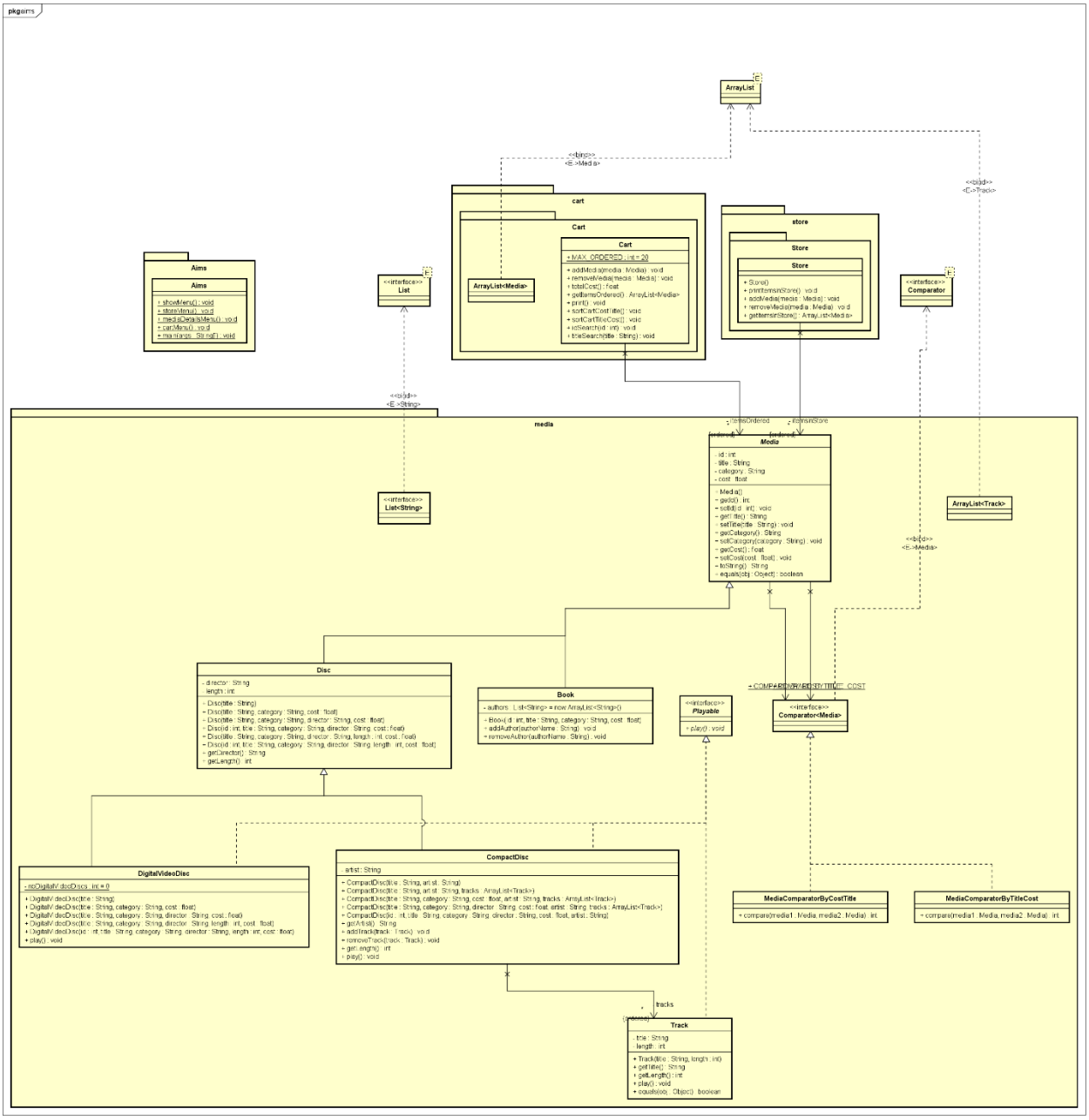
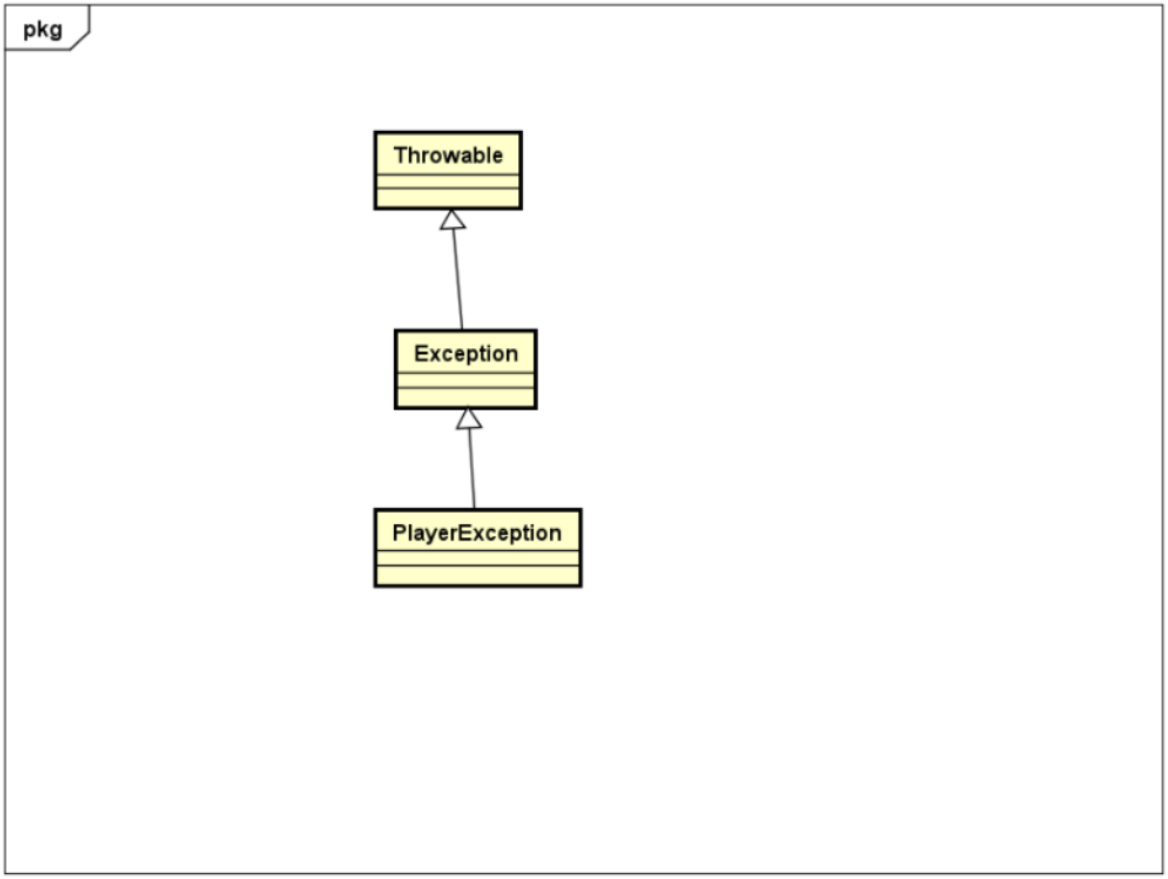
Name: Đặng Thịnh Tường Minh

Student ID: 20235528

- **Class diagram**





1.Swing components

AWTAccumulator

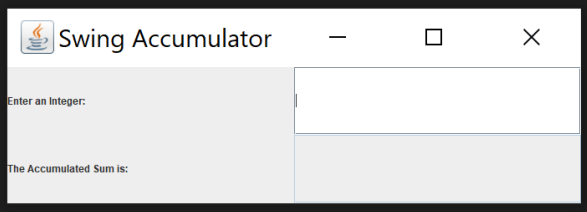
src > hust > soict > dsai > swing > AWTAccumulator.java

```java
package hust.soict.dsai.swing;

import java.awt.*;
import java.awt.event.*;

public class AWTAccumulator extends Frame {
    private TextField tfInput;
    private TextField tfOutput;
    private int sum = 0; // Accumulated sum, init to 0

    // Constructor to setup the GUI components and event handlers
    public AWTAccumulator() {
        setLayout(new GridLayout(rows:2, cols:2));
        add(new Label(text:"Enter an Integer: "));
        tfInput = new TextField(columns:10);
        add(tfInput);
        tfInput.addActionListener(new TFInputListener());
        add(new Label(text:"The Accumulated Sum is: "));
        tfOutput = new TextField(columns:10);
        tfOutput.setEditable(b:false);
        add(tfOutput);
        setTitle(title:"AWT Accumulator");
        setSize(width:350, height:120);
        setVisible(b:true);
    }

    Run | Debug
    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(t:"");
            tfOutput.setText(sum + "");
        }
    }
}
```

SwingAccumulator

hust > soict > dsai > swing > SwingAccumulator.java > SwingAccumulator > SwingAccumulator()

```java
package hust.soict.dsai.swing;

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class SwingAccumulator extends JFrame {
    private JTextField tfInput;
    private JTextField tfOutput;
    private int sum = 0; // Accumulated sum, init to 0

    // Constructor to setup the GUI components and event handlers
    public SwingAccumulator() {
        Container cp = getContentPane();
        cp.setLayout(new GridLayout(rows:2, cols:2));
        cp.add(new JLabel(text:"Enter an Integer: "));
        tfInput = new JTextField(columns:10);
        cp.add(tfInput);
        tfInput.addActionListener(new TFInputListener());
        cp.add(new JLabel(text:"The Accumulated Sum is: "));
        tfOutput = new JTextField(columns:10);
        tfOutput.setEditable(b:false);
        cp.add(tfOutput);
        setTitle(title:"Swing Accumulator");
        setSize(width:350, height:120);
        setVisible(b:true);
    }

    Run | Debug
    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(t:"");
            tfOutput.setText(sum + "");
        }
    }
}
```



2.Organizing Swing components with Layout Managers

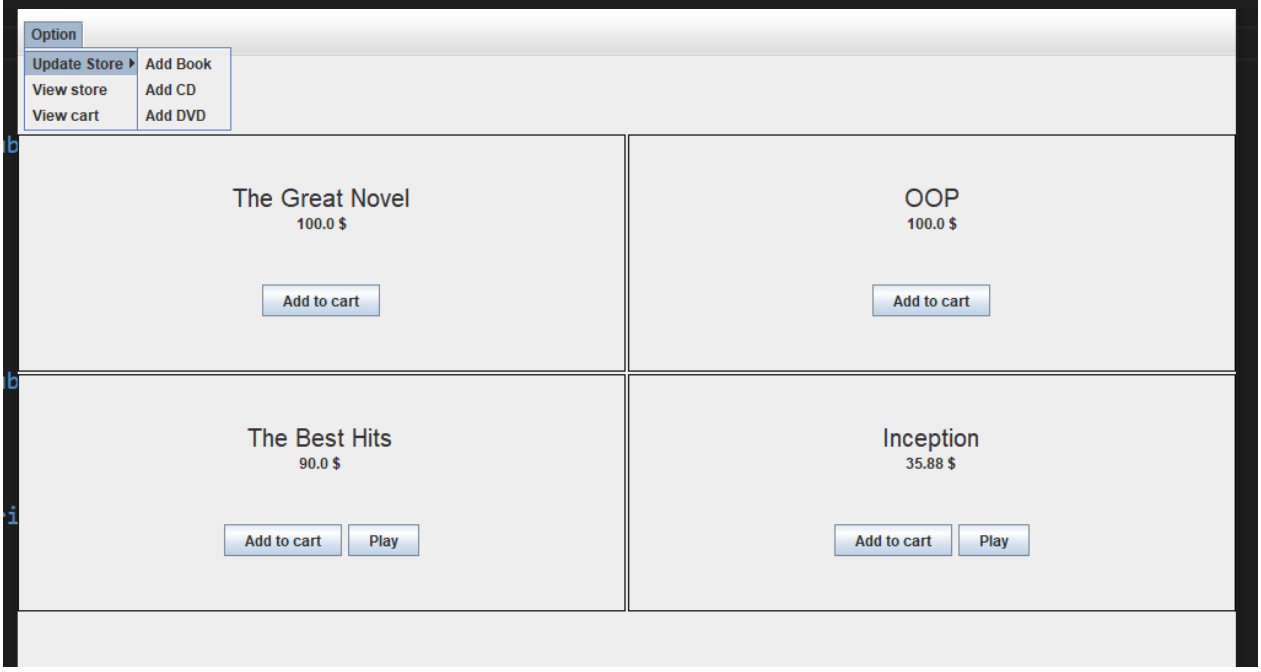
src > hust > soict > dsai > swing > NumberGrid.java > NumberGrid > NumberGrid()

```java
package hust.soict.dsai.swing;

import java.awt.BorderLayout;
import java.awt.ComponentOrientation;
import java.awt.Container;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JTextField;

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

    public NumberGrid() {

        tfDisplay = new JTextField();
        tfDisplay.setComponentOrientation(ComponentOrientation.RIGHT_TO_LEFT);

        JPanel panelButtons = new JPanel(new GridLayout(rows:4, cols: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(title:"Number Grid");
        setSize(width:200, height:200);
        setVisible(b:true);
    }

    private class ButtonListener implements ActionListener{
        @Override
        public void actionPerformed(ActionEvent e) {
            String button = e.getActionCommand();
            if (button.charAt(index:0) >= '0' && button.charAt(index:0) <= '9') {
                tfDisplay.setText(tfDisplay.getText() + button);
            }else if (button.equals(anObject:"DEL")) {
                if (tfDisplay.getText().length() >= 1) {
                    tfDisplay.setText(tfDisplay.getText().substring(beginIndex:0, tfDisplay.getText().length()-1));
                }
            }else if (button.equals(anObject:"C")) {
                tfDisplay.setText(t:"");
            }else {
                tfDisplay.setText(tfDisplay.getText());
            }
        }
    }

    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);
        }
        btnDelete = new JButton(text:"DEL");
        panelButtons.add(btnDelete);
        btnDelete.addActionListener(btnListener);

        btnNumbers[0] = new JButton(text:"0");
        panelButtons.add(btnNumbers[0]);
        btnNumbers[0].addActionListener(btnListener);

        btnReset = new JButton(text:"C");
        panelButtons.add(btnReset);
        btnReset.addActionListener(btnListener);
    }

```

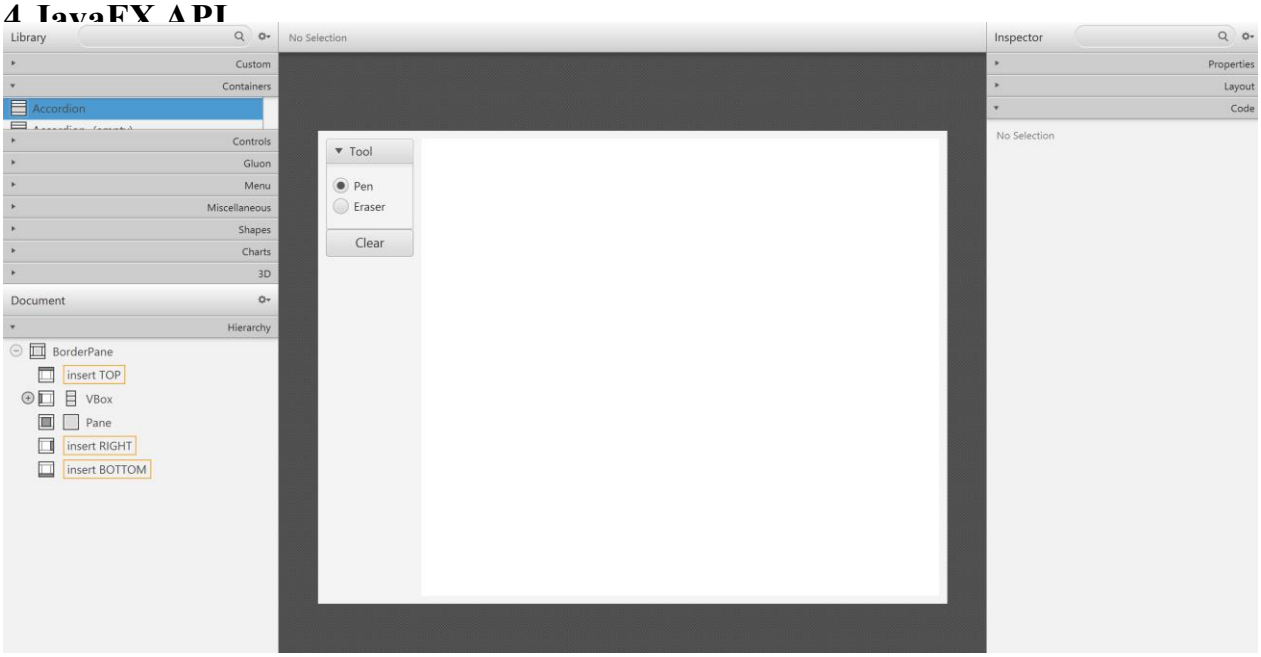
3.Create a graphical user interface for AIMS with Swing

```
public class MediaStore extends JPanel {
    private Media media;
    private Cart cart;
    private ButtonListener btnListener = new ButtonListener();
    private JButton btnAddToCart, btnPlay;

    public class ButtonListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            String button = e.getActionCommand();
            if(button.charAt(index:0) == 'A') {
                JFrame frame = new JFrame(title:"JOptionPane ");
                JOptionPane.showMessageDialog(frame,
                        message:"The media has been added",
                        title:"Add To Cart",
                        JOptionPane.INFORMATION_MESSAGE);
            }
            else if (button.charAt(index:0) == 'P') {
                JFrame frame = new JFrame(title:"JOptionPane ");
                JOptionPane.showMessageDialog(frame,
                        message:"Play The Media",
                        title:"Play",
                        JOptionPane.INFORMATION_MESSAGE);
            }
        }
    }
```

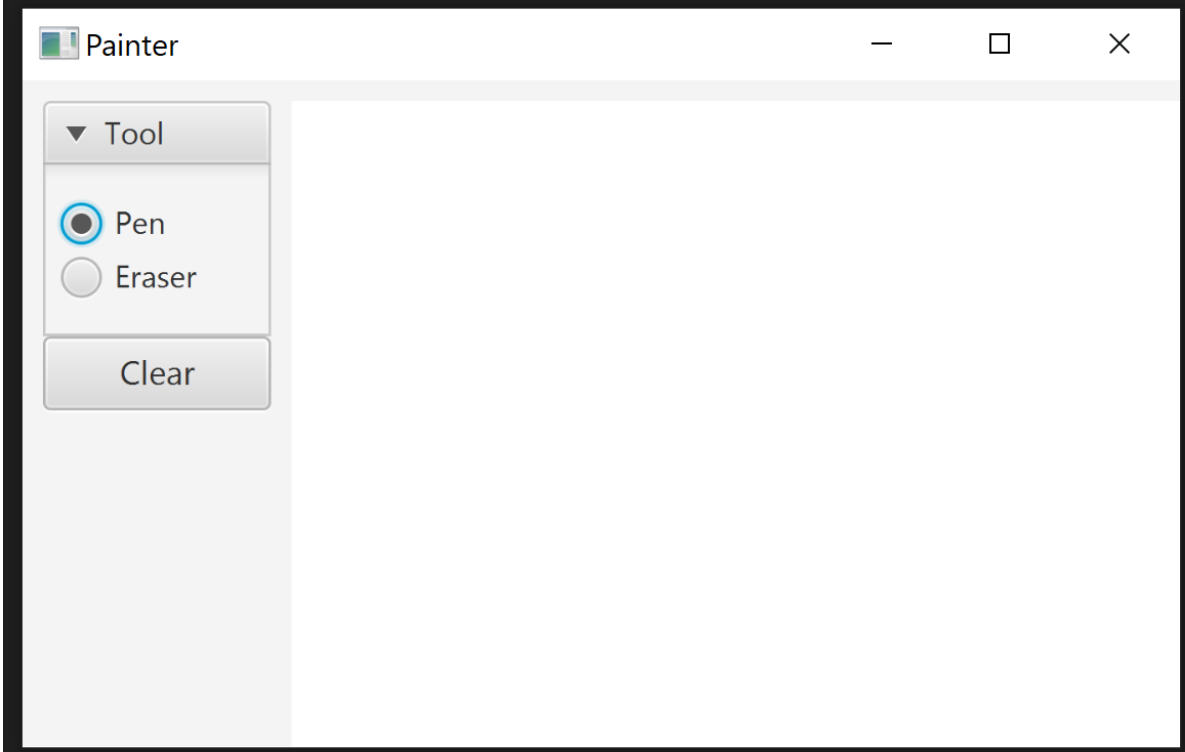
```java
public MediaStore(Media media, Cart cart) {
    this.media = media;
    this.cart = cart;

    this.setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));

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

    JLabel cost = new JLabel("" + media.getCost() + " $");
    cost.setAlignmentX(CENTER_ALIGNMENT);

    JPanel container = new JPanel();
    container.setLayout(new FlowLayout(FlowLayout.CENTER));

    btnAddToCart = new JButton(text:"Add to cart");
    btnAddToCart.addActionListener(e -> addToCartAction());
    container.add(btnAddToCart);
    btnAddToCart.addActionListener(btnListener);
    if(media instanceof Playable) {
        btnPlay = new JButton(text:"Play");
        container.add(btnPlay);
        btnPlay.addActionListener(btnListener);
    }


    this.add(Box.createVerticalGlue());
    this.add(title);
    this.add(cost);
    this.add(Box.createVerticalGlue());
    this.add(container);

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

private void addToCartAction() {
    try {
        cart.addMedia(media);
    } catch (LimitExceededException e) {
        JOptionPane.showMessageDialog(this, message:"Cart is full! Cannot add more items.", title:"Error", JOptionPane.ERROR_MESSAGE);
    }
}
```



4 JavaFX API

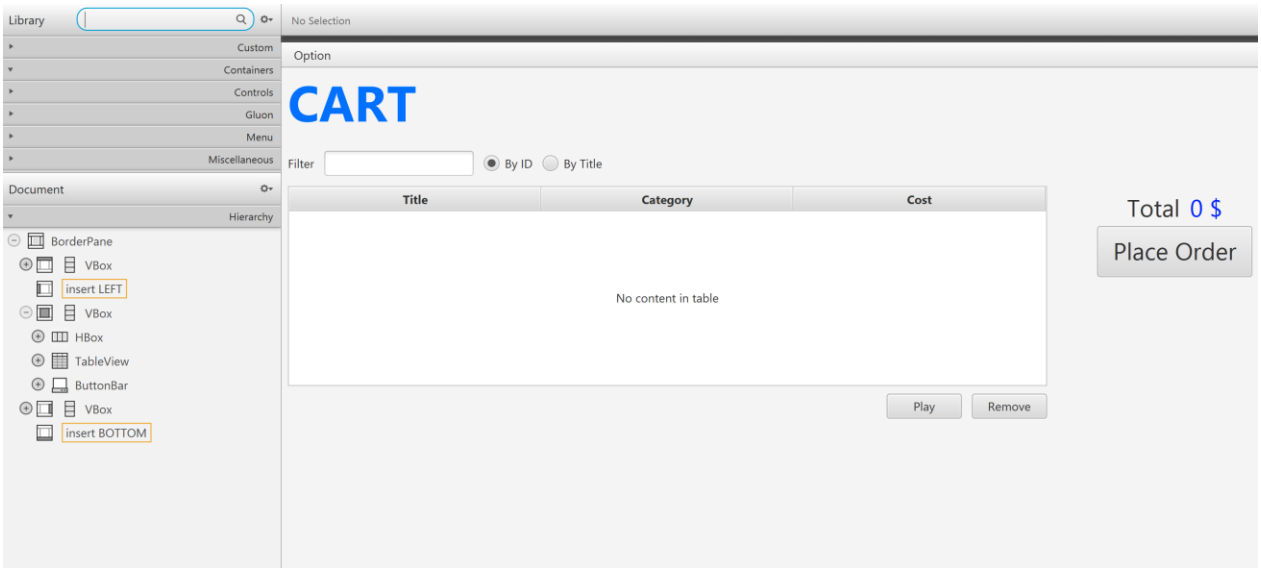

Library
Custom
Containers
Accordion
Controls
Gluon
Menu
Miscellaneous
Shapes
Charts
3D
Document
Hierarchy
BorderPane
insert TOP
VBox
Pane
insert RIGHT
insert BOTTOM
No Selection
Tool
Pen
Eraser
Clear
Inspector
Properties
Layout
Code
No Selection

```java
package hust.soict.dsai.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(name:"/hust/soict/dsai/javafx/Painter.fxml"));

        Scene scene = new Scene(root);
        stage.setTitle(value:"Painter");
        stage.setScene(scene);
        stage.show();
    }

    Run | Debug
    public static void main(String[] args) {
        launch(args);
    }
}
```



```java
package hust.soict.dsai.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.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;


public class PainterController {

    private int color;

    @FXML
    private ToggleGroup identity;

    @FXML
    void chooseOption(ActionEvent event) {
        String button = ((RadioButton)event.getSource()).getText();
        if (button.equals(anObject:"Pen")) {
            System.out.println(x:"1");
            color = 1;
        }else {
            System.out.println(x:"0");
            color = 0;
        }
    }

    @FXML
    private Pane drawingAreaPane;

    @FXML
    void clearButtonPressed(ActionEvent event) {
        drawingAreaPane.getChildren().clear();
    }

    @FXML
    void drawAreaMouseDragged(MouseEvent event) {
        if (color == 1 && event.getX()>=0) {
            Circle newCircle = new Circle(event.getX(), event.getY(), radius:4.0, Color.BLACK);
            drawingAreaPane.getChildren().add(newCircle);
        }else if(color == 0 && event.getX()>=0){
            Circle newCircle = new Circle(event.getX(), event.getY(), radius:4.0, Color.WHITE);
            drawingAreaPane.getChildren().add(newCircle);
        }
    }

}
```

5.Setting up the View Cart Screen with ScreenBuilder



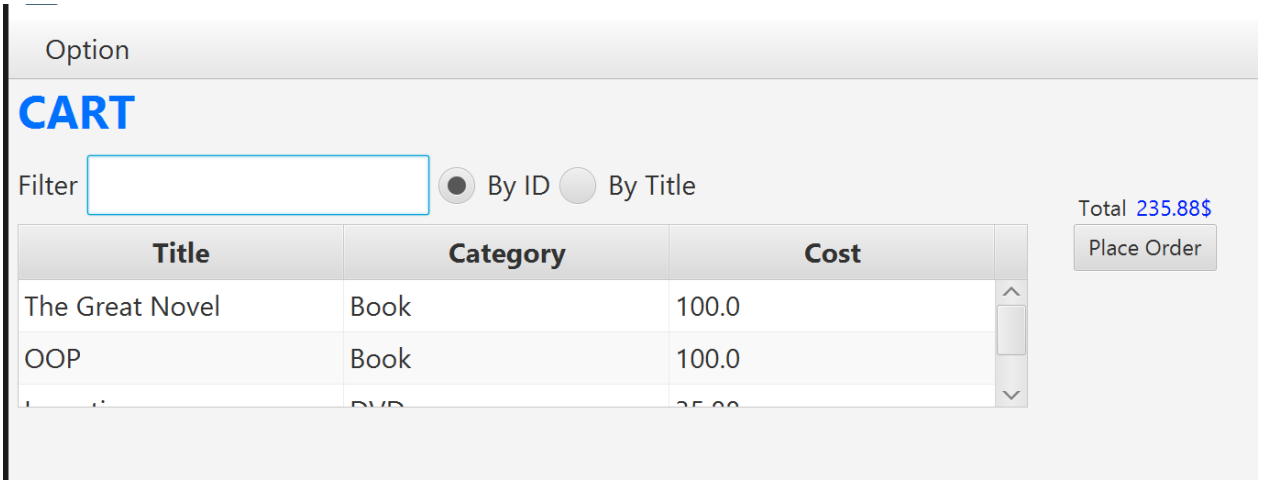
6.Integrating JavaFX into Swing application – The JFXPanel class

```java
package hust.soict.dsai.aims.screen;

import hust.soict.dsai.aims.cart.Cart;
import javafx.application.Platform;
import javafx.embed.swing.JFXPanel;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javax.swing.*;
import java.awt.*;
import java.io.IOException;

public class CartScreen extends JFrame {
    private Cart cart;
    private ScreenManager manager;

    public CartScreen(Cart cart, ScreenManager manager) {
        this.cart = cart;
        this.manager = manager;
        JFXPanel fxPanel = new JFXPanel();
        this.add(fxPanel);
        this.setTitle(title:"Cart");
        this.setVisible(b:true);
        this.setSize(new Dimension(width:1024, height:768));
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        Platform.runLater(new Runnable() {

            @Override
            public void run() {
                // TODO Auto-generated method stub
                try {
                    FXMLLoader loader = new FXMLLoader(getClass().getResource(name:"/hust/soict/dsai/aims/screen/cart.fxml"));
                    CartScreenController controller = new CartScreenController(cart,manager);
                    loader.setController(controller);
                    Parent root = loader.load();
                    Scene scene = new Scene(root);
                    fxPanel.setScene(scene);
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

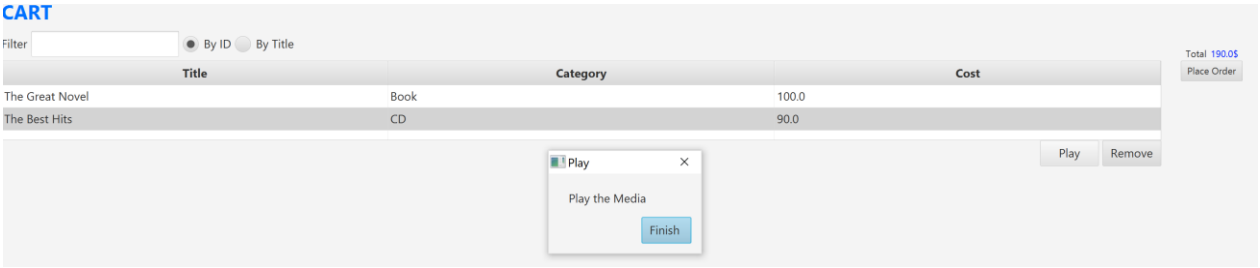
7.View the items in cart – JavaFX's data-driven UI

Update to ObservableList

```java
public class Cart {

    public static final int MAX_NUMBERS_ORDERED = 20;
    private ObservableList<Media> itemsOrdered = FXCollections.observableArrayList();

    public ObservableList<Media> getItemsOrdered() {
        return itemsOrdered;
    }
```



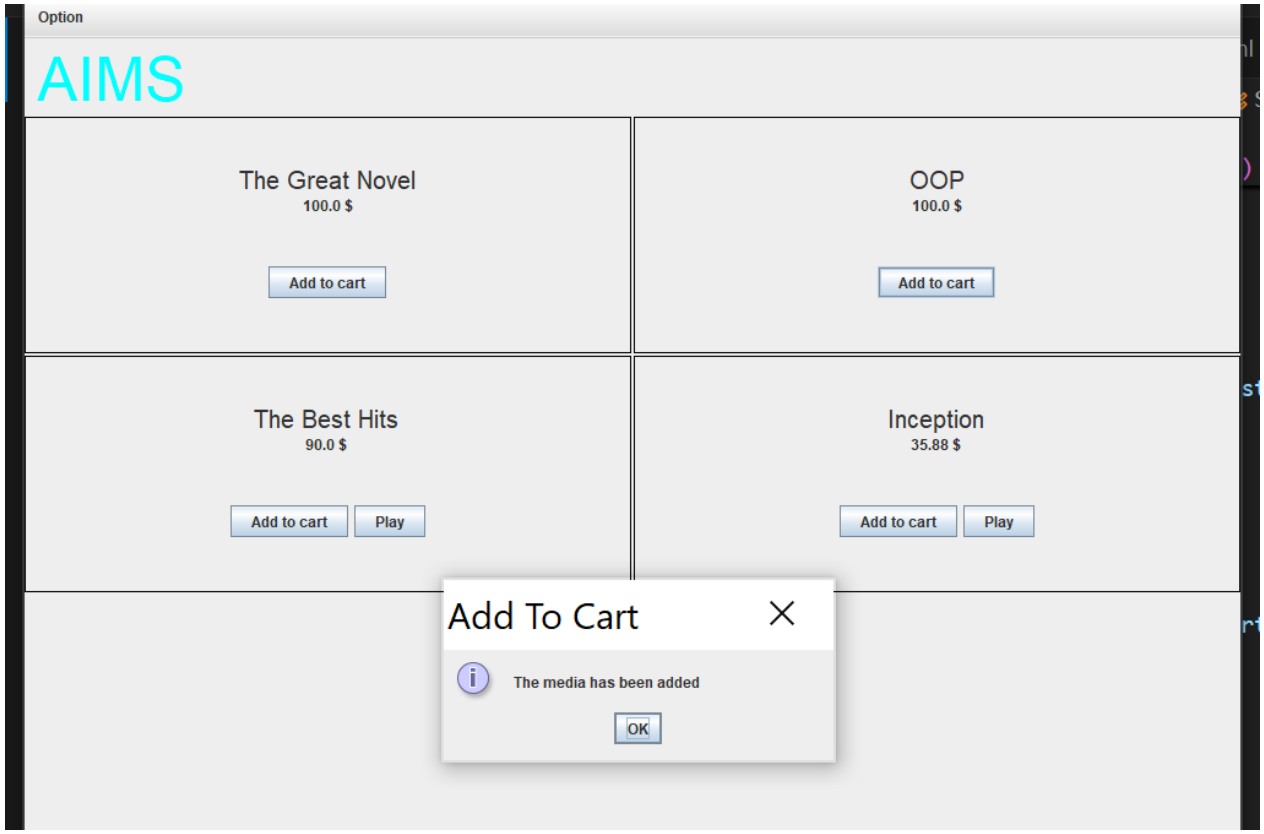
8.Updating buttons based on selected item in TableView – ChangeListener

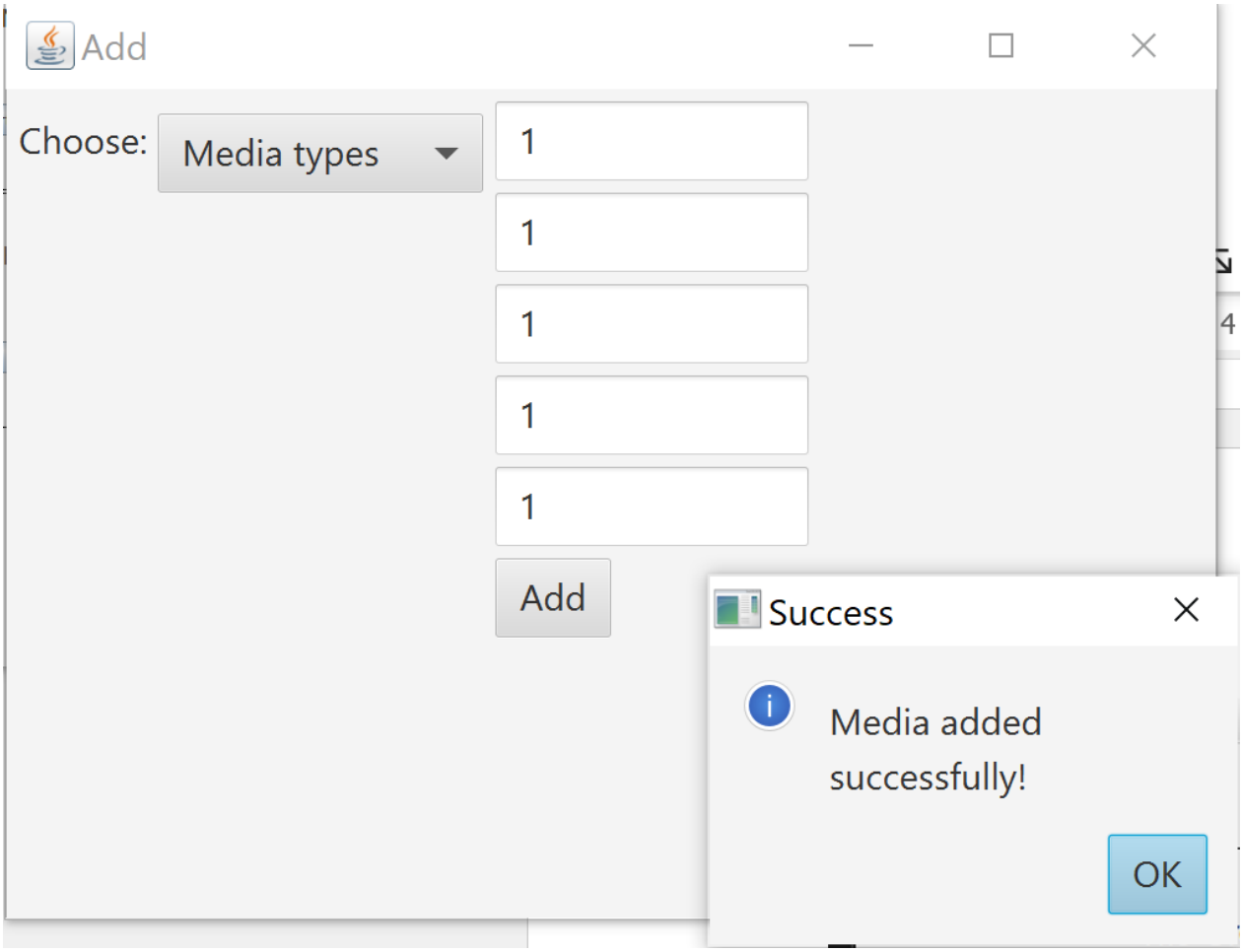


9.Deleting a media

```
@FXML
public void btnRemovePressed(ActionEvent actionEvent) {
    Media media = tblMedia.getSelectionModel().getSelectedItem();
    cart.removeMedia(media);
}
```

10.Complete the Aims GUI application


Option
AIMS
The Great Novel
100.0 $
Add to cart
OOP
100.0 $
Add to cart
The Best Hits
90.0 $
Add to cart
Play
Inception
35.88 $
Add to cart
Play
Add To Cart
The media has been added
OK



UpdateStoreScreen

```java
package hust.soict.dsai.aims.screen;

import hust.soict.dsai.aims.store.Store;
import javafx.application.Platform;
import javafx.embed.swing.JFXPanel;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javax.swing.*;
import java.awt.*;
import java.io.IOException;

public class UpdateStoreScreen extends JFrame {
    private Store store;
    private ScreenManager manager;

    public UpdateStoreScreen(Store store, ScreenManager manager) {
        super();
        this.store = store;
        this.manager = manager;
        JFXPanel fxPanel = new JFXPanel();
        this.add(fxPanel);
        this.setTitle(title:"Add");
        this.setVisible(b:true);
        this.setSize(new Dimension(width:1024, height:768));
        this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        Platform.runLater(new Runnable() {

            @Override
            public void run() {
                // TODO Auto-generated method stub
                try {
                    FXMLLoader loader = new FXMLLoader(getClass().getResource(name:"/hust/soict/dsai/aims/screen/updateStore.fxml"));
                    UpdateStoreController controller = new UpdateStoreController(store);
                    loader.setController(controller);
                    Parent root = loader.load();
                    Scene scene = new Scene(root);
                    fxPanel.setScene(scene);
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

UpdateStoreController

```java
package hust.soict.dsai.aims.screen;

import hust.soict.dsai.aims.media.DigitalVideoDisc;
import hust.soict.dsai.aims.media.Media;
import hust.soict.dsai.aims.store.Store;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.Alert;
import javafx.scene.control.ComboBox;
import javafx.scene.control.TextField;
import java.net.URL;
import java.util.ResourceBundle;


public class UpdateStoreController implements Initializable {
    private Store store;

    public UpdateStoreController() {
        super();
    }

    public UpdateStoreController(Store store) {
        super();
        this.store = store;
    }
    @FXML
    private TextField category;

    @FXML
    private TextField cost;

    @FXML
    private TextField director;

    @FXML
    private TextField length;

    @FXML
    private TextField title;

    @FXML
    private ComboBox<String> listMedia = new ComboBox<String>();
```

```java
@FXML
void btAddPressed(ActionEvent event) {
    try {
        String mediaTitle = title.getText();
        String mediaCategory = category.getText();
        String mediaDirector = director.getText();
        String mediaCostText = cost.getText();
        String mediaLengthText = length.getText();

        if (mediaTitle.isEmpty() || mediaCategory.isEmpty() || mediaCostText.isEmpty()) {
            showAlert(title:"Error", message:"Title, Category, and Cost are required fields!", Alert.AlertType.ERROR);
            return;
        }

        float mediaCost = Float.parseFloat(mediaCostText);
        int mediaLength = mediaLengthText.isEmpty() ? 0 : Integer.parseInt(mediaLengthText);

        Media media = new DigitalVideoDisc(mediaTitle, mediaCategory, mediaDirector, mediaLength, mediaCost);

        this.store.addMedia(media);

        showAlert(title:"Success", message:"Media added successfully!", Alert.AlertType.INFORMATION);

        clearInputFields();

    } catch (NumberFormatException e) {
        showAlert(title:"Error", message:"Cost and Length must be valid numbers.", Alert.AlertType.ERROR);
    } catch (Exception e) {
        showAlert(title:"Error", "An unexpected error occurred: " + e.getMessage(), Alert.AlertType.ERROR);
    }
}

private void showAlert(String title, String message, Alert.AlertType type) {
    Alert alert = new Alert(type);
    alert.setTitle(title);
    alert.setHeaderText(headerText:null);
    alert.setContentText(message);
    alert.showAndWait();
}

private void clearInputFields() {
    title.clear();
    category.clear();
    director.clear();
    cost.clear();
    length.clear();
}

@Override
public void initialize(URL arg0, ResourceBundle arg1) {

}
```

ScreenManager

```java
public class ScreenManager {

    private static JFrame storeScreen;
    private static JFrame cartScreen;

    private Cart cart;
    private Store store;

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

        showStoreScreen();
    }

    public void showStoreScreen() {
        if (storeScreen == null) {
            storeScreen = new StoreScreen(this.store,this.cart,this);
        }
        showScreen(storeScreen);
    }

    public void showCartScreen() {
        if (cartScreen == null) {
            cartScreen = new CartScreen(this.cart, this);
        }
        showScreen(cartScreen);
    }

    public void showUpdateScreen() {
        new UpdateStoreScreen(this.store,this);
    }

    private void showScreen(JFrame screen) {
        if (currentScreen != null) {
            currentScreen.setVisible(b:false);
        }
        currentScreen = screen;
        currentScreen.setVisible(b:true);
    }

    Run | Debug
    public static void main(String[] args) {
        Store store = new Store();
        Cart cart = new Cart();
        // Book
        Book initBook = new Book(title:"The Great Novel", category:"Book", cost:100.00f);
        initBook.addAuthor( authorName:"Jane Smith");
        initBook.addAuthor(authorName:"John D");
        // Book
        Book initBook2 = new Book(title:"OOP", category:"Book", cost:100.00f);
        initBook.addAuthor( authorName:"Walpole");

        // DVD
        DigitalVideoDisc initDVD = new DigitalVideoDisc(title:"Inception", category:"DVD",
                director:"Christopher Nolan", length:150, cost:35.88f);

        // CD
        Track track1 = new Track(title:"Song 1", length:3);
        Track track2 = new Track(title:"Song 2", length:4);
        CompactDisc initCD = new CompactDisc(title:"The Best Hits", category:"CD", length:30,
                cost:90.00f, artist:"Many Composers");
        initCD.addTrack(track1);
        initCD.addTrack(track2);

        store.addMedia(initBook);
        store.addMedia(initBook2);
        store.addMedia(initCD);
        store.addMedia(initDVD);

        new ScreenManager(cart, store);
    }
}
```

```java
public class ScreenManager {

    private static JFrame storeScreen;
    private static JFrame cartScreen;

    private Cart cart;
    private Store store;

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

        showStoreScreen();
    }

    public void showStoreScreen() {
        if (storeScreen == null) {
            storeScreen = new StoreScreen(this.store,this.cart,this);
        }
        showScreen(storeScreen);
    }

    public void showCartScreen() {
        if (cartScreen == null) {
            cartScreen = new CartScreen(this.cart, this);
        }
        showScreen(cartScreen);
    }

    public void showUpdateScreen() {
        new UpdateStoreScreen(this.store,this);
    }

    private void showScreen(JFrame screen) {
        if (currentScreen != null) {
            currentScreen.setVisible(b:false);
        }
        currentScreen = screen;
        currentScreen.setVisible(b:true);
    }

    Run | Debug
    public static void main(String[] args) {
        Store store = new Store();
        Cart cart = new Cart();
        // Book
        Book initBook = new Book(title:"The Great Novel", category:"Book", cost:100.00f);
        initBook.addAuthor( authorName:"Jane Smith");
        initBook.addAuthor(authorName:"John D");
        // Book
        Book initBook2 = new Book(title:"OOP", category:"Book", cost:100.00f);
        initBook.addAuthor( authorName:"Walpole");

        // DVD
        DigitalVideoDisc initDVD = new DigitalVideoDisc(title:"Inception", category:"DVD",
                director:"Christopher Nolan", length:150, cost:35.88f);

        // CD
        Track track1 = new Track(title:"Song 1", length:3);
        Track track2 = new Track(title:"Song 2", length:4);
        CompactDisc initCD = new CompactDisc(title:"The Best Hits", category:"CD", length:30,
                cost:90.00f, artist:"Many Composers");
        initCD.addTrack(track1);
        initCD.addTrack(track2);

        store.addMedia(initBook);
        store.addMedia(initBook2);
        store.addMedia(initCD);
        store.addMedia(initDVD);

        new ScreenManager(cart, store);
    }
}
```

11.Check all the previous source codes to catch/handle/delegate runtime exceptions

```java
public void addMedia(Media media) throws LimitExceededException {

    if (itemsOrdered.size() < MAX_NUMBERS_ORDERED) {
        itemsOrdered.add(media);
        System.out.println(x:"Media added");
    }
    else {
        throw new LimitExceededException(explanation:"Maximum amount reached!");
    }
}
```

12.Create a class which inherits from Exception

```java
package hust.soict.dsai.aims.exception;

public class PlayerException extends RuntimeException {
    public PlayerException() {
        // TODO Auto-generated constructor stub
    }

    public PlayerException(String message) {
        super(message);
        // TODO Auto-generated constructor stub
    }

    public PlayerException(Throwable cause) {
        super(cause);
        // TODO Auto-generated constructor stub
    }

    public PlayerException(String message, Throwable cause) {
        super(message, cause);
        // TODO Auto-generated constructor stub
    }

    public PlayerException(String message, Throwable cause, boolean enableSuppression, boolean writableStackTrace) {
        super(message, cause, enableSuppression, writableStackTrace);
        // TODO Auto-generated constructor stub
    }
}
```

CD

```java
@Override
public void play() throws PlayerException{
    System.out.println("CompactDisc Artist: " + this.getArtist());
    System.out.println("Total length: " + this.getLength());

    if (this.getLength() > 0) {
        System.out.println("Compactdisc: " + this.getTitle());
        System.out.println("CompactDisc Artist: " + this.getArtist());
        System.out.println("Total length: " + this.getLength());
        java.util.Iterator iter = tracks.iterator();
        Track nextTrack;
        while (iter.hasNext()) {
            nextTrack = (Track) iter.next();
            try {
                nextTrack.play();
            }
            catch(PlayerException e ) {
                throw e;
            }
        }
    }
    else {
        throw new PlayerException(message:"Error: CD length is non-positive!");
    }

    System.out.println(x:"----------------Play All Tracks----------------");
    for (Track track: tracks) {
        track.play();
    }
}
```

DVD

```java
@Override
public void play() throws PlayerException{
    if (this.getLength() > 0) {
        System.out.println("Playing DVD: " + this.getTitle());
        System.out.println("DVD length: " + this.getLength());
    }
    else {throw new PlayerException(message:"Error: DVD length is non-positive");}
}
```

Track

```java
@Override
public void play() throws PlayerException {
    if (this.getLength() > 0) {
        System.out.println("Playing track: " + this.getTitle());
        System.out.println("Track length: " + this.getLength());
    }
    else {throw new PlayerException(message:"Error: Track length is non-positive!");}
}
```

13. Update Aims class

```java
switch (choice) {
    case 1:
        cart.addMedia(media);
        System.out.println(x:"Media added to cart.");
        break;
    case 2:
        if (media instanceof Playable) {
            if(media instanceof DigitalVideoDisc){
                try {
                    ((DigitalVideoDisc) media).play();
                } catch (PlayerException e) {
                    e.getMessage();
                    e.toString();
                    e.printStackTrace();
                }
            }
            else if(media instanceof CompactDisc){
                try {
                    ((CompactDisc)media).play();
                } catch (PlayerException e) {
                    e.getMessage();
                    e.toString();
                    e.printStackTrace();
                }
            }
        } else {
            System.out.println(x:"This media cannot be played.");
        }
        break;
    case 0:
        break;
    default:
        System.out.println(x:"Invalid choice! Please choose a number between 0-2.");
}
```

```java
if (media != null) {
    if (media instanceof Playable) {
        if(media instanceof DigitalVideoDisc){
            try {
                ((DigitalVideoDisc) media).play();
            } catch (PlayerException e) {
                e.getMessage();
                e.toString();
                e.printStackTrace();
            }
        }
        else if(media instanceof CompactDisc){
            try {
                ((CompactDisc)media).play();
            } catch (PlayerException e) {
                e.getMessage();
                e.toString();
                e.printStackTrace();
            }
        }
    } else {
        System.out.println(x:"This media cannot be played.")
    }
} else {
    System.out.println(x:"Media not found in the store.");
}
```