;

```



```
import hust.soict.hedspi.aims.screen.view.AddBookScreen;
import hust.soict.hedspi.aims.screen.view.AddCDScreen;
import hust.soict.hedspi.aims.screen.view.AddDVDScreen;
import hust.soict.hedspi.aims.screen.view.CartScreen;
import javafx.collections.ObservableList;

public class StoreScreen extends JFrame {
    private Store store;
    private Cart cart;
    // khai báo các attribute cần thiết là store và cart
    public Store getStore() {
        return store;
    }

    public Cart getCart() {
        return cart;
    }
    //Getter lấy các attributes
    JPanel createNorth() {
        JPanel north = new JPanel();
        north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
        north.add(createMenuBar());
        north.add(createHeader());
        return north;
        //Xây dựng phần north của giao diện
    }

    JMenuBar createMenuBar() {
        //Tạo menubar với các lựa chọn
        ButtonListener btnListener = new ButtonListener();
        JMenu menu = new JMenu("Options");

        JMenu smUpdateStore = new JMenu("Update Store");
        JMenuItem addBook = new JMenuItem("Add Book");

        smUpdateStore.add(addBook);
        addBook.addActionListener(btnListener);
        //Thêm actionlistener để chuyển sang screen addbook

        JMenuItem addCD = new JMenuItem("Add CD");

        smUpdateStore.add(addCD);
        addCD.addActionListener(btnListener);
    }
}
```

```
//Thêm actionlistener để chuyển sang screen addcd
JMenuItem addDVD = new JMenuItem("Add DVD");

smUpdateStore.add(addDVD);
addDVD.addActionListener(btnListener);
//Thêm actionlistener để chuyển sang screen adddv

menu.add(smUpdateStore);
menu.add(new JMenuItem("View store"));
JMenuItem cart = new JMenuItem("View cart");

menu.add(cart);
cart.addActionListener(btnListener);
//Thêm actionlistener để chuyển sang screen view cart
JMenuBar menuBar = new JMenuBar();
menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
menuBar.add(menu);

return menuBar;
}

JPanel createHeader() {
    //Xây dựng phần header
    ButtonListener btnListener = new ButtonListener();
    JPanel header = new JPanel();
    header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));

    JLabel title = new JLabel("AIMS");
    title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
    title.setForeground(Color.CYAN);
    //Thiết kế tiêu đề
    JButton cart = new JButton("View cart");
    cart.addActionListener(btnListener);
    //Tạo nút view cart và add listener
    cart.setPreferredSize(new Dimension(100, 50));
    cart.setMaximumSize(new Dimension(100, 50));

    header.add(Box.createRigidArea(new Dimension(10, 10)));
    header.add(title);
    header.add(Box.createHorizontalGlue());
    header.add(cart);
    header.add(Box.createRigidArea(new Dimension(10, 10)));
    return header;
}
```

```
JPanel createCenter() {
    //Tạo phần trung tâm
    JPanel center = new JPanel();

    ObservableList<Media> mediaInStore = store.getItemsInStore();
    int size = mediaInStore.size();
    //Lấy các item đang có trong store
    // Lấy căn bậc hai của số lượng item đang có và + 1
    int rows = (int) Math.sqrt(size) + 1;
    //Ta được số hàng và cột
    center.setLayout(new GridLayout(rows, rows, 2, 2));
    //Tạo gridlayout theo số hàng và cột
    for (int i = 0; i < size; i++) {
        MediaStore cell = new MediaStore(mediaInStore.get(i));
        cell.setStoreScreen(this);
        center.add(cell);
        //Thêm từng item vào cell
    }

    return center;
}

public StoreScreen (Store store, Cart cart) {
    this.store = store;
    this.cart = cart;
    Container cp = getContentPane();
    cp.setLayout(new BorderLayout());

    cp.add(createNorth(), BorderLayout.NORTH);
    cp.add(createCenter(), BorderLayout.CENTER);

    setVisible(true);
    setTitle("Store");
    setSize(1024, 768);
}

// constructor khởi tạo storescreen
private class ButtonListener implements ActionListener{
    //Method thực hiện actionlistener
    @Override
    public void actionPerformed(ActionEvent e) {
        String button = e.getActionCommand();
        if (button.equals("View cart"))
        {
            CartScreen aCart = new CartScreen(cart,store);
            dispose();
        }
    }
}
```

```
        //Nếu là View cart chuyển sang screen cart
    }
    if (button.equals("Add Book"))
    {
        AddBookScreen addBook = new AddBookScreen(cart,store);

        // Nếu là add book chuyển sang screen add book
    }
    if (button.equals("Add DVD"))
    {
        AddDVDScreen addDVD = new AddDVDScreen(cart,store);

        //Nếu là add dvd chuyển sang screen add dvd
    }

    if (button.equals("Add CD"))
    {
        AddCDScreen addCD = new AddCDScreen(cart,store);
        //Nếu là addcd chuyển sang screen addcd
    }

    }
}
}
```

b) MediaStore.java

```
package hust.soict.hedspi.aims.screen;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.*;

import hust.soict.hedspi.aims.media.*;
```

```

public class MediaStore extends JPanel{
    private StoreScreen storeScreen;
    private Media media;
    //Các attributes
    public void setStoreScreen(StoreScreen storeScreen) {
        this.storeScreen = storeScreen;
    }
    //setter
    public MediaStore (Media media){
        ButtonListener btnListener = new ButtonListener();
        this.media = media;
        this.setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));

        JLabel title = new JLabel(media.getTitle());
        title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 20));
        title.setAlignmentX(CENTER_ALIGNMENT);

        JLabel cost = new JLabel(""+media.getCost()+" $");
        cost.setAlignmentX(CENTER_ALIGNMENT);
        //Xây dựng giao diện
        JPanel container = new JPanel(); container.setLayout(new
FlowLayout(FlowLayout.CENTER));
        JButton btnAdd = new JButton("Add to cart");
        btnAdd.addActionListener(btnListener);
        container.add(btnAdd);
        if(media instanceof Playable) {
            // nếu media thực hiện playable thì làm xuất hiện nút play
            JButton btnPlay = new JButton("Play");
            btnPlay.addActionListener(btnListener);
            container.add(btnPlay);
        }
        //Thêm các button và listener
        this.add(Box.createVerticalGlue());
        this.add(title);
        this.add(cost);
        this.add(Box.createVerticalGlue());
        this.add(container);

        this.setBorder(BorderFactory.createLineBorder(Color.BLACK));
    }

    private class ButtonListener implements ActionListener{
        @Override
        //method thực hiện actionlistener
        public void actionPerformed(ActionEvent e) {

```

```

        String button = e.getActionCommand();
        if (button.equals("Add to cart"))
        {
            storeScreen.getCart().addMedia(media);
            //Nếu là add to cart thì thêm vào cart
        }
        if (button.equals("Play"))
        {
            MediaPlayerDialog playDialog = new MediaPlayerDialog((JFrame)
SwingUtilities.getWindowAncestor(MediaStore.this), true, media);
            playDialog.setVisible(true);
            //Nếu là play thì gọi một cửa sổ mới và chạy
        }
    }
}
}

```

c) MediaPlayerDialog.java(class để thực hiện giao diện việc play cũng như báo lỗi)

```

package hust.soict.hedspi.aims.screen;

import java.awt.Font;

import javax.swing.*;

import hust.soict.hedspi.aims.exception.PlayerException;
import hust.soict.hedspi.aims.media.CompactDisc;
import hust.soict.hedspi.aims.media.DigitalVideoDisc;
import hust.soict.hedspi.aims.media.Media;

public class MediaPlayerDialog extends JDialog {
    /**
     *
     */
    private static final long serialVersionUID = 1L;

    public MediaPlayerDialog(JFrame parent, boolean modal, Media media) {
        super(parent, modal);
        //Sử dụng stringBuilder để lấy kết quả của việc play
        StringBuilder playResult = new StringBuilder();
        JLabel title = new JLabel(media.getTitle());
        title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 20));
        title.setAlignmentX(CENTER_ALIGNMENT);
    }
}

```

```
        if (media instanceof CompactDisc) {
            CompactDisc cd = (CompactDisc) media;
            try {
                // nếu media là playable thì ép kiểu và chạy play rồi gán kết quả
                // vào playResult
                playResult.append(cd.play().toString());
                //

            } catch (PlayerException e) {
                // Xử lý PlayerException
                System.err.println( e.getMessage());
                playResult.append(e.getMessage());
                e.printStackTrace();
                // Đặt tiêu đề là lỗi nếu khi play lỗi
                setTitle("Error: " + e.getClass().getSimpleName());
            }
        } else if (media instanceof DigitalVideoDisc) {
            DigitalVideoDisc dvd = (DigitalVideoDisc) media;
            try {
                // // nếu media là playable thì ép kiểu và chạy play rồi gán kết
                // quả vào playResult
                playResult.append(dvd.play().toString());

            } catch (PlayerException e) {
                // Xử lý PlayerException
                System.err.println(e.getMessage());
                playResult.append(e.getMessage());
                e.printStackTrace();
                // Đặt tiêu đề là lỗi nếu khi play lỗi
                setTitle("Error: " + e.getClass().getSimpleName());
            }
        }
        JTextArea playResultArea = new JTextArea(playResult.toString());
        playResultArea.setEditable(false);
        playResultArea.setLineWrap(true);
        playResultArea.setWrapStyleWord(true);

        JScrollPane scrollPane = new JScrollPane(playResultArea);

        getContentPane().setLayout(new BorderLayout(getContentPane(),
        BorderLayout.Y_AXIS));
        getContentPane().add(title);
        getContentPane().add(scrollPane);
```

```

        // Nếu tiêu đề chưa có thì đặt là media player
        if (getTitle() == null) {
            setTitle("Media Player");
        }

        setSize(400, 300);
        setLocationRelativeTo(parent);
        setDefaultCloseOperation(DISPOSE_ON_CLOSE);

    }
}

```

D. hust.soict.hedspi.aims.screen.controller
a) CartScreenController.java

```

package hust.soict.hedspi.aims.screen.controller;

import java.util.function.Predicate;

import javax.swing.JFrame;
import javax.swing.SwingUtilities;

import hust.soict.hedspi.aims.*;
import hust.soict.hedspi.aims.media.Media;
import hust.soict.hedspi.aims.media.Playable;
import hust.soict.hedspi.aims.screen.MediaPlayerDialog;
import hust.soict.hedspi.aims.screen.StoreScreen;
import javafx.beans.binding.Bindings;
import javafx.beans.value.ChangeListener;
import javafx.beans.value.ObservableValue;
import javafx.collections.transformation.FilteredList;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Alert;
import javafx.scene.control.Alert.AlertType;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.RadioButton;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.TextField;
import javafx.scene.control.ToggleGroup;

```



```
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.paint.Color;
import javafx.stage.Stage;

public class CartScreenController {

    private Cart cart;
    private Store store;

    @FXML
    private Button btnPlay;

    @FXML
    private Button btnRemove;

    @FXML
    private TableView<Media> tblMedia;

    @FXML
    private TableColumn<Media, String> colMediaTitle;

    @FXML
    private TableColumn<Media, String> colMediaCategory;

    @FXML
    private TableColumn<Media, Float> colMediaCost;

    @FXML
    private RadioButton radioBtnFilterId;

    @FXML
    private RadioButton radioBtnFilterTitle;

    @FXML
    private TextField tfFilter;

    private FilteredList<Media> filteredMediaList;

    @FXML
    private Label totalCost;

    @FXML
    private Button placeOrder;

    //Các thuộc tính
```

```

public CartScreenController(Cart cart, Store store) {
    super();
    this.cart = cart;
    this.store = store;
    this.filteredMediaList = new FilteredList<>(cart.getItemOrdered());
}
//Constructor
@FXML
private void initialize() {

    colMediaTitle.setCellValueFactory(
        new PropertyValueFactory<Media, String>("title"));
    colMediaCategory.setCellValueFactory(
        new PropertyValueFactory<Media, String>("category"));
    colMediaCost.setCellValueFactory(
        new PropertyValueFactory<Media, Float>("cost"));
    tblMedia.setItems(this.cart.getItemOrdered());
    //Lấy các item đang có trong giỏ hàng
    btnPlay.setVisible(false);
    btnRemove.setVisible(false);

    tblMedia.getSelectionModel().selectedItemProperty().addListener(
        new ChangeListener<Media>() {
            @Override
            public void changed(ObservableValue<? extends Media> observable,
Media oldValue,
                Media newValue) {
                if(newValue!=null) {
                    //Tùy xem chọn item nào thì cập nhật các nút hiển thị
                    updateButtonBar(newValue);
                }
            }
        });

    ToggleGroup toggleGroup = new ToggleGroup();
    radioBtnFilterId.setToggleGroup(toggleGroup);
    radioBtnFilterTitle.setToggleGroup(toggleGroup);
    //Tạo nhóm toggle cho nút Id và Title
    radioBtnFilterTitle.setSelected(true);
    //Mặc định là Title
    tfFilter.textProperty().addListener((ObservableValue<? extends String>
observable, String oldValue,
        String newValue) -> {
        if (toggleGroup.getSelectedToggle() == radioBtnFilterTitle) {

```

```

        //Nếu chọn Title thì sử dụng filter tương ứng
        showFilteredMedia(item -> item.isMatch(newValue.toLowerCase()));
    } else if (toggleGroup.getSelectedToggle() == radioBtnFilterId) {
        try {
            // Không thì chuyển string thành int để filter id
            int id = Integer.parseInt(newValue);
            showFilteredMedia(item -> item.isMatchID(id));
        } catch (NumberFormatException e) {
            System.out.println("Invalid ID format");
        }
    }
});

totalCost.textProperty().bind(Bindings.createStringBinding(() ->
String.format("%.2f$", cart.getTotalCostProperty().get()),
cart.getTotalCostProperty()));
//Binding cost để giá trị trên giao diện tự cập nhật
}

@FXML
void placeOrderPressed(ActionEvent event) {
    // Khi nhấn place order thì có thông báo đã order thành công
    Alert alert = new Alert(AlertType.INFORMATION);
    alert.setTitle("Order Placed");
    alert.setHeaderText(null);
    alert.setContentText("Your order has been placed successfully!");
    alert.showAndWait();
    cart.emptyCart();
}

@FXML
private void showFilteredMedia(Predicate<Media> filter) {
    //Hàm thực hiện việc filter
    FilteredList<Media> filteredList = new
FilteredList<>(cart.getItemOrdered(), filter);
    tblMedia.setItems(filteredList);
}

@FXML
void btnRemovePressed(ActionEvent event) {
    Media media = tblMedia.getSelectionModel().getSelectedItem();
    cart.removeMedia(media);
    //Nếu ấn nút remove thì bỏ media khỏi cart
}

```

```

    void updateButtonBar(Media media) {
        //Nếu chọn media có thể play thì hiển thị nút play
        btnRemove.setVisible(true);
        if (media instanceof Playable) {
            btnPlay.setVisible(true);
        } else {
            btnPlay.setVisible(false);
        }
    }
    @FXML
    void playMedia(ActionEvent event) {
        MediaPlayerDialog playDialog = new MediaPlayerDialog((JFrame) null, true,
            tblMedia.getSelectionModel().getSelectedItem());
        playDialog.setVisible(true);
        //Khi play media gọi class hiển thị mediaplayerdialog
    }

    @FXML
    void viewStore(ActionEvent event) {

        // quay lại store
        StoreScreen screen = new StoreScreen(store, cart);
    }

}

```

b) AddBookScreenController

```

package hust.soict.hedspi.aims.screen.controller;

import java.util.ArrayList;
import java.util.List;

import hust.soict.hedspi.aims.Cart;
import hust.soict.hedspi.aims.Store;
import hust.soict.hedspi.aims.media.Book;
import hust.soict.hedspi.aims.screen.StoreScreen;
import javafx.collections.transformation.FilteredList;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Alert;
import javafx.scene.control.TextField;

```

```
import javafx.scene.layout.VBox;

public class AddBookScreenController {

    private Store store;
    private Cart cart;
    @FXML
    private TextField Category;

    @FXML
    private TextField Cost;

    @FXML
    private TextField Title;

    @FXML
    private TextField author1;

    private List<TextField> authorFields = new ArrayList<>();

    @FXML
    private VBox authorContainer;

    //Các attribute
    public AddBookScreenController(Cart cart, Store store) {
        super();
        this.store = store;
        this.cart = cart;
    }
    //Constructor
    @FXML
    void addAuthor(ActionEvent event) {
        //Thực hiện khi muốn thêm nhiều tác giả cho 1 book, tạo thêm các text
        field để nhập thêm
        TextField newAuthorField = new TextField();
        newAuthorField.setPrefWidth(250.0);
        authorContainer.getChildren().add(newAuthorField);
        authorFields.add(newAuthorField);
    }

    @FXML
    void addBook(ActionEvent event) {
        //Nhấn nút để addbook vào store
        if (Title.getText().equals("")) {
            // Kiểm tra xem có hợp lệ
        }
    }
}
```

```
        Alert alert = new Alert(Alert.AlertType.WARNING);
        alert.setTitle("Warning");
        alert.setHeaderText(null);
        alert.setContentText("Please enter a title for the book.");
        alert.showAndWait();
        return; // dừng thực hiện và báo lỗi nếu sách ko có title
    }

    float cost;
    try {
        cost = Float.parseFloat(Cost.getText());
    } catch (NumberFormatException e) {
        // Báo lỗi nếu cost không hợp lệ
        Alert alert = new Alert(Alert.AlertType.WARNING);
        alert.setTitle("Warning");
        alert.setHeaderText(null);
        alert.setContentText("Please enter a valid cost for the book.");
        alert.showAndWait();
        return;
    }
    Book book = new Book(Title.getText(), Category.getText(), cost);
    book.addAuthor(author1.getText());
    for (TextField author : authorFields)
    {
        book.addAuthor(author.getText());
        //Duyệt tuần tự qua các tác giả và thêm vào
    }
    store.addMedia(book);

    // Thông báo thành công
    showSuccessAlert("Book added successfully.");

    // Reset mọi thứ lại như cũ sau khi add xong
    resetFields();
}

private void showSuccessAlert(String message) {
    //Method thông báo add thành công
    Alert alert = new Alert(Alert.AlertType.INFORMATION);
    alert.setTitle("Success");
    alert.setHeaderText(null);
    alert.setContentText(message);
    alert.showAndWait();
}
```

```
private void resetFields() {
    //Method reset
    Title.clear();
    Category.clear();
    Cost.clear();
    author1.clear();
    authorContainer.getChildren().clear();
    authorFields.clear();

    TextField firstAuthorField = new TextField();
    firstAuthorField.setPrefWidth(250.0);
    authorContainer.getChildren().add(firstAuthorField);
    authorFields.add(firstAuthorField);
}

@FXML
void viewStore(ActionEvent event) {
    StoreScreen Screen = new StoreScreen(store, cart);
    //Quay lại store
}
}
```

C) AddCDScreenController.java

```
package hust.soict.hedspi.aims.screen.controller;

import java.util.ArrayList;
import java.util.List;

import hust.soict.hedspi.aims.Cart;
import hust.soict.hedspi.aims.Store;
import hust.soict.hedspi.aims.media.CompactDisc;
import hust.soict.hedspi.aims.media.Track;
import hust.soict.hedspi.aims.screen.StoreScreen;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.geometry.Insets;
import javafx.scene.control.Alert;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
import javafx.scene.layout.HBox;
```

```
import javafx.scene.layout.VBox;
import javafx.scene.text.Font;

public class AddCDScreenController {

    private Store store;
    private Cart cart;

    @FXML
    private TextField Category;

    @FXML
    private TextField Cost;

    @FXML
    private TextField Director;

    @FXML
    private TextField Length;

    @FXML
    private TextField Title;

    @FXML
    private TextField Artist;

    @FXML
    private TextField Track1;

    @FXML
    private VBox trackContainer;

    @FXML
    private TextField Length1;

    @FXML
    private VBox lengthContainer;

    @FXML
    private Label header1;

    @FXML
    private Label header2;
    //Các attribute
    public AddCDScreenController(Cart cart, Store store) {
```



```

        super();
        this.store = store;
        this.cart = cart;
    }
    //Constructor
    private List<TextField> trackFields = new ArrayList<>();
    private List<TextField> lengthFields = new ArrayList<>();
    //List lưu trữ các text field để điền nhiều tracks
    @FXML
    void addTrack(ActionEvent event) {
        // Tạo track title field mới để thêm track
        TextField newTrackField = new TextField();
        newTrackField.setPrefWidth(250.0);
        VBox.setMargin(newTrackField, new Insets(0, 0, 0, 35.0));
        trackContainer.getChildren().add(newTrackField);
        trackFields.add(newTrackField);

        // Tạo length field mới để thêm length
        TextField newLengthField = new TextField();
        newLengthField.setPrefWidth(40.0);
        VBox.setMargin(newLengthField, new Insets(0, 0, 0, 35.0));
        lengthContainer.getChildren().add(newLengthField);
        lengthFields.add(newLengthField);
    }

    @FXML
    void addCD(ActionEvent event) {
        //Nhấn nút để addcd vào store
        if (Title.getText().equals("")) {
            // Kiểm tra xem có hợp lệ
            Alert alert = new Alert(Alert.AlertType.WARNING);
            alert.setTitle("Warning");
            alert.setHeaderText(null);
            alert.setContentText("Please enter a title for the cd.");
            alert.showAndWait();
            return; // dừng thực hiện và báo lỗi nếu sách ko có title
        }

        float cost;
        try {
            cost = Float.parseFloat(Cost.getText());
        } catch (NumberFormatException e) {
            // Báo lỗi nếu cost không hợp lệ
            Alert alert = new Alert(Alert.AlertType.WARNING);
            alert.setTitle("Warning");

```

```
        alert.setHeaderText(null);
        alert.setContentText("Please enter a valid cost for the cd.");
        alert.showAndWait();
        return;
    }

    int length;
    try {
        length = Integer.parseInt(Length.getText());
    } catch (NumberFormatException e) {
        // Báo lỗi nếu length không hợp lệ
        Alert alert = new Alert(Alert.AlertType.WARNING);
        alert.setTitle("Warning");
        alert.setHeaderText(null);
        alert.setContentText("Please enter a valid length for the cd.");
        alert.showAndWait();
        return;
    }
    CompactDisc cd = new CompactDisc(Title.getText(), Category.getText(),
Director.getText(), length, cost,
        Artist.getText());

    Track firstTrack = new Track(Track1.getText(),
Integer.parseInt(Length1.getText()));
    cd.addTrack(firstTrack);

    int i = 0;
    for (TextField track : trackFields) {
        Track trackAdd = new Track(track.getText(),
Integer.parseInt(lengthFields.get(i).getText()));
        cd.addTrack(trackAdd);
        i++;
        //Duyệt tuần tự qua các track và thêm vào cd
    }

    store.addMedia(cd);
    // thêm cd vào cửa hàng
    // báo thành công
    showSuccessAlert("CD added successfully.");

    // reset về ban đầu
    resetFields();
}

private void showSuccessAlert(String message) {
```

```
//Method báo thành công
Alert alert = new Alert(Alert.AlertType.INFORMATION);
alert.setTitle("Success");
alert.setHeaderText(null);
alert.setContentText(message);
alert.showAndWait();
}

private void resetFields() {
    //Method reset sau khi add
    Title.clear();
    Category.clear();
    Cost.clear();
    Director.clear();
    Length.clear();
    Artist.clear();
    Track1.clear();
    Length1.clear();
    trackContainer.getChildren().clear();
    lengthContainer.getChildren().clear();
    trackFields.clear();
    lengthFields.clear();

    // tạo lại header labels
    header1 = new Label("Title");
    header1.setTextFill(javafx.scene.paint.Color.web("#9e0f3f"));
    header1.setFont(new Font(14.0));
    header1.setPadding(new Insets(35.0, 0, 0, 35.0));

    header2 = new Label("Length");
    header2.setTextFill(javafx.scene.paint.Color.web("#9e0f3f"));
    header2.setFont(new Font(14.0));
    header2.setPadding(new Insets(35.0, 0, 0, 35.0));

    trackContainer.getChildren().add(header1);
    lengthContainer.getChildren().add(header2);

    // thêm first track và length fields lại
    TextField newTrackField = new TextField();
    newTrackField.setPrefWidth(250.0);
    VBox.setMargin(newTrackField, new Insets(0, 0, 0, 35.0));
    trackContainer.getChildren().add(newTrackField);
    trackFields.add(newTrackField);

    TextField newLengthField = new TextField();
```

```

        newLengthField.setPrefWidth(40.0);
        VBox.setMargin(newLengthField, new Insets(0, 0, 0, 35.0));
        lengthContainer.getChildren().add(newLengthField);
        lengthFields.add(newLengthField);
    }

    @FXML
    void viewStore(ActionEvent event) {
        StoreScreen Screen = new StoreScreen(store, cart);

    }
}

```

D) AddDVDScreenController

```

package hust.soict.hedspi.aims.screen.controller;

import hust.soict.hedspi.aims.Cart;
import hust.soict.hedspi.aims.Store;
import hust.soict.hedspi.aims.media.DigitalVideoDisc;
import hust.soict.hedspi.aims.screen.StoreScreen;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Alert;
import javafx.scene.control.TextField;

public class AddDVDScreenController {

    private Store store;
    private Cart cart;

    @FXML
    private TextField Category;

    @FXML
    private TextField Cost;

    @FXML
    private TextField Director;

    @FXML
    private TextField Length;
}

```

```
@FXML
private TextField Title;

//Các attribute
public AddDVDScreenController(Cart cart, Store store) {
    super();
    this.store = store;
    this.cart = cart;
}
//Constructor
@FXML
void addDVD(ActionEvent event) {
    //Nhấn nút để adddvd vào store
    if (Title.getText().equals("")) {
        // Kiểm tra xem có hợp lệ
        Alert alert = new Alert(Alert.AlertType.WARNING);
        alert.setTitle("Warning");
        alert.setHeaderText(null);
        alert.setContentText("Please enter a title for the dvd.");
        alert.showAndWait();
        return;
    }

    float cost;
    try {
        cost = Float.parseFloat(Cost.getText());
    } catch (NumberFormatException e) {
        // Báo lỗi nếu cost không hợp lệ
        Alert alert = new Alert(Alert.AlertType.WARNING);
        alert.setTitle("Warning");
        alert.setHeaderText(null);
        alert.setContentText("Please enter a valid cost for the dvd.");
        alert.showAndWait();
        return;
    }

    int length;
    try {
        length = Integer.parseInt(Length.getText());
    } catch (NumberFormatException e) {
        // Báo lỗi nếu cost không hợp lệ
        Alert alert = new Alert(Alert.AlertType.WARNING);
        alert.setTitle("Warning");
```

```

        alert.setHeaderText(null);
        alert.setContentText("Please enter a valid length for the dvd.");
        alert.showAndWait();
        return;
    }
    DigitalVideoDisc dvd = new DigitalVideoDisc(Title.getText(),
        Category.getText(), Director.getText(), length, cost );
    store.addMedia(dvd);
    // báo thành công
    showSuccessAlert("DVD added successfully.");

    // reset về ban đầu
    resetFields();
}

private void showSuccessAlert(String message) {
    //Method báo thành công
    Alert alert = new Alert(Alert.AlertType.INFORMATION);
    alert.setTitle("Success");
    alert.setHeaderText(null);
    alert.setContentText(message);
    alert.showAndWait();
}

private void resetFields() {
    //Method reset sau khi add
    Title.clear();
    Category.clear();
    Cost.clear();
    Director.clear();
    Length.clear();
}

@FXML
void viewStore(ActionEvent event) {
    StoreScreen Screen = new StoreScreen(store, cart);
}
}

```

E. hust.soict.hedspi.aims.screen.view
a) AddBookScreen

```
package hust.soict.hedspi.aims.screen.view;

import java.awt.Dimension;
import java.io.IOException;

import javax.swing.JFrame;
import javax.swing.WindowConstants;

import hust.soict.hedspi.aims.Cart;
import hust.soict.hedspi.aims.Store;
import hust.soict.hedspi.aims.screen.controller.AddBookScreenController;
import javafx.application.Platform;
import javafx.embed.swing.JFXPanel;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;

public class AddBookScreen extends JFrame{

    private Store store;
    private Cart cart;
    public AddBookScreen (Cart cart, Store store) {
        super();
        this.store = store;
        this.cart = cart;

        JFXPanel fxPanel = new JFXPanel();
        this.add(fxPanel);

        this.setTitle("AddingBook");
        this.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);

        // Mở rộng frame khi khởi tạo
        this.setSize(new Dimension(1024, 768));
        this.setVisible(true);
        Platform.runLater(new Runnable() {
            @Override
            public void run() {
                try {
                    //Lấy file fxml và chạy
                    FXMLLoader loader = new FXMLLoader(getClass()
                        .getResource("/hust/soict/hedspi/aims/screen/view/AddBook
                            .fxml"));
                    AddBookScreenController controller =
                        new AddBookScreenController(cart, store);
```

```

        loader.setController(controller);
        Parent root = loader.load();
        fxPanel.setScene(new Scene(root));

    } catch (IOException e) {
        e.printStackTrace();
    }

    }
});
}
}

```

b) AddCDScreen.java

```

package hust.soict.hedspi.aims.screen.view;

import java.awt.Dimension;
import java.io.IOException;

import javax.swing.JFrame;
import javax.swing.WindowConstants;

import hust.soict.hedspi.aims.Cart;
import hust.soict.hedspi.aims.Store;
import hust.soict.hedspi.aims.screen.controller.AddCDScreenController;
import javafx.application.Platform;
import javafx.embed.swing.JFXPanel;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;

public class AddCDScreen extends JFrame{

    private Store store;
    private Cart cart;
    public AddCDScreen (Cart cart, Store store) {
        super();
        this.store = store;
        this.cart = cart;

        JFXPanel fxPanel = new JFXPanel();
        this.add(fxPanel);

        this.setTitle("AddingCD");
    }
}

```



```

        this.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);

// Mở rộng frame khi khởi tạo
        this.setSize(new Dimension(1024, 768));
        this.setVisible(true);
        Platform.runLater(new Runnable() {
            @Override
            public void run() {
                try {
                    //Lấy file fxml và chạy
                    FXMLLoader loader = new FXMLLoader(getClass()
                        .getResource("/hust/soict/hedsapi/aims/screen/view/AddCD.f
xml"));

                    AddCDScreenController controller =
                        new AddCDScreenController(cart, store);
                    loader.setController(controller);
                    Parent root = loader.load();
                    fxPanel.setScene(new Scene(root));

                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

c) AddDVDScreen

```

package hust.soict.hedsapi.aims.screen.view;

import java.awt.Dimension;
import java.io.IOException;

import javax.swing.JFrame;
import javax.swing.WindowConstants;

import hust.soict.hedsapi.aims.Cart;
import hust.soict.hedsapi.aims.Store;
import hust.soict.hedsapi.aims.screen.controller.AddDVDScreenController;
import javafx.application.Platform;
import javafx.embed.swing.JFXPanel;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;

```

```

import javafx.scene.Scene;

public class AddDVDScreen extends JFrame{

    private Store store;
    private Cart cart;
    public AddDVDScreen (Cart cart, Store store) {
        super();
        this.store = store;
        this.cart = cart;

        JFXPanel fxPanel = new JFXPanel();
        this.add(fxPanel);

        this.setTitle("AddingDVD");
        this.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);

        // Mở rộng frame khi khởi tạo
        this.setSize(new Dimension(1024, 768));
        this.setVisible(true);
        Platform.runLater(new Runnable() {
            @Override
            public void run() {
                try {
                    //Lấy file fxml và chạy
                    FXMLLoader loader = new FXMLLoader(getClass()
                        .getResource("/hust/soict/hedspi/aims/screen/view/AddDVD.
fxml"));

                    AddDVDScreenController controller =
                        new AddDVDScreenController(cart, store);
                    loader.setController(controller);
                    Parent root = loader.load();
                    fxPanel.setScene(new Scene(root));

                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

d) CartScreen

```
package hust.soict.hedspi.aims.screen.view;
```

```
import java.awt.Dimension;
import java.io.IOException;

import javax.swing.JFrame;
import javax.swing.WindowConstants;

import hust.soict.hedspi.aims.Cart;
import hust.soict.hedspi.aims.Store;
import hust.soict.hedspi.aims.screen.StoreScreen;
import hust.soict.hedspi.aims.screen.controller.CartScreenController;
import javafx.application.Platform;
import javafx.embed.swing.JFXPanel;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;

public class CartScreen extends JFrame{

    private Cart cart;
    private Store store;

    public CartScreen (Cart cart, Store store) {
        super();
        this.cart = cart;
        this.store = store;

        JFXPanel fxPanel = new JFXPanel();
        this.add(fxPanel);

        this.setTitle("Cart");
        this.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);

        // Mở rộng frame khi khởi tạo
        this.setSize(new Dimension(1024, 768));
        this.setVisible(true);
        Platform.runLater(new Runnable() {
            @Override
            public void run() {
                try {
                    FXMLLoader loader = new FXMLLoader(getClass()
                        .getResource("/hust/soict/hedspi/aims/screen/view/Cart.fxml"));
                    CartScreenController controller =
```

```

        new CartScreenController(cart,store);
        loader.setController(controller);
        Parent root = loader.load();
        fxPanel.setScene(new Scene(root));

    } catch (IOException e) {
        e.printStackTrace();
    }

}

});
}
}
}

```

e) AddBook.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.Menu?>
<?import javafx.scene.control.MenuBar?>
<?import javafx.scene.control.MenuItem?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.HBox?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>

<BorderPane prefHeight="768.0" prefWidth="1024.0"
xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1">
    <top>
        <VBox prefHeight="100.0" prefWidth="100.0"
BorderPane.alignment="CENTER">
            <children>
                <MenuBar>
                    <menus>
                        <Menu mnemonicParsing="false" text="Options">
                            <items>
                                <Menu mnemonicParsing="false" text="Update Store">
                                    <items>
                                        <MenuItem mnemonicParsing="false" text="Add Book"
/>
                                <MenuItem mnemonicParsing="false" text="Add CD"
/>
                                <MenuItem mnemonicParsing="false" text="Add
DVD" />
                            </items>

```

```

        </Menu>
        <MenuItem mnemonicParsing="false"
onAction="#viewStore" text="View Store" />
        <MenuItem mnemonicParsing="false" text="View Cart" />
    </items>
</Menu>
</menus>
</MenuBar>
<Label alignment="CENTER" contentDisplay="CENTER" text="Adding
book" textAlignment="CENTER" textFill="#ff009b">
    <font>
        <Font size="24.0" />
    </font>
    <padding>
        <Insets left="400.0" />
    </padding>
</Label>
</children>
</VBox>
</top>
<center>
    <VBox prefHeight="100.0" BorderPane.alignment="CENTER">
        <children>
            <HBox prefHeight="100.0" prefWidth="200.0">
                <children>
                    <Label prefWidth="120.0" text="Title" textFill="#a11a5b">
                        <HBox.margin>
                            <Insets left="150.0" />
                        </HBox.margin>
                        <font>
                            <Font size="18.0" />
                        </font>
                    </Label>
                    <TextField fx:id="Title" prefWidth="250.0">
                        <HBox.margin>
                            <Insets left="10.0" />
                        </HBox.margin>
                    </TextField>
                </children>
            </HBox>
            <HBox layoutX="10.0" layoutY="10.0" prefHeight="100.0"
prefWidth="200.0">
                <children>
                    <Label prefWidth="120.0" text="Category"
textFill="#a11a5b">
                        <HBox.margin>
                            <Insets left="150.0" />
                        </HBox.margin>
                        <font>
                            <Font size="18.0" />
                        </font>
                    </Label>
                    <TextField fx:id="Category" prefWidth="250.0">
                        <HBox.margin>
                            <Insets left="10.0" />
                        </HBox.margin>
                    </TextField>
                </children>
            </HBox>
        </children>
    </VBox>
</center>

```

```

        </children>
    </HBox>
    <HBox layoutX="10.0" layoutY="110.0" prefHeight="100.0"
prefWidth="200.0">
        <children>
            <Label prefWidth="120.0" text="Cost" textFill="#a11a5b">
                <HBox.margin>
                    <Insets left="150.0" />
                </HBox.margin>
                <font>
                    <Font size="18.0" />
                </font>
            </Label>
            <TextField fx:id="Cost" prefWidth="250.0">
                <HBox.margin>
                    <Insets left="10.0" />
                </HBox.margin>
            </TextField>
        </children>
    </HBox>
    <HBox layoutX="10.0" layoutY="210.0" prefHeight="100.0"
prefWidth="200.0">
        <children>
            <Label prefWidth="80.0" text="Author" textFill="#a11a5b">
                <HBox.margin>
                    <Insets left="150.0" />
                </HBox.margin>
                <font>
                    <Font size="18.0" />
                </font>
            </Label>
            <Button mnemonicParsing="false" onAction="#addAuthor"
text="+" />
            <VBox fx:id="authorContainer">
                <children>
                    <TextField fx:id="author1" prefWidth="250.0" />
                </children>
                <HBox.margin>
                    <Insets left="25.0" />
                </HBox.margin>
            </VBox>
        </children>
    </HBox>
    <Button mnemonicParsing="false" onAction="#addBook"
prefHeight="50.0" prefWidth="130.0" style="-fx-background-color: Navy;"
text="ADD" textFill="#e7e7e7">
        <font>
            <Font size="18.0" />
        </font>
        <VBox.margin>
            <Insets left="400.0" />
        </VBox.margin>
    </Button>
</children>
</VBox>
</center>
</BorderPane>

```

f) addCD.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.Menu?>
<?import javafx.scene.control.MenuBar?>
<?import javafx.scene.control.MenuItem?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.HBox?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>

<BorderPane prefHeight="768.0" prefWidth="1024.0"
xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1">
    <top>
        <VBox prefHeight="100.0" prefWidth="100.0"
BorderPane.alignment="CENTER">
            <children>
                <MenuBar>
                    <menus>
                        <Menu mnemonicParsing="false" text="Options">
                            <items>
                                <Menu mnemonicParsing="false" text="Update Store">
                                    <items>
                                        <MenuItem mnemonicParsing="false" text="Add Book"
/>
                                        <MenuItem mnemonicParsing="false" text="Add CD"
/>
                                        <MenuItem mnemonicParsing="false" text="Add
DVD" />
                                    </items>
                                </Menu>
                                <MenuItem mnemonicParsing="false"
onAction="#viewStore" text="View Store" />
                                <MenuItem mnemonicParsing="false" text="View Cart" />
                            </items>
                        </Menu>
                    </menus>
                </MenuBar>
                <Label alignment="CENTER" contentDisplay="CENTER" text="Adding
CD" textAlignment="CENTER" textFill="#ff009b">
                    <font>
                        <Font size="24.0" />
                    </font>
                    <padding>
                        <Insets left="400.0" />
                    </padding>
                </Label>
            </children>
        </VBox>

```

```

</top>
<center>
  <VBox prefHeight="100.0" BorderPane.alignment="CENTER">
    <children>
      <HBox prefHeight="60.0" prefWidth="200.0">
        <children>
          <Label prefWidth="120.0" text="Title" textFill="#a11a5b">
            <HBox.margin>
              <Insets left="150.0" />
            </HBox.margin>
            <font>
              <Font size="18.0" />
            </font>
          </Label>
          <TextField fx:id="Title" prefWidth="250.0">
            <HBox.margin>
              <Insets left="10.0" />
            </HBox.margin>
          </TextField>
        </children>
      </HBox>
      <HBox layoutX="10.0" layoutY="10.0" prefHeight="60.0"
prefWidth="200.0">
        <children>
          <Label prefWidth="120.0" text="Category"
textFill="#a11a5b">
            <HBox.margin>
              <Insets left="150.0" />
            </HBox.margin>
            <font>
              <Font size="18.0" />
            </font>
          </Label>
          <TextField fx:id="Category" prefWidth="250.0">
            <HBox.margin>
              <Insets left="10.0" />
            </HBox.margin>
          </TextField>
        </children>
      </HBox>
      <HBox layoutX="10.0" layoutY="110.0" prefHeight="60.0"
prefWidth="200.0">
        <children>
          <Label prefWidth="120.0" text="Cost" textFill="#a11a5b">
            <HBox.margin>
              <Insets left="150.0" />
            </HBox.margin>
            <font>
              <Font size="18.0" />
            </font>
          </Label>
          <TextField fx:id="Cost" prefWidth="250.0">
            <HBox.margin>
              <Insets left="10.0" />
            </HBox.margin>
          </TextField>
        </children>
      </HBox>
    </children>
  </VBox>

```



```

        </HBox>
        <HBox layoutX="10.0" layoutY="260.0" prefHeight="60.0"
prefWidth="200.0">
            <children>
                <Label prefWidth="120.0" text="Director"
textFill="#a11a5b">
                    <HBox.margin>
                        <Insets left="150.0" />
                    </HBox.margin>
                    <font>
                        <Font size="18.0" />
                    </font>
                </Label>
                <TextField fx:id="Director" prefWidth="250.0">
                    <HBox.margin>
                        <Insets left="10.0" />
                    </HBox.margin>
                </TextField>
            </children>
        </HBox>
        <HBox layoutX="10.0" layoutY="310.0" prefHeight="60.0"
prefWidth="200.0">
            <children>
                <Label prefWidth="120.0" text="Length" textFill="#a11a5b">
                    <HBox.margin>
                        <Insets left="150.0" />
                    </HBox.margin>
                    <font>
                        <Font size="18.0" />
                    </font>
                </Label>
                <TextField fx:id="Length" prefWidth="250.0">
                    <HBox.margin>
                        <Insets left="10.0" />
                    </HBox.margin>
                </TextField>
            </children>
        </HBox>
        <HBox layoutX="10.0" layoutY="410.0" prefHeight="60.0"
prefWidth="200.0">
            <children>
                <Label prefWidth="120.0" text="Artist" textFill="#a11a5b">
                    <HBox.margin>
                        <Insets left="150.0" />
                    </HBox.margin>
                    <font>
                        <Font size="18.0" />
                    </font>
                </Label>
                <TextField fx:id="Artist" prefWidth="250.0">
                    <HBox.margin>
                        <Insets left="10.0" />
                    </HBox.margin>
                </TextField>
            </children>
        </HBox>

```

```

        <HBox layoutX="10.0" layoutY="510.0" prefHeight="100.0"
prefWidth="200.0">
            <children>
                <Label prefWidth="70.0" text="Track" textFill="#a11a5b">
                    <HBox.margin>
                        <Insets left="150.0" />
                    </HBox.margin>
                    <font>
                        <Font size="18.0" />
                    </font>
                </Label>
                <Button mnemonicParsing="false" onAction="#addTrack"
text="+" />
                <VBox fx:id="trackContainer" prefHeight="200.0">
                    <children>
                        <Label fx:id="header1" text="Title"
textFill="#9e0f3f">
                            <font>
                                <Font size="14.0" />
                            </font>
                            <padding>
                                <Insets left="35.0" />
                            </padding>
                        </Label>
                        <TextField fx:id="Track1" prefWidth="250.0">
                            <VBox.margin>
                                <Insets left="35.0" />
                            </VBox.margin>
                        </TextField>
                    </children>
                </VBox>
                <VBox fx:id="lengthContainer" layoutX="256.0"
layoutY="10.0" prefHeight="200.0">
                    <children>
                        <Label fx:id="header2" text="Length"
textFill="#9e0f3f">
                            <font>
                                <Font size="14.0" />
                            </font>
                            <padding>
                                <Insets left="35.0" />
                            </padding>
                        </Label>
                        <TextField fx:id="Length1" prefWidth="40.0">
                            <VBox.margin>
                                <Insets left="35.0" />
                            </VBox.margin>
                        </TextField>
                    </children>
                </VBox>
            </children>
        </HBox>
        <Button fx:id="addCD" mnemonicParsing="false" onAction="#addCD"
prefHeight="50.0" prefWidth="130.0" style="-fx-background-color: Navy;"
text="ADD" textFill="#e7e7e7">
            <VBox.margin>
                <Insets left="400.0" />
            </VBox.margin>
        </Button>
    </children>
</HBox>

```

```

        </VBox.margin>
        <font>
            <Font size="18.0" />
        </font>
    </Button>
</children>
</VBox>
</center>
</BorderPane>

```

g) AddDVD.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.Menu?>
<?import javafx.scene.control.MenuBar?>
<?import javafx.scene.control.MenuItem?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.HBox?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>

<BorderPane prefHeight="768.0" prefWidth="1024.0"
xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1">
    <top>
        <VBox prefHeight="100.0" prefWidth="100.0"
BorderPane.alignment="CENTER">
            <children>
                <MenuBar>
                    <menus>
                        <Menu mnemonicParsing="false" text="Options">
                            <items>
                                <Menu mnemonicParsing="false" text="Update Store">
                                    <items>
                                        <MenuItem mnemonicParsing="false" text="Add Book"
/>
                                        <MenuItem mnemonicParsing="false" text="Add CD"
/>
                                        <MenuItem mnemonicParsing="false" text="Add
DVD" />
                                    </items>
                                </Menu>
                                <MenuItem mnemonicParsing="false"
onAction="#viewStore" text="View Store" />
                                <MenuItem mnemonicParsing="false" text="View Cart" />
                            </items>
                        </Menu>
                    </menus>
                </MenuBar>
            </children>
        </VBox>
    </top>

```

```

        <Label alignment="CENTER" contentDisplay="CENTER" text="Adding
DVD" textAlignment="CENTER" textFill="#ff009b">
            <font>
                <Font size="24.0" />
            </font>
            <padding>
                <Insets left="400.0" />
            </padding>
        </Label>
    </children>
</VBox>
</top>
<center>
    <VBox prefHeight="100.0" BorderPane.alignment="CENTER">
        <children>
            <HBox prefHeight="100.0" prefWidth="200.0">
                <children>
                    <Label prefWidth="120.0" text="Title" textFill="#a11a5b">
                        <HBox.margin>
                            <Insets left="150.0" />
                        </HBox.margin>
                        <font>
                            <Font size="18.0" />
                        </font>
                    </Label>
                    <TextField fx:id="Title" prefWidth="250.0">
                        <HBox.margin>
                            <Insets left="10.0" />
                        </HBox.margin>
                    </TextField>
                </children>
            </HBox>
            <HBox layoutX="10.0" layoutY="10.0" prefHeight="100.0"
prefWidth="200.0">
                <children>
                    <Label prefWidth="120.0" text="Category"
textFill="#a11a5b">
                        <HBox.margin>
                            <Insets left="150.0" />
                        </HBox.margin>
                        <font>
                            <Font size="18.0" />
                        </font>
                    </Label>
                    <TextField fx:id="Category" prefWidth="250.0">
                        <HBox.margin>
                            <Insets left="10.0" />
                        </HBox.margin>
                    </TextField>
                </children>
            </HBox>
            <HBox layoutX="10.0" layoutY="110.0" prefHeight="100.0"
prefWidth="200.0">
                <children>
                    <Label prefWidth="120.0" text="Cost" textFill="#a11a5b">
                        <HBox.margin>
                            <Insets left="150.0" />

```

```

        </HBox.margin>
        <font>
            <Font size="18.0" />
        </font>
    </Label>
    <TextField fx:id="Cost" prefWidth="250.0">
        <HBox.margin>
            <Insets left="10.0" />
        </HBox.margin>
    </TextField>
</children>
</HBox>
<HBox layoutX="10.0" layoutY="260.0" prefHeight="100.0"
prefWidth="200.0">
    <children>
        <Label prefWidth="120.0" text="Director"
textFill="#a11a5b">
            <HBox.margin>
                <Insets left="150.0" />
            </HBox.margin>
            <font>
                <Font size="18.0" />
            </font>
        </Label>
        <TextField fx:id="Director" prefWidth="250.0">
            <HBox.margin>
                <Insets left="10.0" />
            </HBox.margin>
        </TextField>
    </children>
</HBox>
<HBox layoutX="10.0" layoutY="310.0" prefHeight="100.0"
prefWidth="200.0">
    <children>
        <Label prefWidth="120.0" text="Length" textFill="#a11a5b">
            <HBox.margin>
                <Insets left="150.0" />
            </HBox.margin>
            <font>
                <Font size="18.0" />
            </font>
        </Label>
        <TextField fx:id="Length" prefWidth="250.0">
            <HBox.margin>
                <Insets left="10.0" />
            </HBox.margin>
        </TextField>
    </children>
</HBox>
    <Button fx:id="addDVD" mnemonicParsing="false" onAction="#addDVD"
prefHeight="50.0" prefWidth="130.0" style="-fx-background-color: Navy;"
text="ADD" textFill="#e7e7e7">
        <VBox.margin>
            <Insets left="400.0" />
        </VBox.margin>
        <font>
            <Font size="18.0" />
        </font>
    </Button>

```

```

        </font>
    </Button>
</children>
</VBox>
</center>
</BorderPane>

```

h) Cart.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ButtonBar?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.Menu?>
<?import javafx.scene.control.MenuBar?>
<?import javafx.scene.control.MenuItem?>
<?import javafx.scene.control.RadioButton?>
<?import javafx.scene.control.TableColumn?>
<?import javafx.scene.control.TableView?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.control.ToggleGroup?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.HBox?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>

<BorderPane prefHeight="768.0" prefWidth="1024.0"
xmlns="http://javafx.com/javafx/21" xmlns:fx="http://javafx.com/fxml/1">
    <top>
        <VBox prefWidth="100.0" BorderPane.alignment="CENTER">
            <children>
                <MenuBar>
                    <menus>
                        <Menu mnemonicParsing="false" text="Options">
                            <items>
                                <Menu mnemonicParsing="false" text="Update Store">
                                    <items>
                                        <MenuItem mnemonicParsing="false" text="Add Book"
/>
                                        <MenuItem mnemonicParsing="false" text="Add CD"
/>
                                        <MenuItem mnemonicParsing="false" text="Add
DVD" />
                                    </items>
                                </Menu>
                                <MenuItem mnemonicParsing="false" onAction="#viewStore"
text="View Store" />
                                <MenuItem mnemonicParsing="false" text="View Cart" />
                            </items>
                        </Menu>
                    </menus>
                </MenuBar>
            </children>
        </VBox>
    </top>

```

```

        <Label text="CART" textFill="AQUA">
            <font>
                <Font size="50.0" />
            </font>
            <padding>
                <Insets left="10.0" />
            </padding>
        </Label>
    </children>
</VBox>
</top>
<center>
    <VBox prefHeight="200.0" prefWidth="100.0"
BorderPane.alignment="CENTER">
        <padding>
            <Insets left="10.0" />
        </padding>
        <children>
            <HBox alignment="CENTER_LEFT" prefWidth="200.0" spacing="10.0">
                <padding>
                    <Insets bottom="10.0" top="10.0" />
                </padding>
                <children>
                    <Label text="Filter:" />
                    <TextField fx:id="tfFilter" />
                    <RadioButton fx:id="radioBtnFilterId"
mnemonicParsing="false" selected="true" text="By ID">
                        <toggleGroup>
                            <ToggleGroup fx:id="filterCategory" />
                        </toggleGroup>
                    </RadioButton>
                    <RadioButton fx:id="radioBtnFilterTitle"
mnemonicParsing="false" text="By Title" toggleGroup="$filterCategory" />
                </children>
            </HBox>
            <TableView fx:id="tblMedia">
                <columns>
                    <TableColumn fx:id="colMediaTitle" prefWidth="75.0"
text="Title" />
                    <TableColumn fx:id="colMediaCategory" prefWidth="75.0"
text="Category" />
                    <TableColumn fx:id="colMediaCost" prefWidth="75.0"
text="Cost" />
                </columns>
                <columnResizePolicy>
                    <TableView fx:constant="CONSTRAINED_RESIZE_POLICY" />
                </columnResizePolicy>
            </TableView>
            <ButtonBar prefHeight="40.0" prefWidth="200.0">
                <buttons>
                    <Button fx:id="btnPlay" mnemonicParsing="false"
onAction="#playMedia" text="Play" />
                    <Button fx:id="btnRemove" layoutX="948.0" layoutY="17.0"
mnemonicParsing="false" onAction="#btnRemovePressed" text="Remove" />
                </buttons>
            </ButtonBar>
        </children>
    </VBox>
</center>

```

```

        </VBox>
    </center>
</right>
    <VBox prefHeight="200.0" BorderPane.alignment="TOP_CENTER">
        <padding>
            <Insets top="50.0" />
        </padding>
        <children>
            <HBox alignment="CENTER">
                <children>
                    <Label text="Total:">
                        <font>
                            <Font size="24.0" />
                        </font>
                    </Label>
                    <Label fx:id="totalCost" text="0 $" textFill="AQUA">
                        <font>
                            <Font size="24.0" />
                        </font>
                    </Label>
                </children>
            </HBox>
            <Button fx:id="placeOrder" mnemonicParsing="false"
onAction="#placeOrderPressed" style="-fx-background-color: red;" text="Place
Order" textFill="WHITE">
                <font>
                    <Font size="24.0" />
                </font>
            </Button>
        </children>
    </VBox>
</right>
</BorderPane>

```

E) hust.soict.hedspi.aims.exception

a) PlayerException.java

```

package hust.soict.hedspi.aims.exception;

public class PlayerException extends Exception {
    public PlayerException(String message) {
        super(message);
    }
}

```

F. hust.soict.hedspi.javafx

a) Painter.java

```

package hust.soict.hedspi.javafx;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;

```



```
import javafx.scene.Scene;
import javafx.stage.Stage;

public class Painter extends Application{

    @Override

    public void start(Stage stage) throws Exception {

        Parent root = FXMLLoader.load(getClass()
            .getResource("/hust/soict/hedsapi/javafx/Painter.fxml"));
        //Lấy file fxml
        Scene scene = new Scene(root);
        stage.setTitle("Painter");
        stage.setScene(scene);
        stage.show();
        //Khởi tạo

    }
    public static void main(String[] args) {
        launch(args);

        //Chạy
    }
}
```

b) PainterController.java

```
package hust.soict.hedsapi.javafx;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.RadioButton;
import javafx.scene.control.ToggleGroup;
import javafx.scene.input.MouseEvent;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;
import javafx.scene.layout.Pane;

public class PainterController {

    private Color penColor = Color.BLACK;
    //Biến lưu giữ màu

    @FXML
    private Pane drawingAreaPane;
```

```
@FXML
private RadioButton Pen;

@FXML
private RadioButton Eraser;
//Các thuộc tính khác
@FXML
void clearButtonPressed(ActionEvent event) {
    drawingAreaPane.getChildren().clear();
    //Khi xóa bảng
}

@FXML
void drawingAreaMouseDragged(MouseEvent event) {
    Circle newCircle = new Circle(event.getX(),
        event.getY() , 4,penColor);
    drawingAreaPane.getChildren().add(newCircle);
    //Khi vẽ
}

@FXML
public void initialize() {
    ToggleGroup toggleGroup = new ToggleGroup();
    Pen.setToggleGroup(toggleGroup);
    Eraser.setToggleGroup(toggleGroup);

    Pen.setSelected(true);

    // Event handlers khi chọn pen hoặc eraser
    Pen.setOnAction(this::handlePenSelection);
    Eraser.setOnAction(this::handleEraserSelection);
}

private void handlePenSelection(ActionEvent event) {
    penColor = Color.BLACK;
    //Chọn bút thì màu đen
}

private void handleEraserSelection(ActionEvent event) {
    // Chọn tẩy màu trắng
    penColor = Color.WHITE;
}
```

}

C) painter.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.RadioButton?>
<?import javafx.scene.control.TitledPane?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.Pane?>
<?import javafx.scene.layout.VBox?>

<BorderPane id="drawingAreaPane" maxHeight="-Infinity" maxWidth="-Infinity"
minHeight="-Infinity" minWidth="-Infinity" prefHeight="480.0"
prefWidth="640.0" xmlns="http://javafx.com/javafx/21"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="hust.soict.hedspi.javafx.PainterController">
    <padding>
        <Insets bottom="8.0" left="8.0" right="8.0" top="8.0" />
    </padding>
    <left>
        <VBox maxHeight="1.7976931348623157E308" spacing="8.0"
BorderPane.alignment="CENTER">
            <BorderPane.margin>
                <Insets right="8.0" />
            </BorderPane.margin>
            <children>
                <TitledPane animated="false" text="Tools">
                    <content>
                        <AnchorPane>
                            <children>
                                <RadioButton fx:id="Pen" layoutX="6.0" layoutY="5.0"
mnemonicParsing="false" text="Pen" />
                                <RadioButton fx:id="Eraser" layoutX="6.0"
layoutY="28.0" mnemonicParsing="false" text="Eraser" />
                            </children>
                        </AnchorPane>
                    </content>
                </TitledPane>
                <Button maxWidth="1.7976931348623157E308" mnemonicParsing="false"
onAction="#clearButtonPressed" text="Clear" />
            </children>
        </VBox>
    </left>
    <center>
        <Pane fx:id="drawingAreaPane" onMouseDragged="#drawingAreaMouseDragged"
prefHeight="200.0" prefWidth="200.0" style="-fx-background-color: white;"
BorderPane.alignment="CENTER" />
    </center>
</BorderPane>

```

G) hust.soict.hedspi.swing
a) AWTAccumulator

```
package hust.soict.hedspi.swing;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class AWTAccumulator extends Frame {
    private TextField tfInput;
    private TextField tfOutput;
    private int sum = 0; // tổng để là 0

    // Constructor để setup các thành phần GUI và events handler
    public AWTAccumulator() {
        setLayout(new GridLayout(2, 2));

        add(new Label("Enter an Integer: "));

        tfInput = new TextField(10);
        add(tfInput);
        tfInput.addActionListener(new TFInputListener());

        add(new Label("The Accumulated Sum is: "));

        tfOutput = new TextField(10);
        tfOutput.setEditable(false);
        add(tfOutput);

        setTitle("AWT Accumulator");
        setSize(350, 120);
        setVisible(true);
    }

    public static void main(String[] args) {
        new AWTAccumulator();
    }

    private class TFInputListener implements ActionListener {
        @Override
        public void actionPerformed(ActionEvent evt) {
            int numberIn = Integer.parseInt(tfInput.getText());
            sum += numberIn;
            tfInput.setText("");
            tfOutput.setText(sum + "");
            //ActionListener, khi nhập một số mới vào thì cộng vào tổng
        }
    }
}
```

b) NumberGrid.java

```
package hust.soict.hedspi.swing;
```

```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.*;

public class NumberGrid extends JFrame{
    private JButton [] btnNumbers = new JButton [10];
    private JButton btnDelete, btnReset;
    private JTextField tfDisplay;

    public NumberGrid() {
        //Khởi tạo gui
        tfDisplay = new JTextField();
        tfDisplay.setComponentOrientation(
            ComponentOrientation.RIGHT_TO_LEFT);

        JPanel panelButtons = new JPanel(new GridLayout(4, 3));
        addButtons(panelButtons);

        Container cp = getContentPane();
        cp.setLayout(new BorderLayout());
        cp.add(tfDisplay, BorderLayout.NORTH);
        cp.add(panelButtons, BorderLayout.CENTER);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setTitle("Number Grid");
        setSize(200, 200);
        setVisible(true);
    }

    void addButtons (JPanel panelButtons) {
        ButtonListener btnListener = new ButtonListener();
        for(int i = 1; i <= 9; i++) {
            btnNumbers[i] = new JButton(""+i);
            panelButtons.add(btnNumbers[i]);
            btnNumbers[i].addActionListener(btnListener);
        }
        //Thêm các số từ 1 đến 0
        btnDelete = new JButton("DEL");
        panelButtons.add(btnDelete);
        btnDelete.addActionListener(btnListener);
        //Thêm Del
        btnNumbers [0] = new JButton("0");
    }
}
```

```

        panelButtons.add(btnNumbers[0]);
        //Thêm số 0
        btnNumbers[0].addActionListener(btnListener);

        btnReset = new JButton("C");
        panelButtons.add(btnReset);
        btnReset.addActionListener(btnListener);
        //Thêm C để clear
    }

    private class ButtonListener implements ActionListener{
        @Override
        public void actionPerformed(ActionEvent e) {
            String button = e.getActionCommand();
            if(button.charAt(0) >= '0' && button.charAt(0) <= '9') {
                tfDisplay.setText(tfDisplay.getText() + button);
            } // Nếu chọn là 0 đến 9 thì gán vào string
            else if (button.equals("DEL")) {
                tfDisplay.setText(tfDisplay.getText().substring(0,
tfDisplay.getText().length() - 1));
                //Nếu là del thì bỏ ký tự cuối
            }
            else { //Không là C thì chuyển thành xâu rỗng
                tfDisplay.setText("");
            }
        }
    }

    public static void main(String[] args) {
        new NumberGrid();
    }
}

```

c) SwingAccumulator.java

```

package hust.soict.hedspi.swing;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.*;

public class SwingAccumulator extends JFrame {
    private JTextField tfInput;
    private JTextField tfOutput; private int sum = 0;

```

```
// tổng để là 0

// Constructor để setup các thành phần GUI và events handler
public SwingAccumulator() {
    Container cp = getContentPane();
    cp.setLayout(new GridLayout(2, 2));

    cp.add(new JLabel("Enter an Integer: "));

    tfInput = new JTextField(10);
    cp.add(tfInput); tfInput.addActionListener(new TFInputListener());
    cp.add(new JLabel("The Accumulated Sum is: "));

    tfOutput = new JTextField(10);
    tfOutput.setEditable(false);
    cp.add(tfOutput);

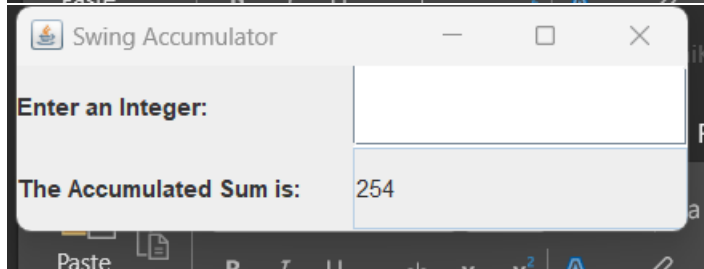
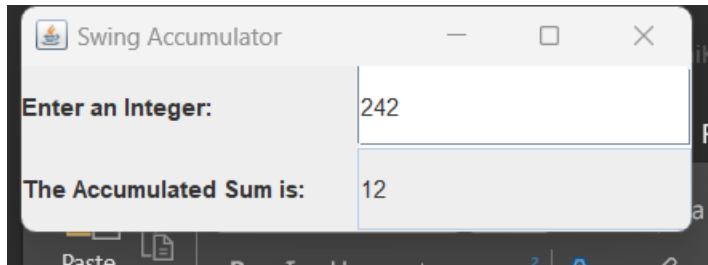
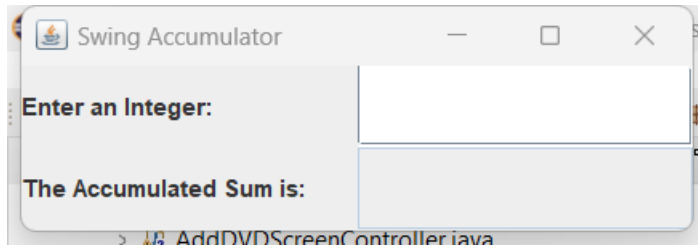
    setTitle("Swing Accumulator");
    setSize(350, 120);
    setVisible(true);
}

public static void main(String[] args) {
    new SwingAccumulator();
}

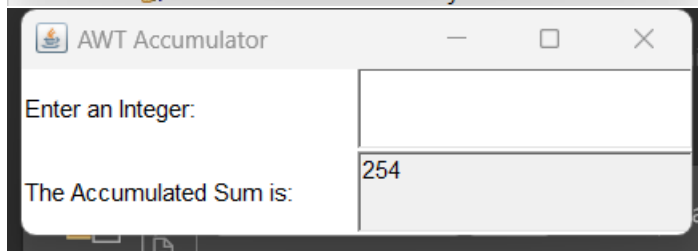
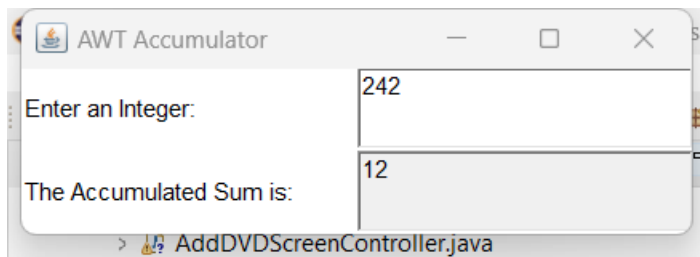
private class TFInputListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent evt) {
        int numberIn = Integer.parseInt(tfInput.getText());
        sum += numberIn;
        tfInput.setText("");
        tfOutput.setText(sum + "");
        //ActionListener, khi nhập một số mới vào thì cộng vào tổng
    }
}
}
```

7. Demo chương trình

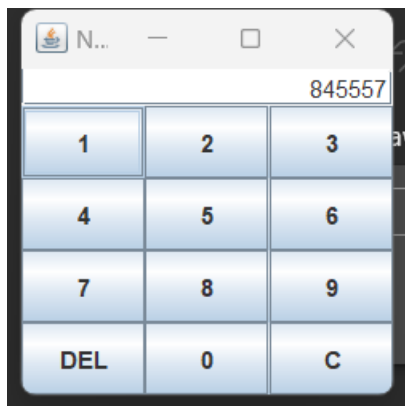
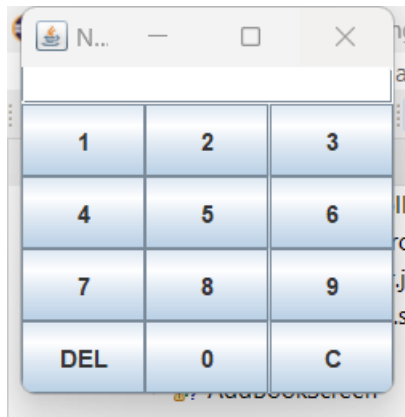
- A) Swing
 - a) Swing accumulator



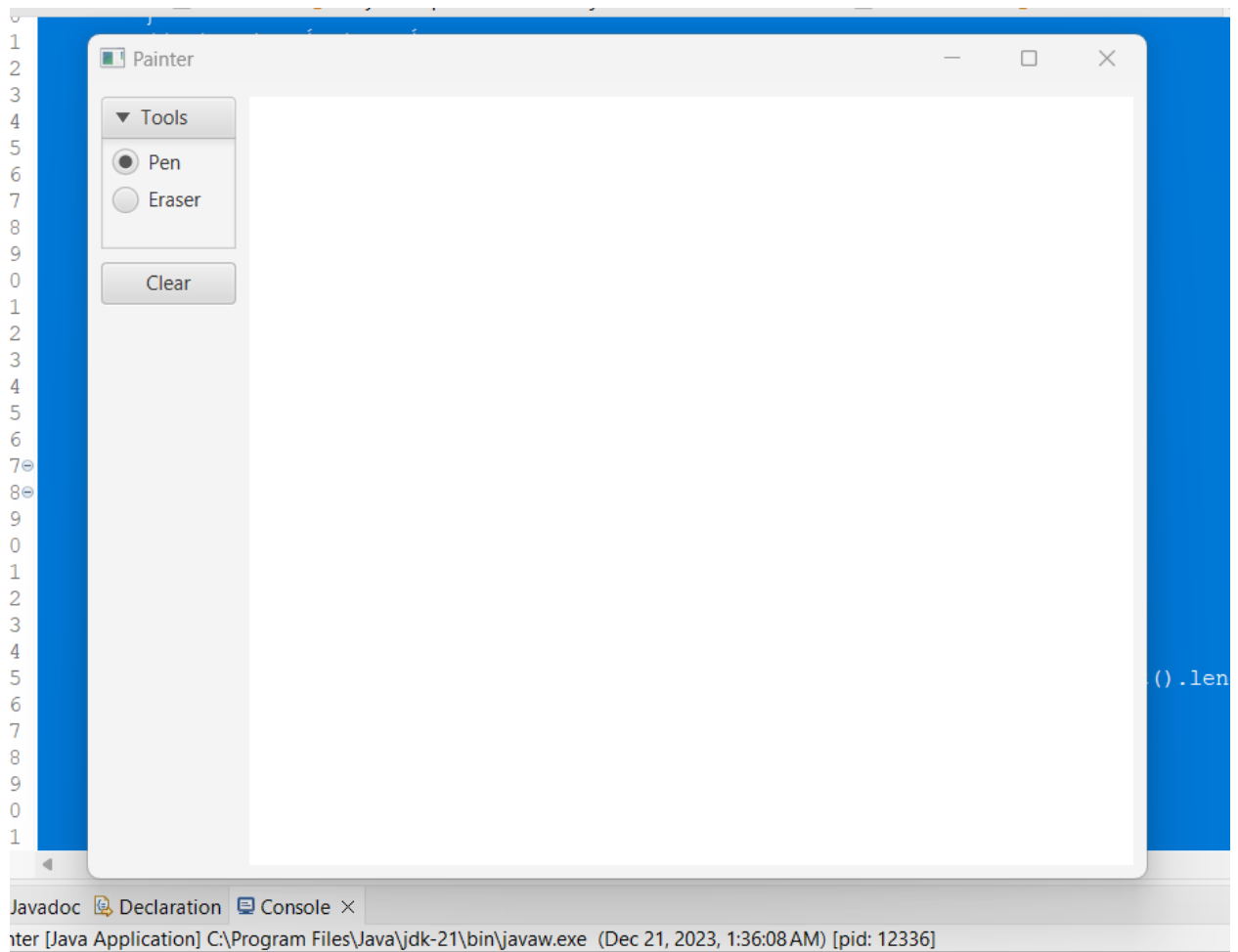
b) AWT accumulator



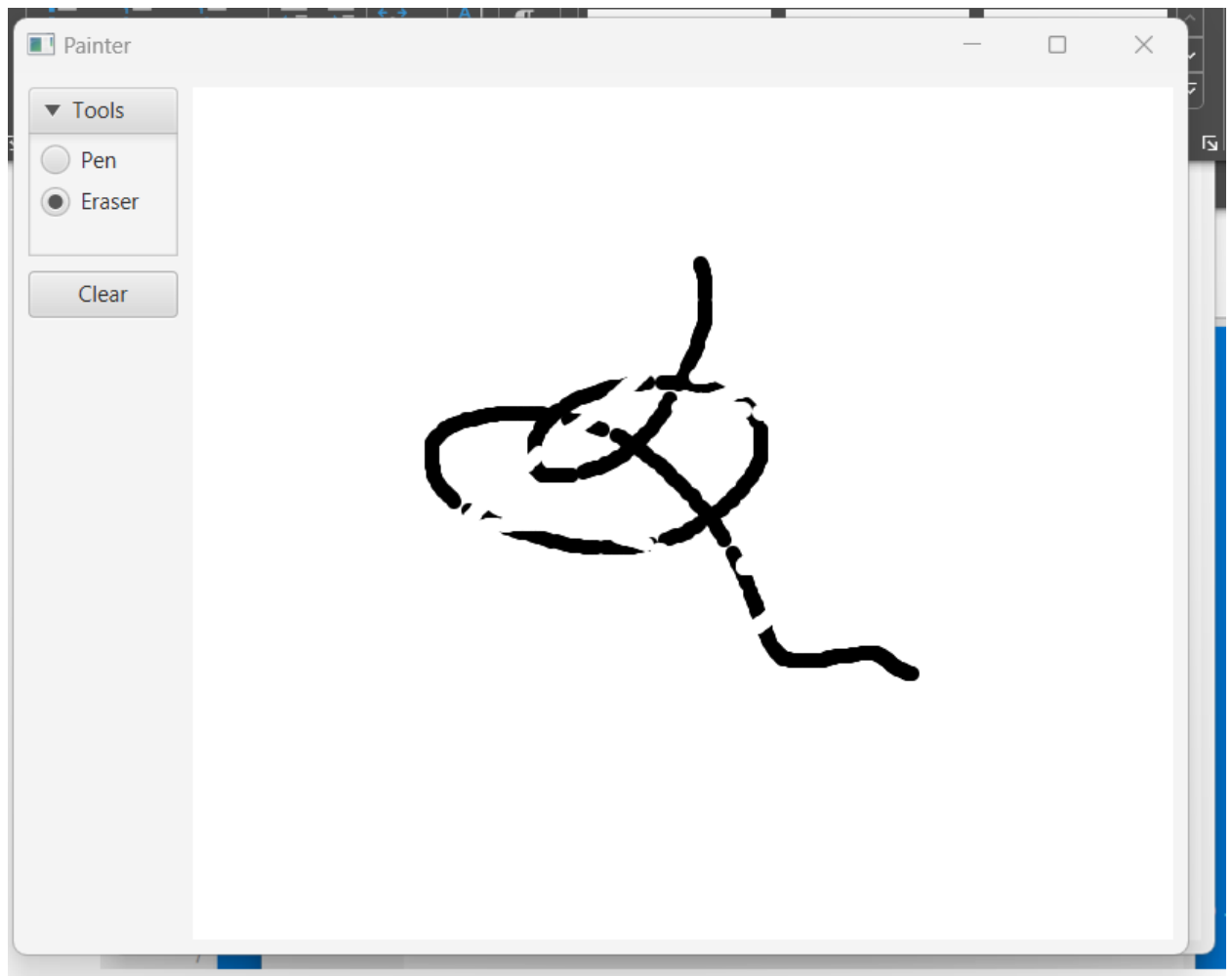
c) NumberGrid



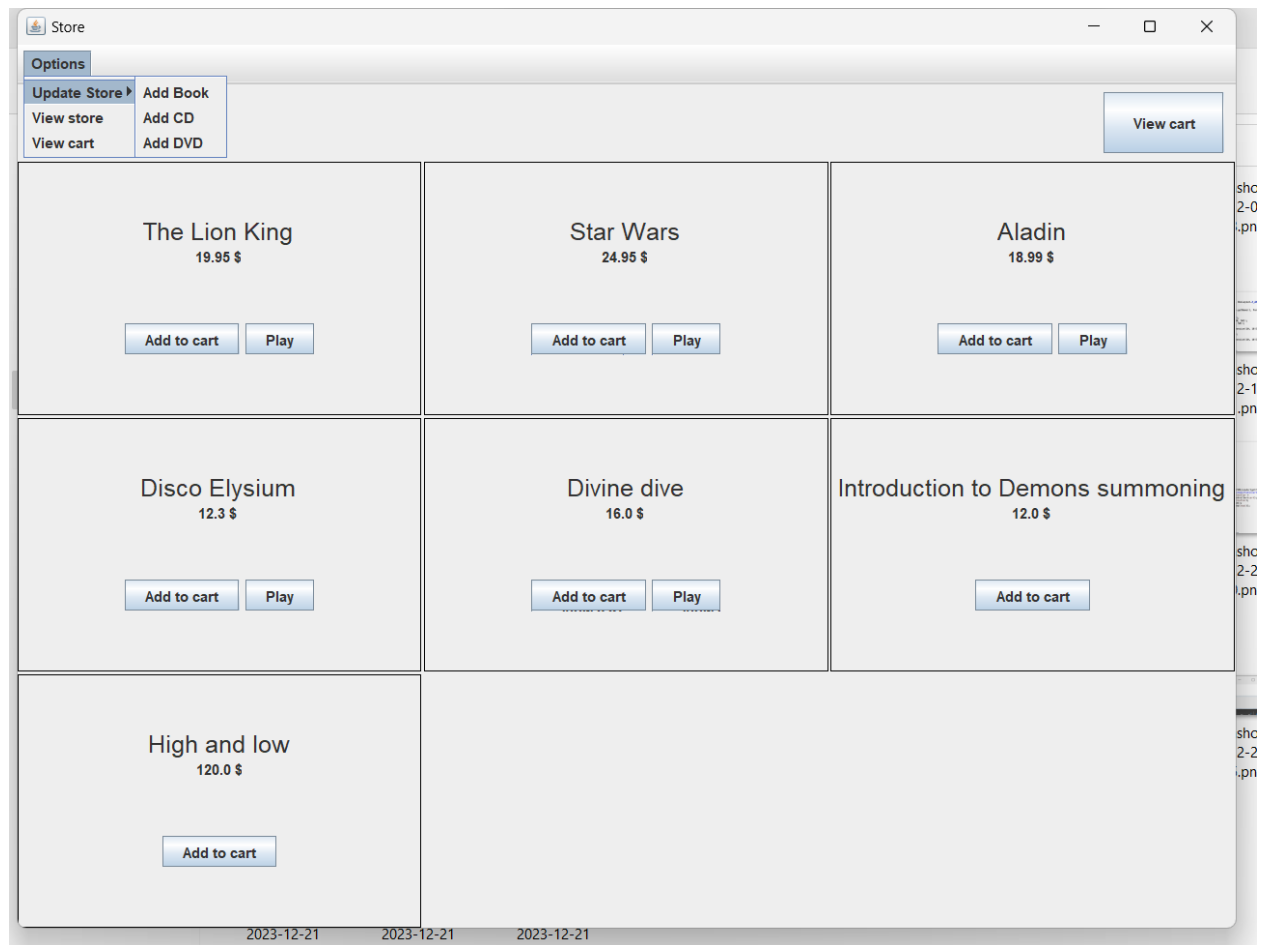
B) JavaFX

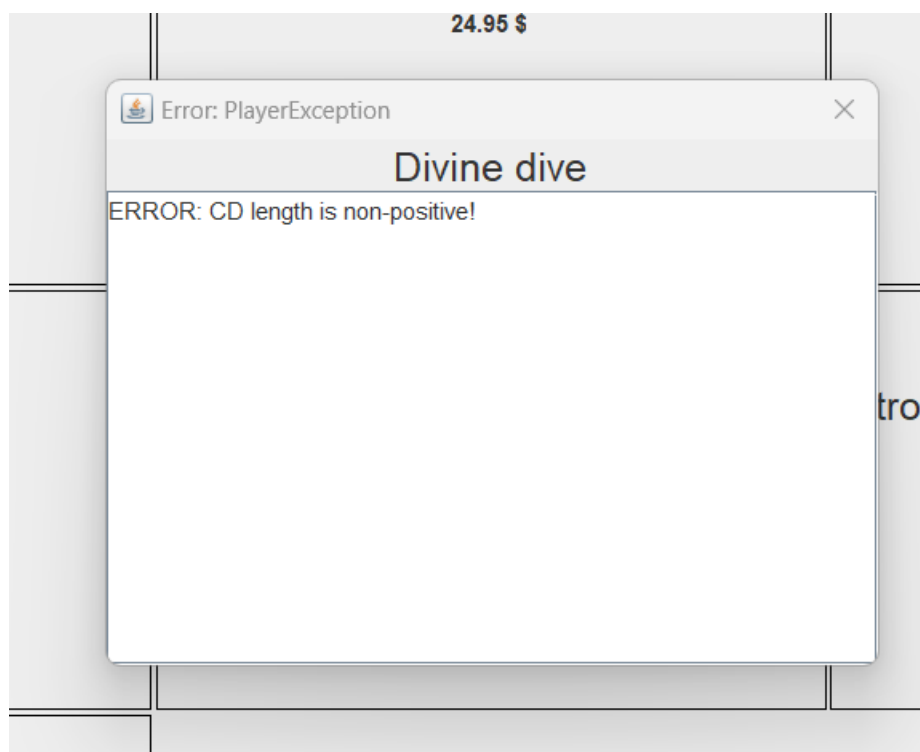
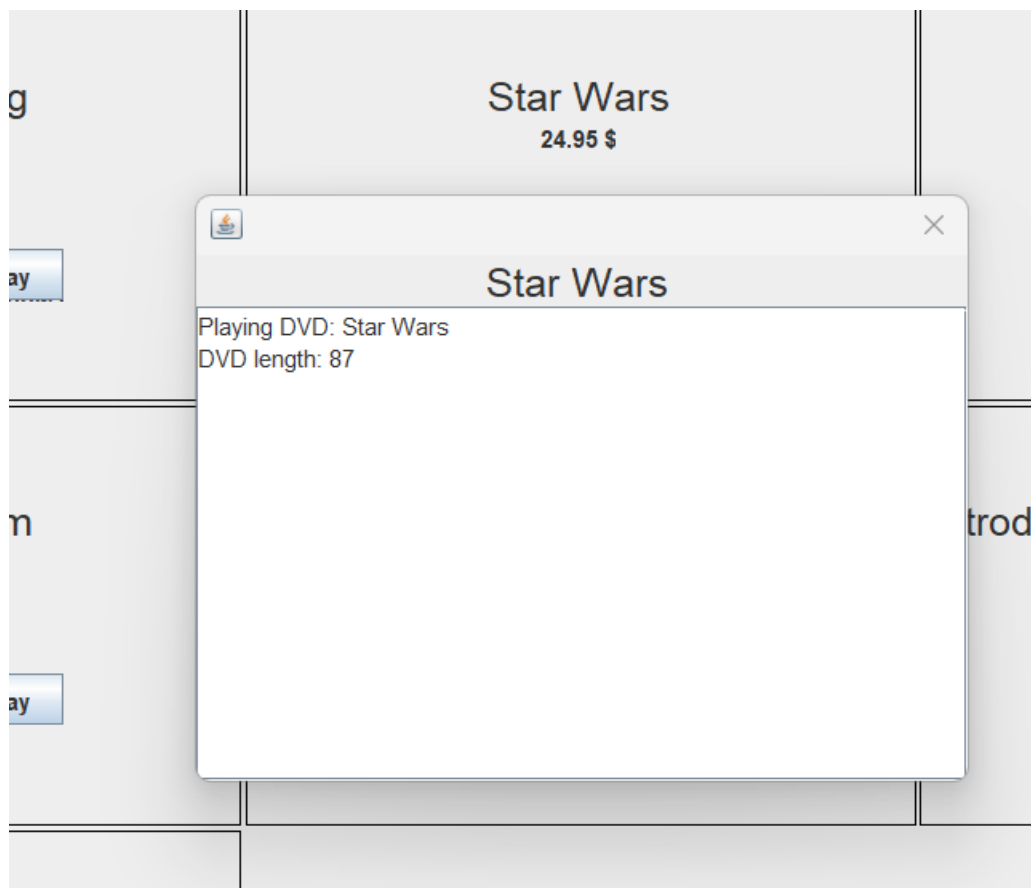






d) Aims





Cart

Options

CART

Filter: ☐ By ID ☒ By Title

Title	Category	Cost
Star Wars	Science Fiction	24.95
The Lion King	Animation	19.95
Disco Elysium	High Classic	12.3
High and low	shitpost	120.0
Divine dive	Blue	16.0
Introduction to Demons summoning	Occult	12.0
Aladin	Animation	18.99

Total:224.19\$

Place Order

Cart

Options

CART

Filter:

☐ By ID
 ☒ By Title

Title	Category	Cost
Star Wars	Science Fiction	24.95
Disco Elysium	High Classic	12.3
Introduction to Demons summoning	Occult	12.0

Total: 224.19\$

Place Order

Play Remove

Options

CART

Filter: ☒ By ID ☐ By Title

Title	Category	Cost
Aladin	Animation	18.99

Play

Remove

Total:224.19!

Place Order

Options

CART

Filter:

☐ By ID ☒ By Title

Title	Category	Cost
Star Wars	Science Fiction	24.95
The Lion King	Animation	19.95
Disco Elysium	High Classic	12.5
High and low	shitpost	
Divine dive	Blue	
Introduction to Demons summoning	Occult	
Aladin	Animation	

Total: 224.19\$

Place Order

Error: PlayerException

Disco Elysium

ERROR: CD length is non-positive!

Play Remove

Cart

Options

CART

Filter:

☐ By ID
 ☒ By Title

Title	Category	Cost
Star Wars	Science Fiction	24.95
The Lion King	Animation	19.95
Disco Elysium	High Classic	12.2
High and low	shitpost	
Divine dive	Blue	
Introduction to Demons summoning	Occult	
Aladin	Animation	

Total:224.19\$

Place Order

Star Wars

Playing DVD: Star Wars

DVD length: 87

Play

Remove

Cart

Options

CART

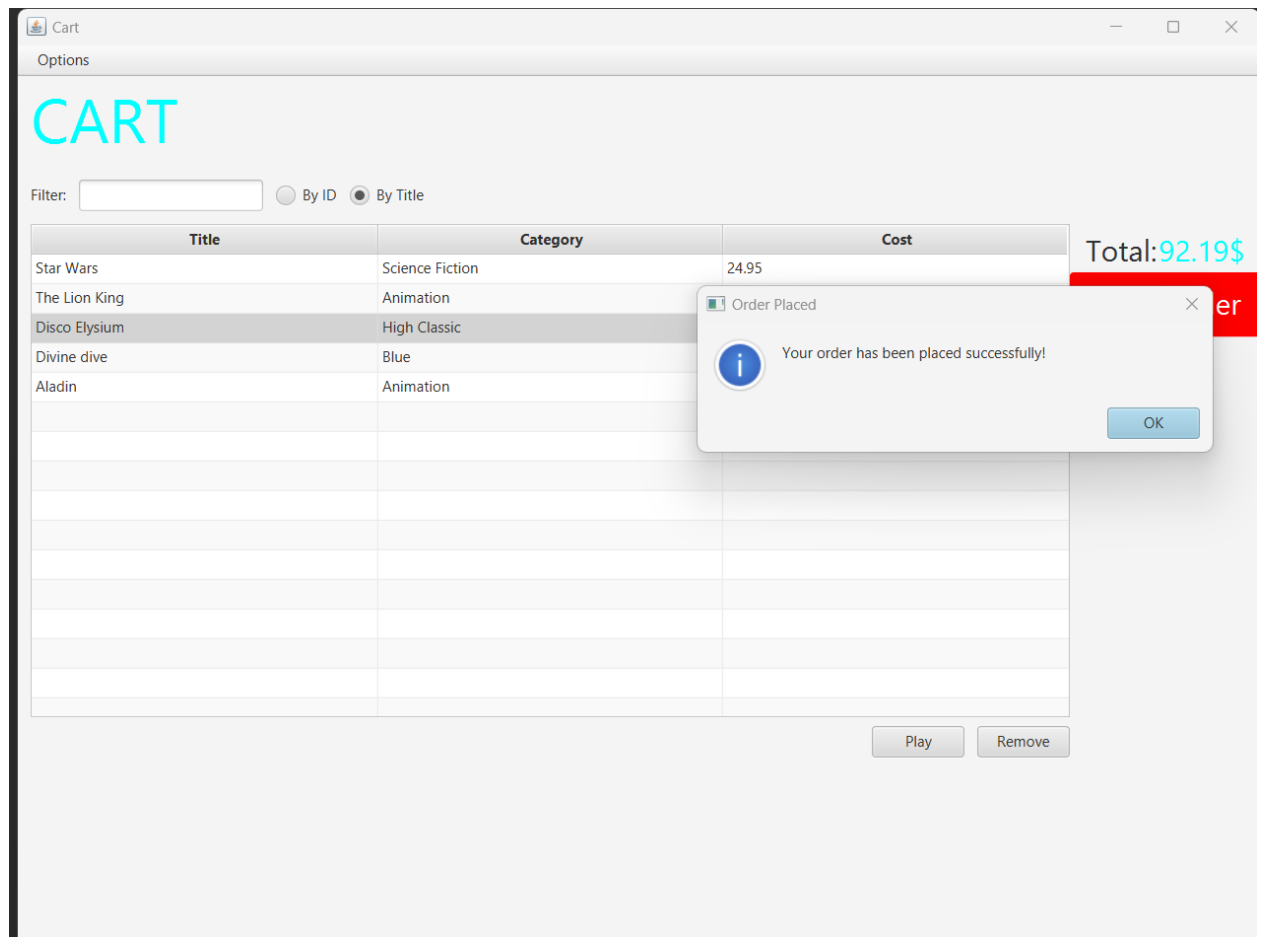
Filter:
☐ By ID
 ☒ By Title


Title	Category	Cost
Star Wars	Science Fiction	24.95
The Lion King	Animation	19.95
Disco Elysium	High Classic	12.3
High and low	shitpost	120.0
Divine dive	Blue	16.0
Introduction to Demons summoning	Occult	12.0
Aladin	Animation	18.99

Total:224.19\$

Place Order

Remove



 Cart

Options

CART

Filter: ☐ By ID ☒ By Title

Title	Category	Cost
No content in table		

Play Remove

Total:0.00\$
Place Order

AddingBook

Options

Adding book

Title

Category

Cost

Author

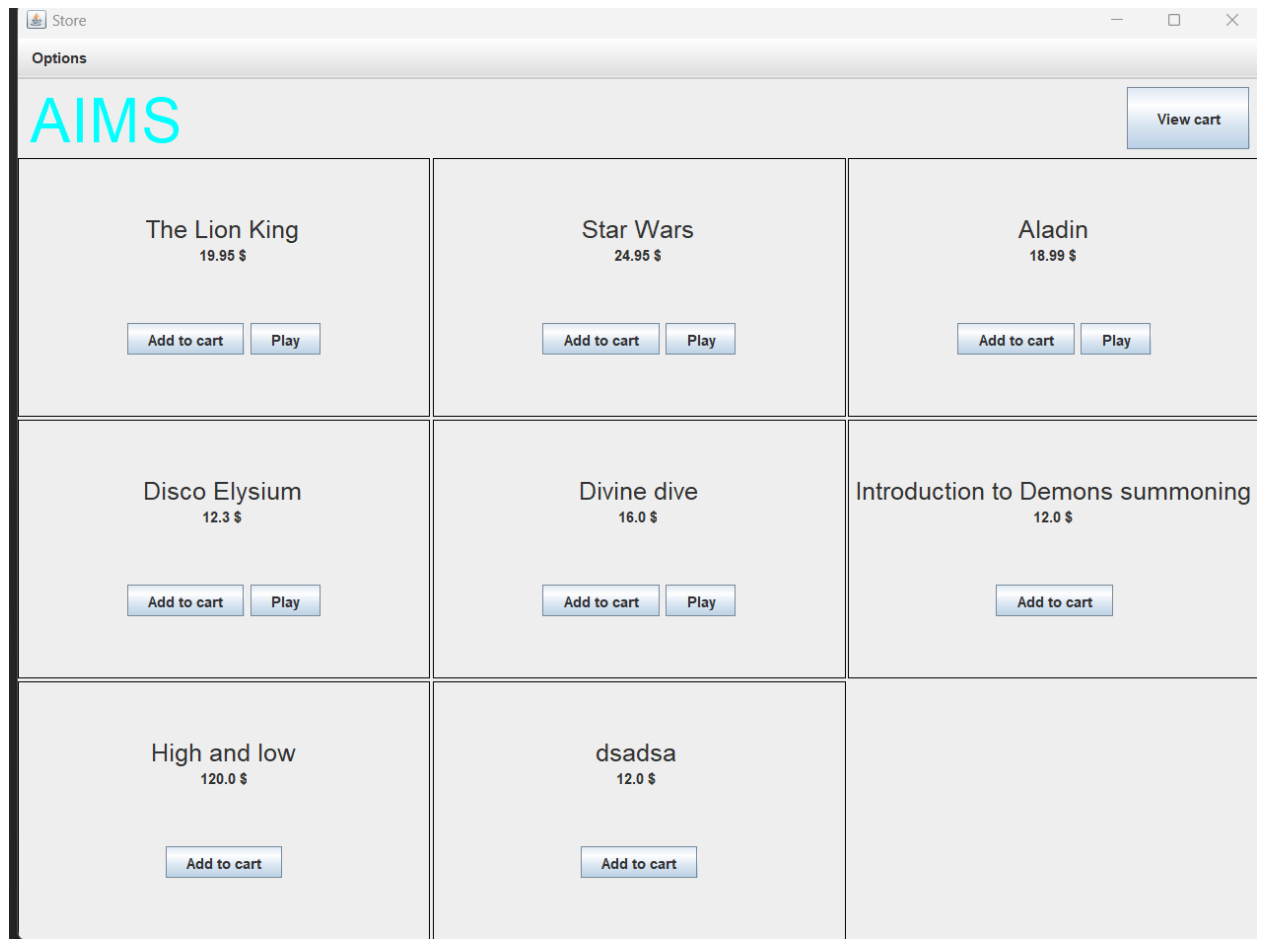
+

ADD

Success

Book added successfully.

OK



AddingDVD

Options

Adding DVD

Title

Category


Cost

Director

Length

ADD

Success

 DVD added successfully.

OK

AddingDVD

Options

Adding DVD

Title

Category


Cost

Director

Length

ADD

Warning

 Please enter a title for the dvd.

OK

AddingCD

Options

Adding CD

Title

vsvdsv

Category

123

Cost

21

Director

fsdf

Length

12

Artist

fafwf

Track

+

Title

dasd

Length

23

sad

12

ADD

Success

i

CD added successfully.

OK

