# ĐẠI HỌC BÁCH KHOA HÀ NỘI

# TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

-------- * --------


# BÁO CÁO THỰC HÀNH LAP 5

# MÔN HỌC: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

**Giảng viên hướng dẫn: Lê Thị Hoa**

**Sinh viên thực hiện:**

**Phan Thế Anh** **20204941**

# Content

# 1. Swing components

## 1.1 AWTAccumulator

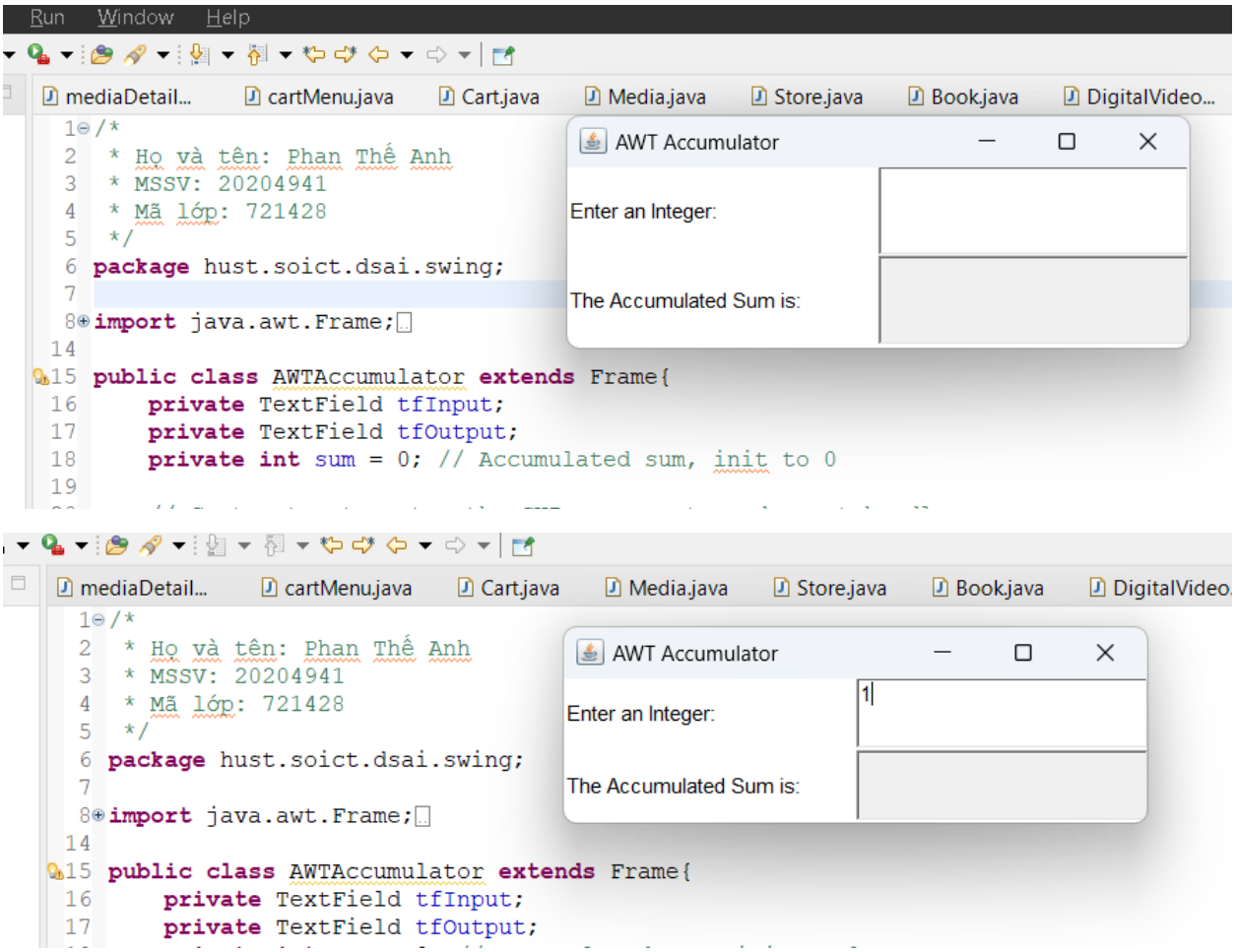
### 1.1.1 Create class AWTAccumulator

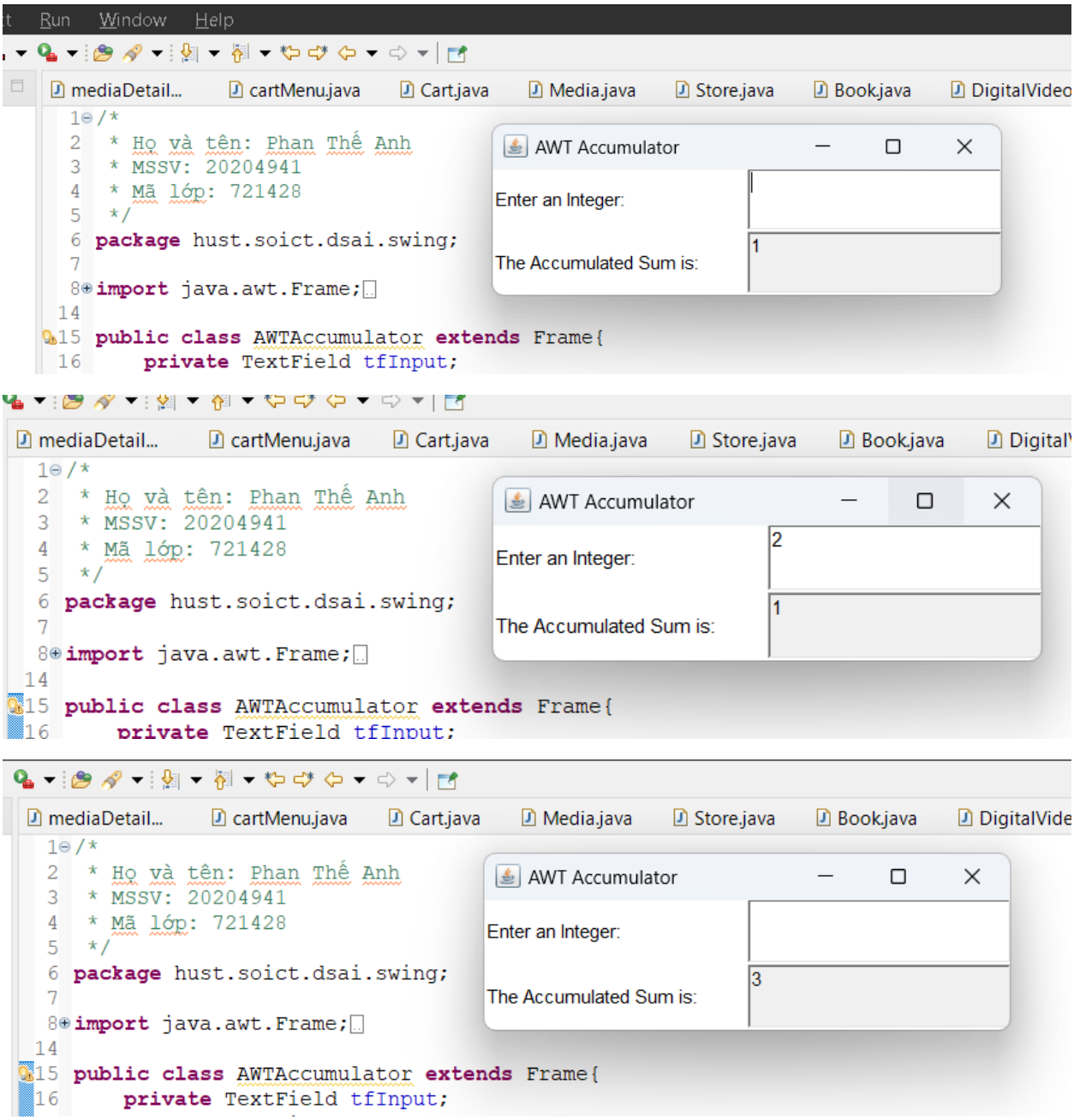
Code

```java
/*
 * Họ và tên: Phan Thế Anh
 * MSSV: 20204941
 * Mã lớp: 721428
 */
package hust.soict.dsai.swing;

import java.awt.Frame;

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

    // Contructor to setup the GUI components and event handlers
    public AWTAccumulator() {
        // TODO Auto-generated constructor stub
        setLayout(new GridLayout(2, 2)); // Set layout

        add(new Label("Enter an Integer: ")); // Add label

        // Input
        tfInput = new TextField(10); // Create field for tfInput
        add(tfInput); // Add Input
        tfInput.addActionListener(new TFInputListener()); // Add input from user

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

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

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

```

```
44
45    public static void main(String[] args) {
46        new AWTAccumulator();
47    }
48
49    public class TFInputListener implements ActionListener {
50        @Override
51        public void actionPerformed(ActionEvent e) {
52            // TODO Auto-generated method stub
53            int numberIn = Integer.parseInt(tfInput.getText()); // Number from user
54            sum += numberIn; // calculate sum
55            tfInput.setText("");
56            tfOutput.setText(sum + "");
57        }
58    }
59 }
60
```

Result






## 1.1.2 Explanation

- In AWT, the top-level container is `Frame`, which is inherited by the application class.
- In the constructor, we set up the GUI components in the `Frame` object and the event-handling:
  - In line 23, the layout of the frame is set as `GridLayout`
  - In line 55, we add the first component to our `Frame`, an anonymous `Label`
  - In line 28-30, we add a `TextField` component to our `Frame`, where the user will enter values. We add a listener which takes this `TextField` component as the source, using a named inner class.
  - In line 32, we add another anonymous `Label` to our `Frame`

- In line 35 – 37, we add a `TextField` component to our `Frame`, where the accumulated sum of entered values will be displayed. The component is set to read-only in line 24.
- In line 40 – 42, the title & size of the `Frame` is set, and the `Frame` visibility is set to true, which shows the `Frame` to us.

- In the listener class (line 49 - 56), the `actionPerformed()` method is implemented, which handles the event when the user hit "Enter" on the source `TextField`.
  - In line 53-56, the entered number is parsed, added to the sum, and the output `TextField`'s text is changed to reflect the new sum.
- In the `main()` method, we invoke the `AWTAccumulator` constructor to set up the GUI

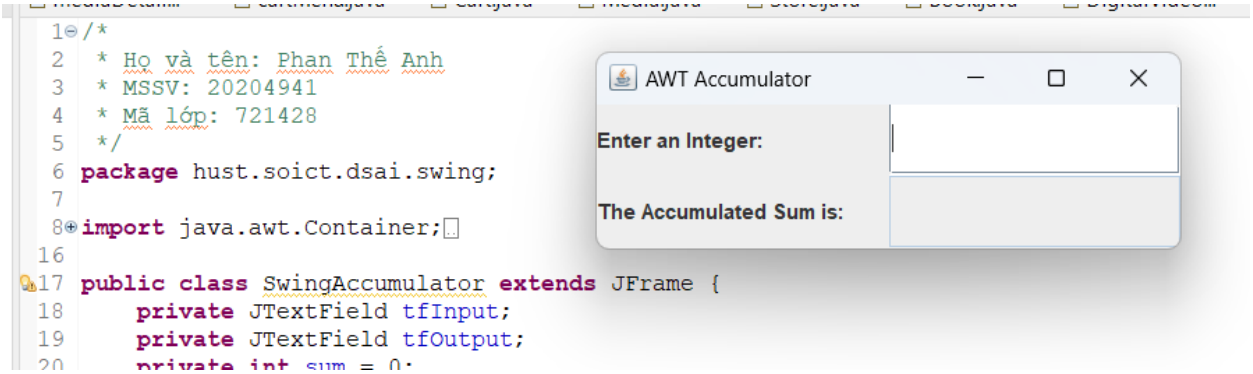
## 1.2 SwingAccumulator

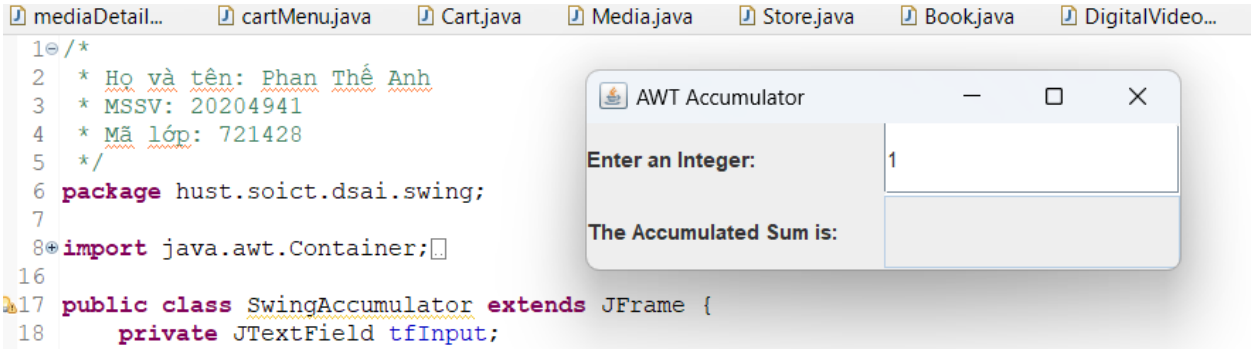
### 1.2.1 Create class `SwingAccumulator`

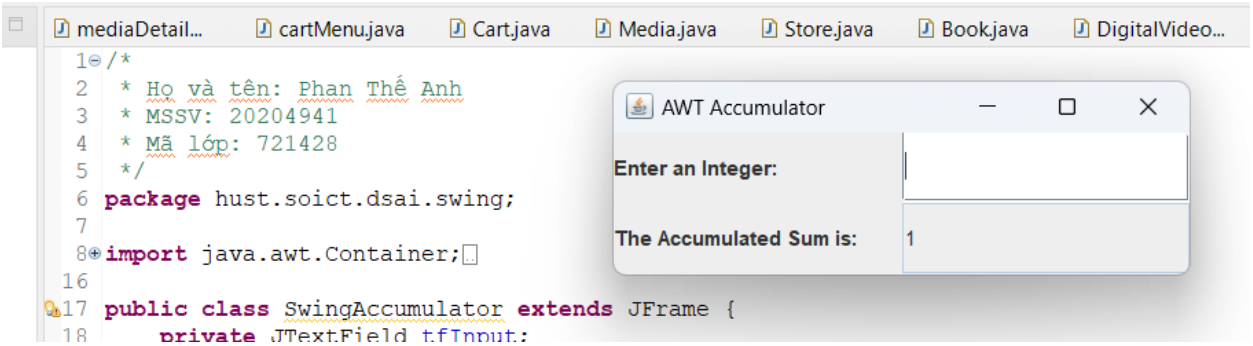
Code

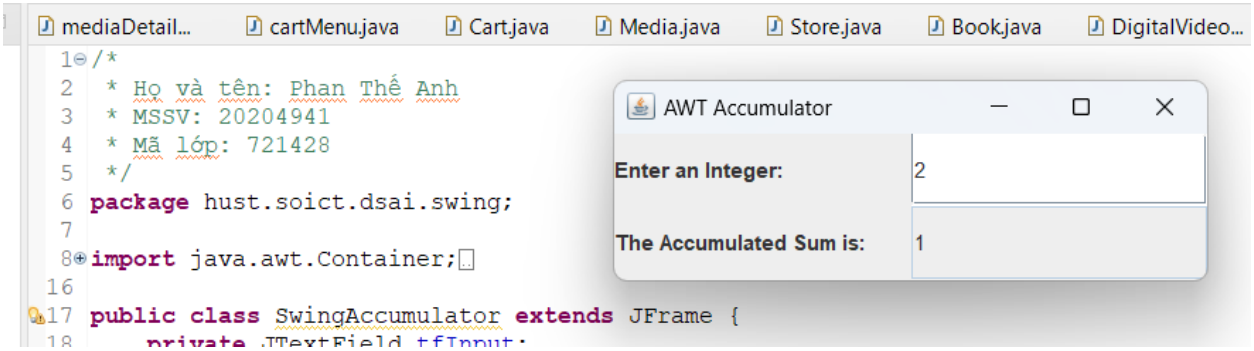
```java
/*
 * Họ và tên: Phan Thế Anh
 * MSSV: 20204941
 * Mã lớp: 721428
 */
package hust.soict.dsai.swing;

import java.awt.Container;

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

    // Constructor to setup the GUI components and event handlers
    public SwingAccumulator() {
        // TODO Auto-generated constructor stub
        Container cp = getContentPane();
        cp.setLayout(new GridLayout(2, 2));

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

        // Input
        tfInput = new JTextField(10);
        cp.add(tfInput);
        tfInput.addActionListener(new TFInputListener());

        cp.add(new JLabel("The Accumulated Sum is: "));

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

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

    public static void main(String[] args) {
        new SwingAccumulator();
    }
    public class TFInputListener implements ActionListener {
        @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            int numberIn = Integer.parseInt(tfInput.getText()); // Number from user
            sum += numberIn; // calculate sum
            tfInput.setText("");
            tfOutput.setText(sum + "");
        }
    }
}
```

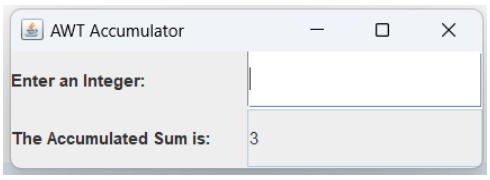
Result


/*
* Họ và tên: Phan Thế Anh
* MSSV: 20204941
* Mã lớp: 721428
*/
package hust.soict.dsai.swing;
import java.awt.Container;
public class SwingAccumulator extends JFrame {
private JTextField tfInput;
private JTextField tfOutput;
AWT Accumulator
Enter an Integer:
The Accumulated Sum is:


mediaDetail...
cartMenu.java
Cart.java
Media.java
Store.java
Book.java
DigitalVideo...
/*
* Họ và tên: Phan Thế Anh
* MSSV: 20204941
* Mã lớp: 721428
*/
package hust.soict.dsai.swing;
import java.awt.Container;
public class SwingAccumulator extends JFrame {
private JTextField tfInput;
AWT Accumulator
Enter an Integer:
1
The Accumulated Sum is:


mediaDetail...
cartMenu.java
Cart.java
Media.java
Store.java
Book.java
DigitalVideo...
/*
* Họ và tên: Phan Thế Anh
* MSSV: 20204941
* Mã lớp: 721428
*/
package hust.soict.dsai.swing;
import java.awt.Container;
public class SwingAccumulator extends JFrame {
AWT Accumulator
Enter an Integer:
The Accumulated Sum is:
1


mediaDetail...
cartMenu.java
Cart.java
Media.java
Store.java
Book.java
DigitalVideo...
/*
* Họ và tên: Phan Thế Anh
* MSSV: 20204941
* Mã lớp: 721428
*/
package hust.soict.dsai.swing;
import java.awt.Container;
public class SwingAccumulator extends JFrame {
AWT Accumulator
Enter an Integer:
2
The Accumulated Sum is:
1


AWT Accumulator
Enter an Integer:
The Accumulated Sum is:
3

### 1.2.2 Explanation

- In Swing, the top-level container is `JFrame` which is inherited by the application class.
- In the constructor, we set up the GUI components in the `JFrame` object and the event-handling:
  - Unlike AWT, the `JComponents` shall not be added onto the top-level container (e.g., `JFrame`, `JApplet`) directly because they are lightweight components. The `JComponents` must be added onto the so-called content-pane of the top-level container. Content-pane is in fact a `java.awt.Container` that can be used to group and layout components.
  - In line 23, we get the content-pane of the top-level container.
  - In line 26, the layout of the content-pane is set as `GridLayout`
  - In line 28, we add the first component to our content-pane, an anonymous `JLabel`
  - In line 31-33, we add a `JTextField` component to our content-pane, where the user will enter values. We add a listener which takes this `JTextField` component as the source.
  - In line 35, we add another anonymous `JLabel` to our content-pane
  - In line 37 – 40, we add a `JTextField` component to our content-pane, where the accumulated sum of entered values will be displayed. The component is set to read-only in line 39.
  - In line 43 – 45, the title & size of the `JFrame` is set, and the Frame visibility is set to true, which shows the `JFrame` to us.
- In the listener class (line 51 - 60), the code for event-handling is exactly like the `AWTAccumulator`.
- In the `main()` method, we invoke the `SwingAccumulator` constructor to set up the GUI

## 1.3 Compare Swing and AWT elements

Programming with AWT and Swing is quite similar (similar elements including container/components, event-handling). However, there are some differences that you need to note:

- The top-level containers in Swing and AWT
- The class name of components in AWT and corresponding class's name in Swing

# 2. Organizing Swing components with Layout Managers

## 2.1 Swing top-level and secondary-level containers

A container is used to hold components. A container can also hold containers because it is a (subclass of) component. Swing containers are divided into top-level and secondary-level containers:

- Top-level containers:
  - `JFrame`: used for the application's main window (with an icon, a title, minimize/maximize/close buttons, an optional menu-bar, and a content-pane)
  - `JDialog`: used for secondary pop-up window (with a title, a close button, and a content-pane).
  - `JApplet`: used for the applet's display-area (content-pane) inside a browser's window
- Secondary-level containers can be used to group and layout relevant components (most commonly used is `JPanel`)

## 2.2 Using JPanel as secondary-level container to organize components

### 2.2.1 Create class `NumberGrid`

This class allows us to input a number digit-by-digit from a number grid into a text field display. We can also delete the latest digit or delete the entire number and start over.

```java
/*
 * Họ và tên: Phan Thế Anh
 * MSSV: 20204941
 * Mã lớp: 721428
 */
package hust.soict.dsai.swing;

import java.awt.BorderLayout;

public class NumberGrid extends JFrame {
    private JButton[] btnNumbers = new JButton[10]; // Array of button
    private JButton btnDeltte, btnReset; // button delete and reset
    private JTextField tfDisplay; // The top display

    public NumberGrid() {
        tfDisplay = new JTextField(); // The top display
        tfDisplay.setComponentOrientation(
                ComponentOrientation.RIGHT_TO_LEFT);
        JPanel panelButtons = new JPanel(new GridLayout(4,3)); // layout panel buttons
        addButtons(panelButtons);

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

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

The class has several attributes:

- The `btnNumbers` array for the digit buttons
- The `btnDelete` for the DEL button
- The `btnReset` for the C button
- The `tfDisplay` for the top display

In the constructor, we add two components to the content pane of the `JFrame`:

- A `JTextField` for the display text field
- A `JPanel`, which will group all of the buttons and put them in a grid layout

### 2.2.2 Adding buttons

We add the buttons to the `panelButtons` in the `addButtons()` method:

```
/*
 * Họ và tên: Phan Thế Anh
 * MSSV: 20204941
 * Mã lớp: 721428
 */
// Create medthod to add buttons
void addButtons(JPanel panelButton) {
    ButtonListener btnListener = new ButtonListener();
    // Create button "1" ~ "9"
    for (int i = 1; i <= 9; i++) {
        btnNumbers[i] = new JButton(""+i); // Create button
        panelButton.add(btnNumbers[i]);
        btnNumbers[i].addActionListener(btnListener);
    }
    // Create button "DEL"
    btnDelete = new JButton("DEL");
    panelButton.add(btnDelete);
    btnDelete.addActionListener(btnListener);

    // Create button "0"
    btnNumbers[0] = new JButton("0");
    panelButton.add(btnNumbers[0]);
    btnNumbers[0].addActionListener(btnListener);

    // Create button "C"
    btnReset = new JButton("C");
    panelButton.add(btnReset);
    btnReset.addActionListener(btnListener);
}
```

### 2.2.3 Complete inner class `ButtonListener`

In the `actionPerformed()` method, we will handle the button pressed event. Since we have many sources, we need to determine which source is firing the event (which button is pressed) and handle each case accordingly (change the text of the display text field). Here, we have three cases:

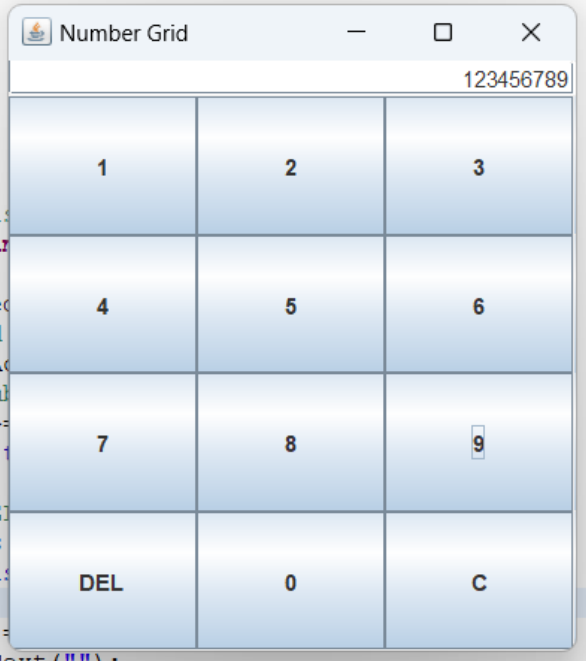
- A digit button: a digit is appended to the end
- DEL button: delete the last digit
- C button: clears all digits

Code

```
69
70      /*
71       * Họ và tên: Phan Thế Anh
72       * MSSV: 20204941
73       * Mã lớp: 721428
74       */
75      // Create inner class ButtonListener
76      private class ButtonListener implements ActionListener {
77          @Override
78          public void actionPerformed(ActionEvent e) {
79              // TODO Auto-generated method stub
80              String button = e.getActionCommand(); // get action of user
81              // Check button is number
82              if (button.charAt(0) >= '0' && button.charAt(0) <= '9') {
83                  tfDisplay.setText(tfDisplay.getText() + button);
84              }
85              // Check button is "DEL"
86              else if (button.equals("DEL")) {
87                  String text = tfDisplay.getText();
88                  // Check is null
89                  if (text.length() == 0) {
90                      tfDisplay.setText("");
91                  }
92                  else {
93                      text = text.substring(0, text.length()-1); // Remove element
94                      tfDisplay.setText(text); // Set display
95                  }
96              }
97              // Check button is "c"
98              else {
99                  tfDisplay.setText("");;
100             }
101         }
```

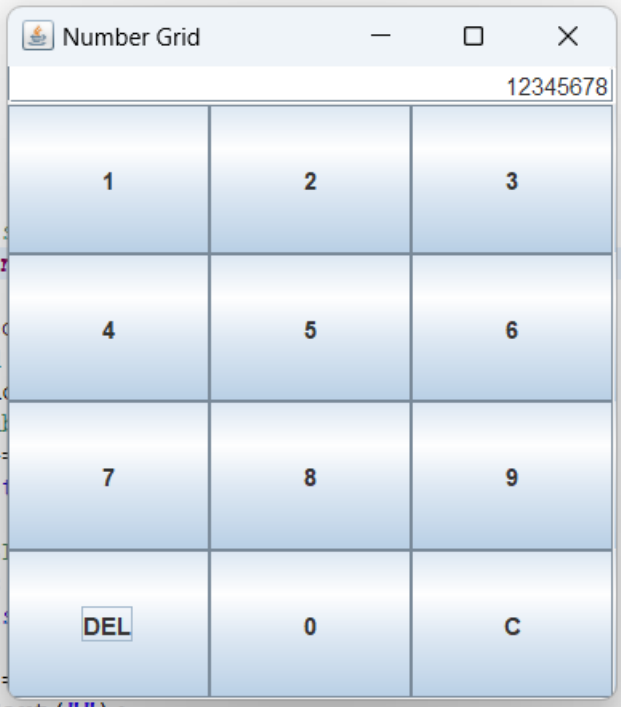
Result

```
        panelButton.add(btnReset);
        btnReset.addActionListener(btnListener);
    }

    /*
     * Họ và tên: Phan Thế Anh
     * MSSV: 20204941
     * Mã lớp: 721428
     */
    // Create inner class ButtonLis
    private class ButtonListener im
        @Override
        public void actionPerformed
            // TODO Auto-generated
            String button = e.getAc
            // Check button is numb
            if (button.charAt(0) >=
                tfDisplay.setText(t
            }
            // Check button is "DEL
            else if (button.equals
                String text = tfDis
                // Check is null
                if (text.length() =
                    tfDisplay.setText("");
```



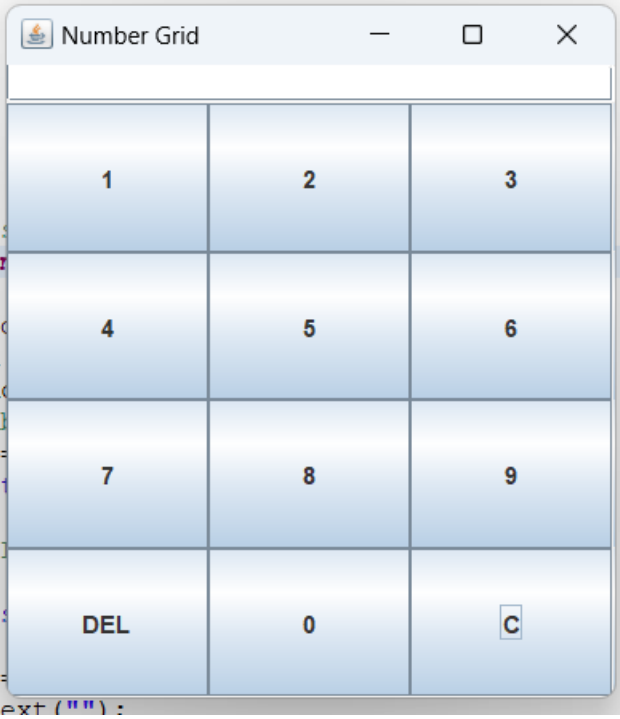
Click “DEL”

```
        panelButton.add(btnReset);
        btnReset.addActionListener(btnListener);
    }

    /*
     * Họ và tên: Phan Thế Anh
     * MSSV: 20204941
     * Mã lớp: 721428
     */
    // Create inner class ButtonLis
    private class ButtonListener ir
        @Override
        public void actionPerforme
            // TODO Auto-generated
            String button = e.getA
            // Check button is numb
            if (button.charAt(0) >=
                tfDisplay.setText(t
            }
            // Check button is "DE
            else if (button.equals
                String text = tfDis
                // Check is null
                if (text.length() =
                    tfDisplay.setText("");
```




Click “C”

```
            btnReset.addActionListener(btnListener);
        }

        /*
         * Họ và tên: Phan Thế Anh
         * MSSV: 20204941
         * Mã lớp: 721428
         */
        // Create inner class ButtonLis
        private class ButtonListener ir
            @Override
            public void actionPerforme
                // TODO Auto-generated
                String button = e.getA
                // Check button is numb
                if (button.charAt(0) >=
                    tfDisplay.setText(t
                }
                // Check button is "DE
                else if (button.equals
                    String text = tfDis
                    // Check is null
                    if (text.length() =
                        tfDisplay.setText("");
```



# 3. Create a graphical user interface for AIMS with Swing

## 3.1 View Store Screen

### 3.1.1 Create the `StoreScreen` class

```java
/*
 * Họ và tên: Phan Thế Anh
 * MSSV: 20204941
 * Mã lớp: 721428
 */
package hust.soict.dsai.aims.screen;

import hust.soict.dsai.aims.store.Store;

public class StoreScreen {
    private Store store; // information on the items in the store to display them
}
```

### 3.1.2 The `NORTH` component

Create the method createNorth(), which will create our NORTH component

```java
    /*
     * Họ và tên: Phan Thế Anh
     * MSSV: 20204941
     * Mã lớp: 721428
     */
    // Create method createNorth()
    JPanel createNorth() {
        JPanel north =  new JPanel();
        north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS)); // Location
        north.add(createMenuBar()); // menu
        north.add(createHeader()); // Header
        return north;
    }
```

Create the method `createMenuBar()`:

```
38
39      /*
40       * Họ và tên: Phan Thế Anh
41       * MSSV: 20204941
42       * Mã lớp: 721428
43       */
44      // Create the method createMenuBar()
45      JMenuBar createMenuBar() {
46          JMenu menu = new JMenu("Option");
47
48          // Display to choose update store
49          JMenu smUpdateStore = new JMenu("Update Store");
50          smUpdateStore.add(new JMenuItem("Add Book"));
51          smUpdateStore.add(new JMenuItem("Add CD"));
52          smUpdateStore.add(new JMenuItem("Add DVD"));
53
54          // display to view store, cart
55          menu.add(smUpdateStore);
56          menu.add(new JMenuItem("View store"));
57          menu.add(new JMenuItem("View cart"));
58
59          JMenuBar menuBar = new JMenuBar();
60          menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
61          menuBar.add(menu);
62          return menuBar;
63
64      }
```

Create the method `createHeader()`:

```
65
66      /*
67       * Họ và tên: Phan Thế Anh
68       * MSSV: 20204941
69       * Mã lớp: 721428
70       */
71      // Create method createHeader();
72      JPanel createHeader() {
73          JPanel header = new JPanel();
74          header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));
75
76          // Create label
77          JLabel title = new JLabel("AISM");
78          title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
79          title.setForeground(Color.CYAN);
80
81          // Create button
82          JButton cart = new JButton("View cart");
83          cart.setPreferredSize(new Dimension(100, 50));
84          cart.setMaximumSize(new Dimension(100, 50));
85
86          // Display
87          header.add(Box.createRigidArea(new Dimension(10, 10)));
88          header.add(title);
89          header.add(Box.createHorizontalGlue());
90          header.add(cart);
91          header.add(Box.createRigidArea(new Dimension(10, 10)));
92
93          return header;
94
```

### 3.1.3 The CENTER component

```
/*
 * Họ và tên: Phan Thế Anh
 * MSSV: 20204941
 * Mã lớp: 721428
 */
// The CENTER component
JPanel createCenter() {

    JPanel center = new JPanel();
    center.setLayout(new GridLayout(3, 3, 2, 2));

    ArrayList<Media> mediaInStore = store.getItemsInStore();
    for (int i = 0; i < 9; i++) {
        MediaStore cell = new MediaStore(mediaInStore.get(i));
        center.add(cell);
    }
    return center;
}
```

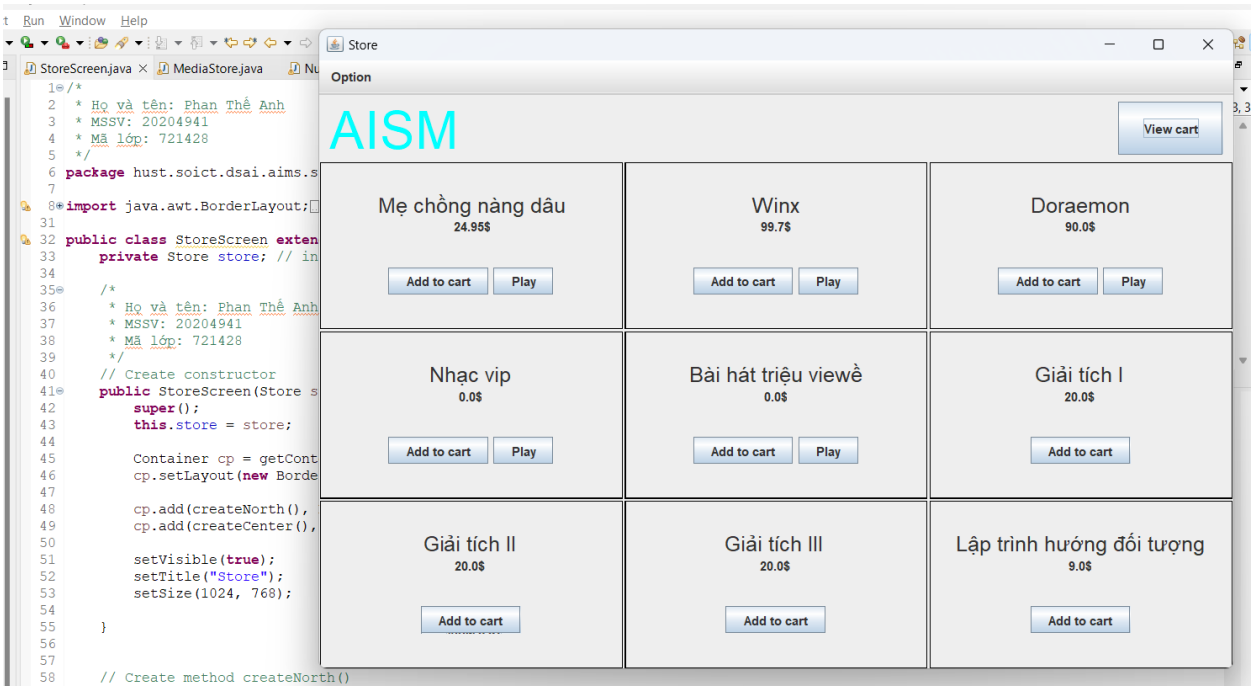
### 3.1.4 The `MediaStore` class

```java
/*
 * Họ và tên: Phan Thế Anh
 * MSSV: 20204941
 * Mã lớp: 721428
 */
package hust.soict.dsai.aims.screen;

import java.awt.BorderLayout;

public class MediaStore extends JPanel{
    private Media media; // attribute

    // Create constructor
    public MediaStore(Media media) {
        super();
        this.media = media;
        this.setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));

        // Design Layout
        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(BOTTOM_ALIGNMENT);

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

        container.add(new JButton("Add to cart"));
        if (media instanceof Playable) {
            container.add(new JButton("Play"));
        }

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

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

### 3.1.5 Putting it all together

```java
/*
 * Họ và tên: Phan Thế Anh
 * MSSV: 20204941
 * Mã lớp: 721428
 */
// Create constructor
public StoreScreen(Store store) {
    this.store = store;

    Container cp = getContentPane();
    cp.setLayout(new BorderLayout());

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

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

}
```

### 3.1.6 Result




## 3.2 Adding more user interaction

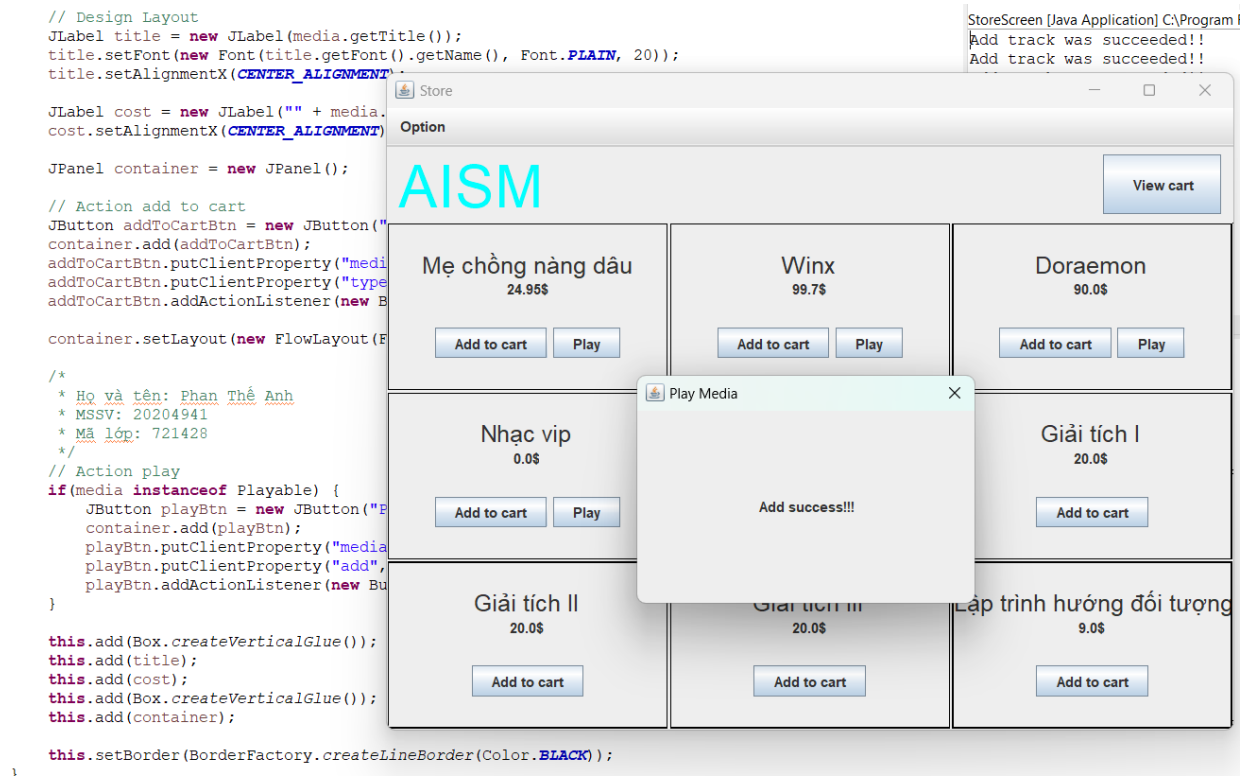
### 3.2.1 Click button “Add to cart”

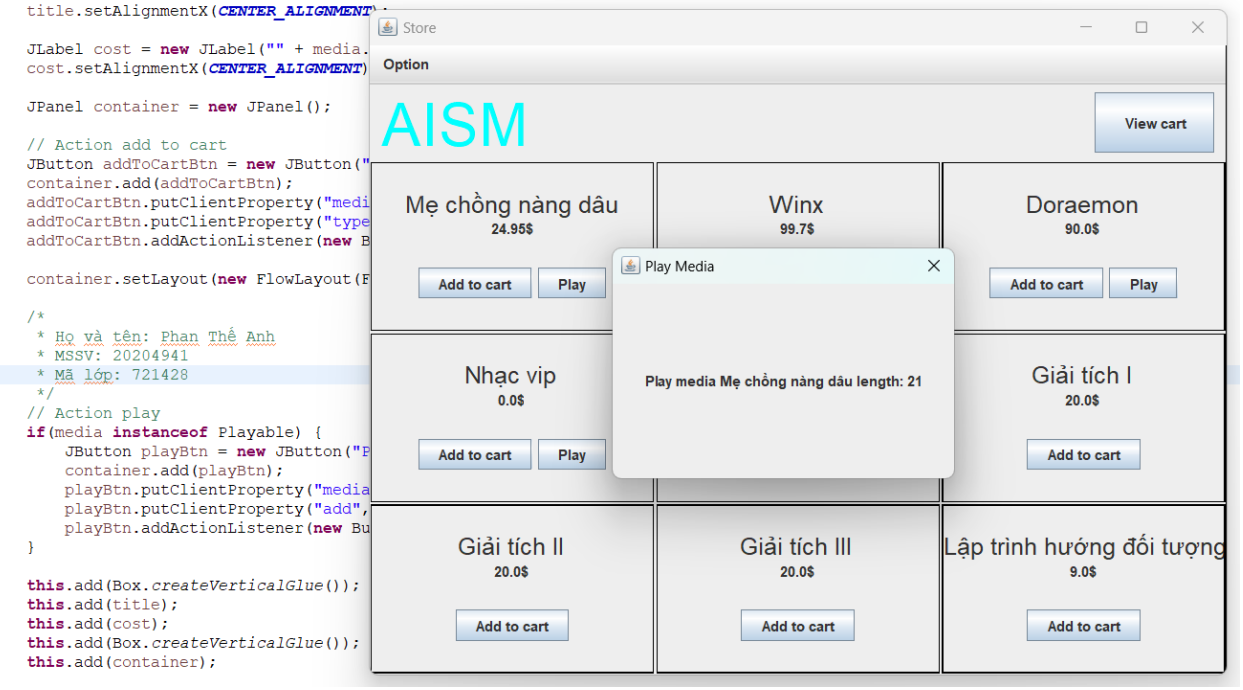
```
    /*
     * Họ và tên: Phan Thế Anh
     * MSSV: 20204941
     * Mã lớp: 721428
     */
    // Action add to cart
    JButton addToCartBtn = new JButton("Add to cart");
    container.add(addToCartBtn);
    addToCartBtn.putClientProperty("media", media);
    addToCartBtn.putClientProperty("type", "add_to_cart");
    addToCartBtn.addActionListener(new ButtonAddToCartListener());


// Event click button "Add to cart"
private class ButtonAddToCartListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        JButton button = (JButton) e.getSource();
        String type = (String) button.getClientProperty("type");
        Media selectedMedia = (Media)button.getClientProperty("media");
        // Add to cart
        try {
            StoreData.myCart.addMedia(selectedMedia);
        } catch (LimitExceededException e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }
        // Noti
        JDialog dialog = new JDialog(parentFrame, "Play Media");
        JLabel label = new JLabel("Add success!!!", SwingConstants.CENTER);
        dialog.add(label);
        dialog.setLocationRelativeTo(parentFrame);
        dialog.setSize(300, 200);
        dialog.setVisible(true);
    }
}
```

### 3.2.2 Click button “Play”

```
/*
 * Họ và tên: Phan Thế Anh
 * MSSV: 20204941
 * Mã lớp: 721428
 */
// Action play
if(media instanceof Playable) {
    JButton playBtn = new JButton("Play");
    container.add(playBtn);
    playBtn.putClientProperty("media", media);
    playBtn.putClientProperty("add", "Add media to cart");
    playBtn.addActionListener(new ButtonPlayListener());
}


/* ------------- The buttons on MediaHome ------------- */
// Event click  button "Play"
private class ButtonPlayListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        JButton button = (JButton) e.getSource();
        String type = (String) button.getClientProperty("type");
        Disc selectedMedia = (Disc)button.getClientProperty("media");
        ((Playable) selectedMedia).play();
        JDialog dialog = new JDialog(parentFrame, "Play Media");
        JLabel label = new JLabel(selectedMedia.getTitle() + " has length: " +
                String.valueOf(selectedMedia.getLength()), SwingConstants.CENTER);
        dialog.add(label);
        dialog.setLocationRelativeTo(parentFrame);
        dialog.setSize(300, 200);
        dialog.setVisible(true);
    }
}
```

### 3.2.3 Demo

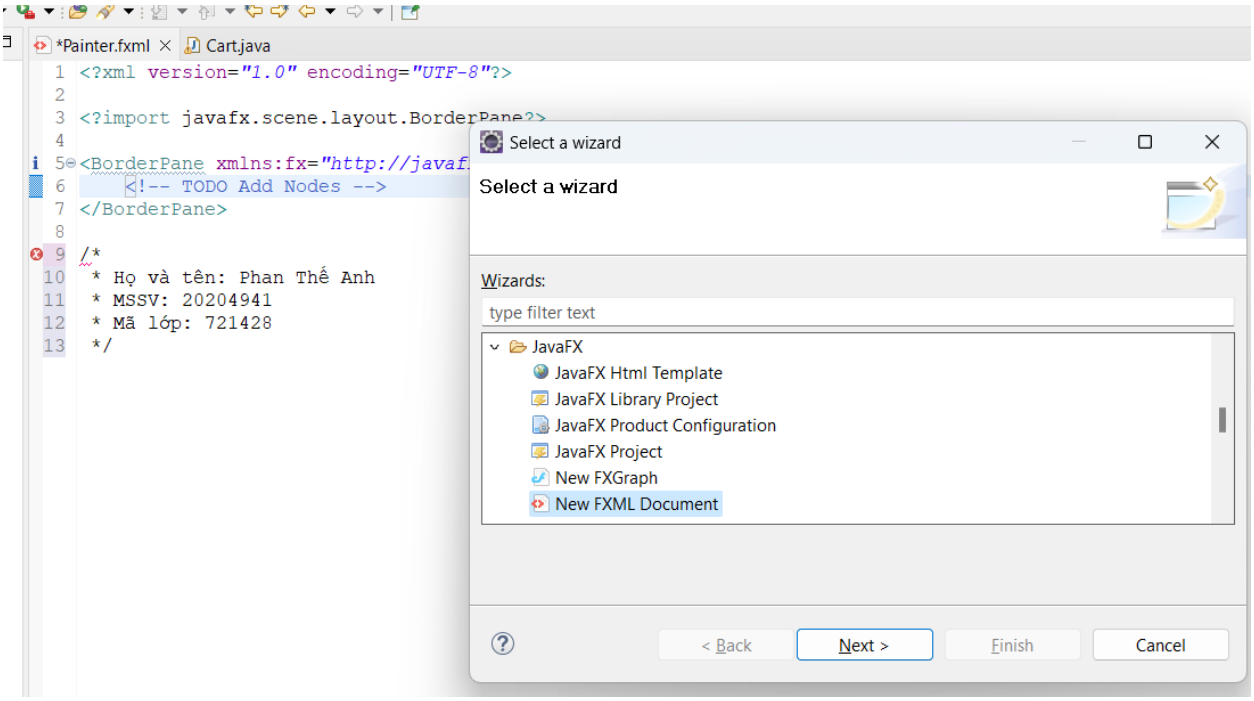


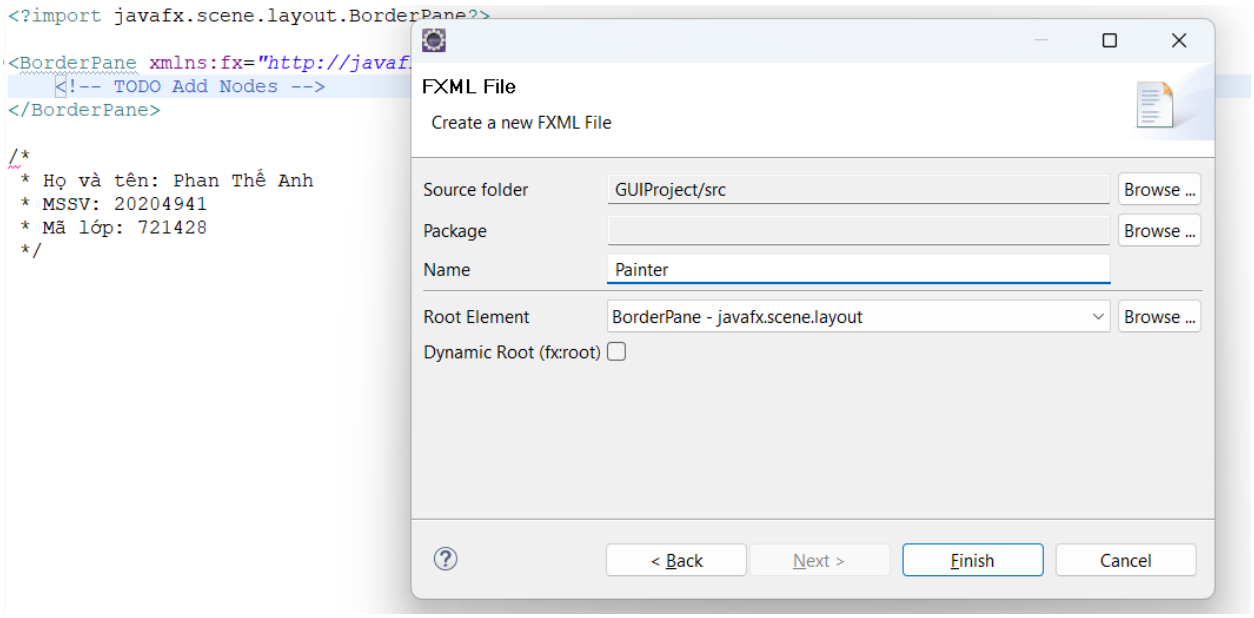



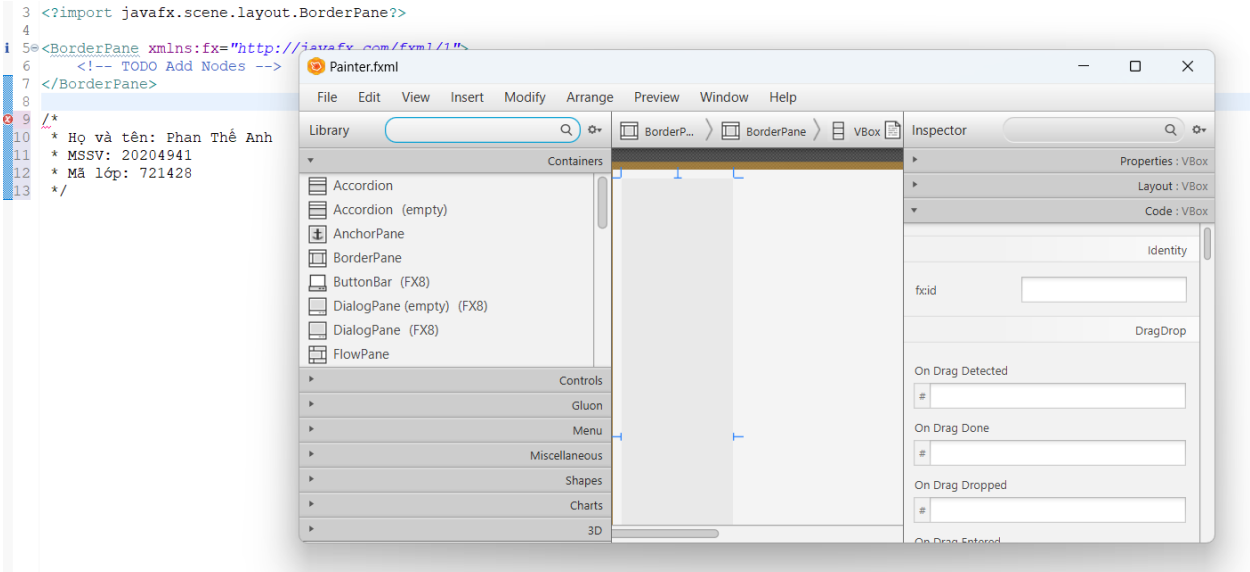

# 4. JavaFX API

## 4.1. Create the FXML file

### 4.1.1 Create and open the FXML file in Scene Builder from Eclipse

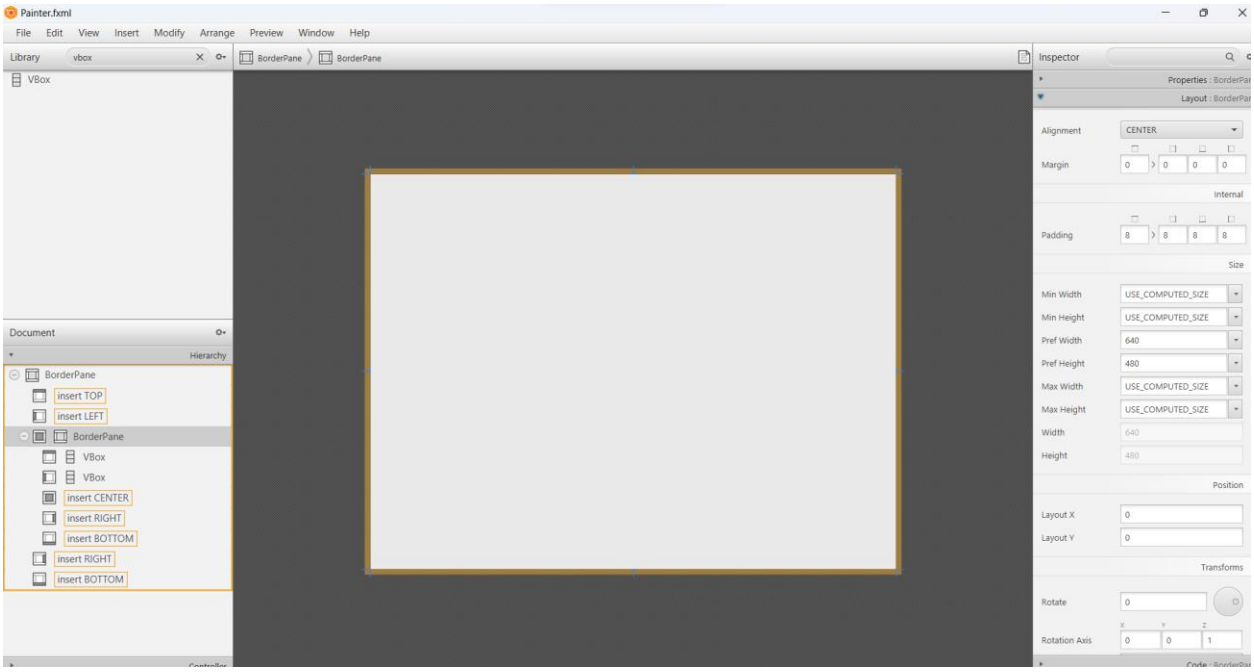

*Painter.fxml
Cart.java
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.layout.BorderPane?>
<BorderPane xmlns:fx="http://javaf
<!-- TODO Add Nodes -->
</BorderPane>
/*
* Họ và tên: Phan Thế Anh
* MSSV: 20204941
* Mã lớp: 721428
*/
Select a wizard
Select a wizard
Wizards:
type filter text
JavaFX
JavaFX Html Template
JavaFX Library Project
JavaFX Product Configuration
JavaFX Project
New FXGraph
New FXML Document
< Back
Next >
Finish
Cancel


<?import javafx.scene.layout.BorderPane?>
<BorderPane xmlns:fx="http://javaf
<!-- TODO Add Nodes -->
</BorderPane>
/*
* Họ và tên: Phan Thế Anh
* MSSV: 20204941
* Mã lớp: 721428
*/
FXML File
Create a new FXML File
Source folder
GUIProject/src
Browse ...
Package
Browse ...
Name
Painter
Root Element
BorderPane - javafx.scene.layout
Browse ...
Dynamic Root (fx:root)
< Back
Next >
Finish
Cancel

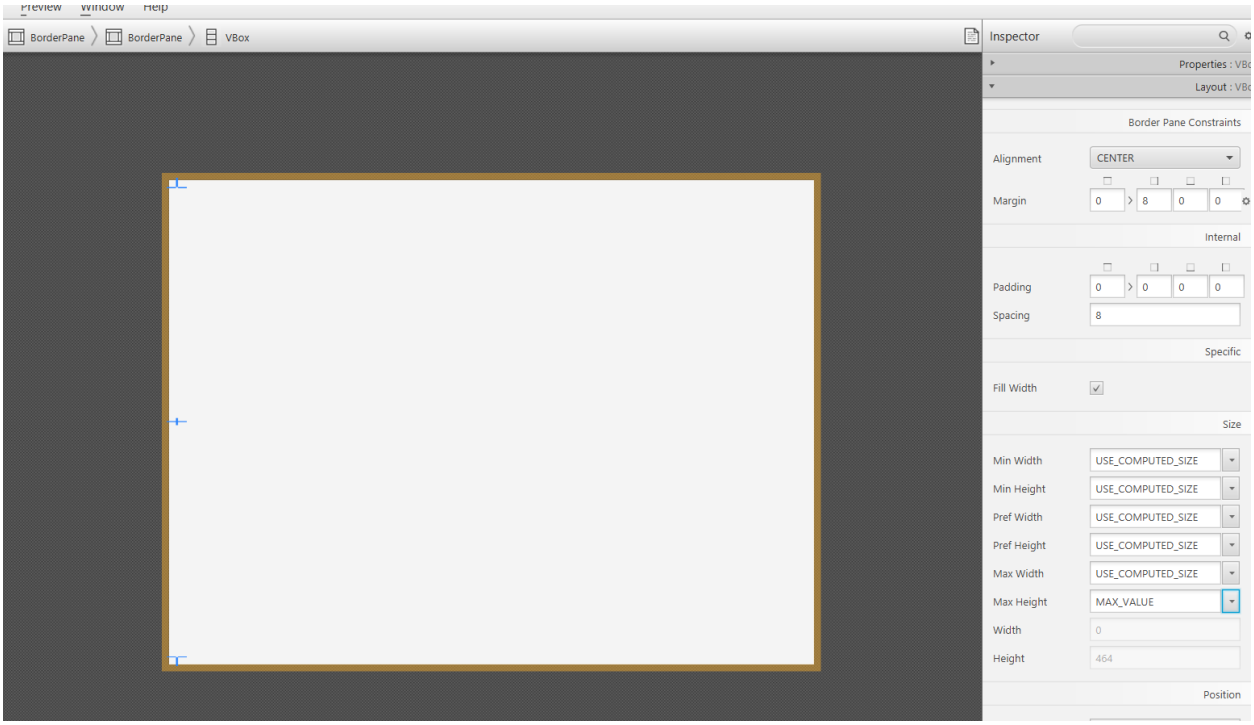


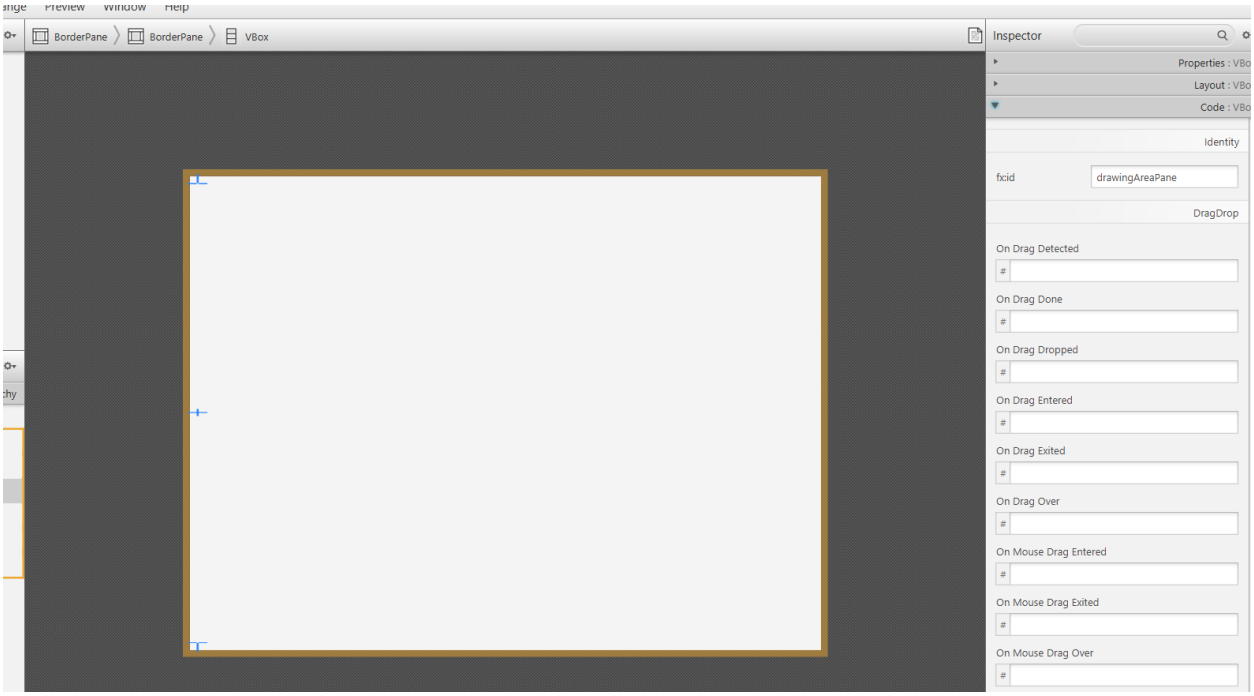

## 4.1.2 Building the GUI

**Step 1**. Configuring the BorderPane – the root element of the scene

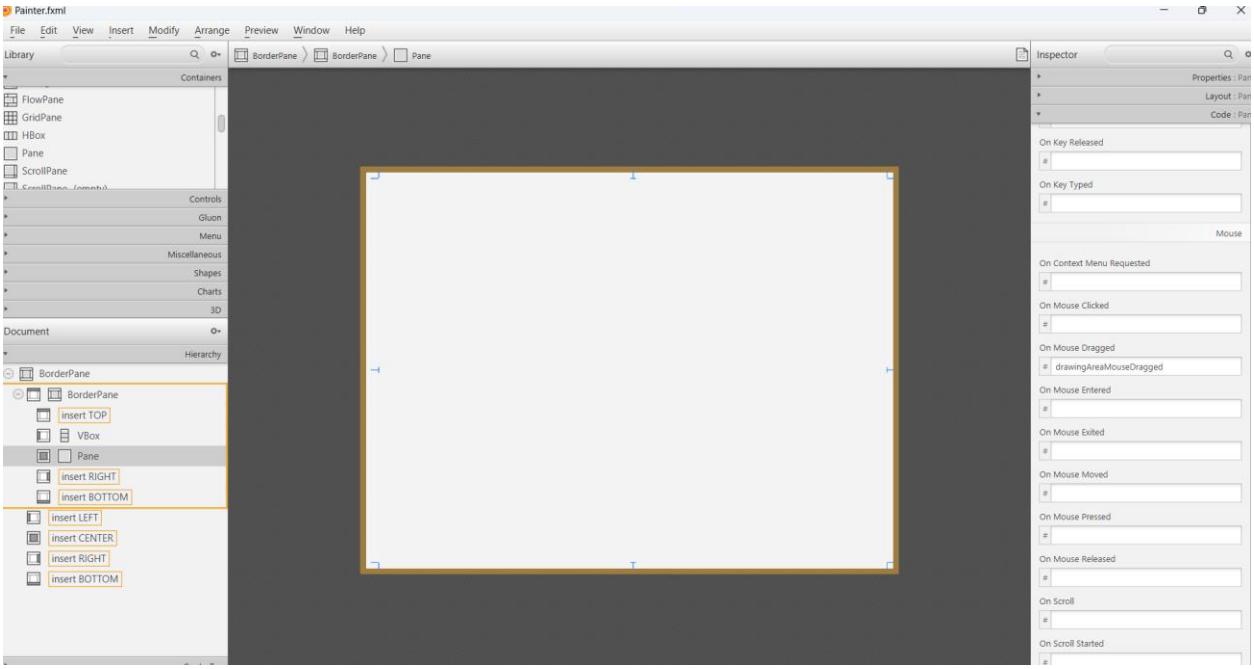



**Step 2**. Adding the Vbox

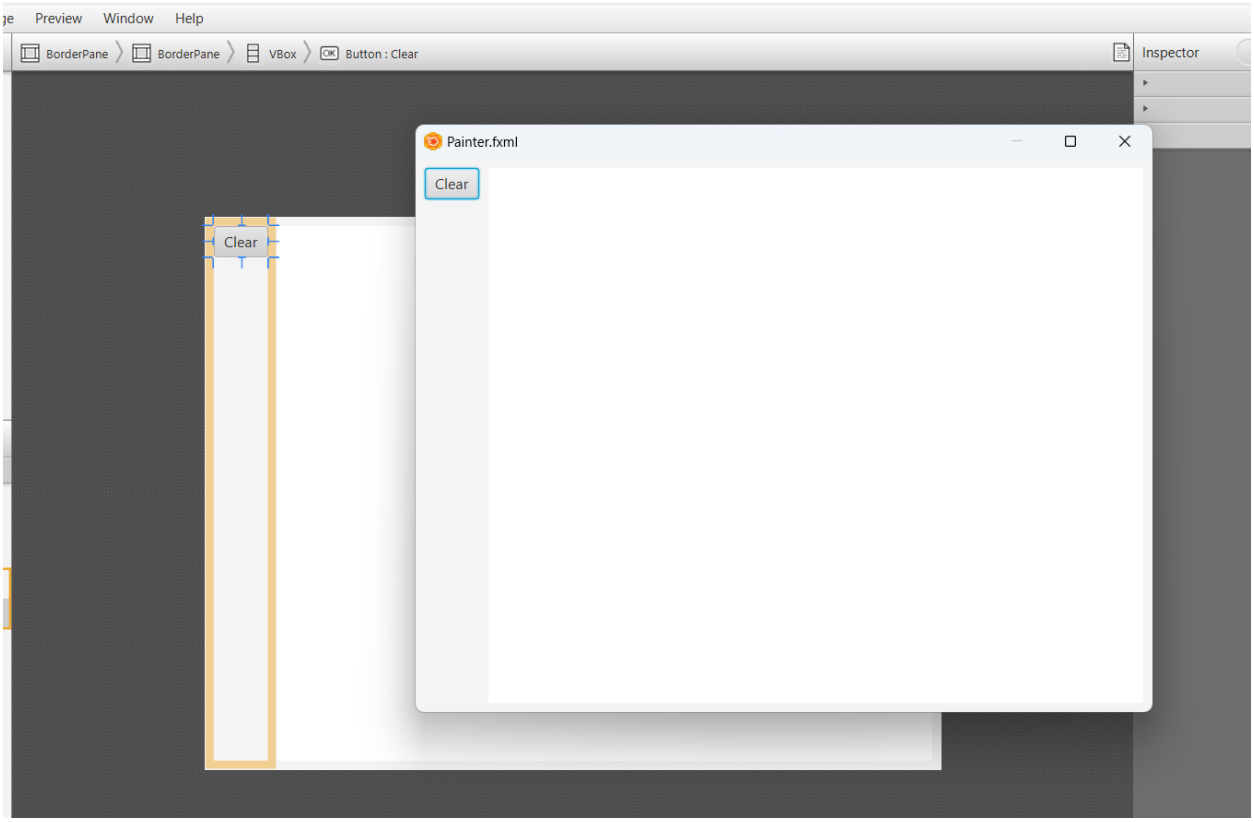


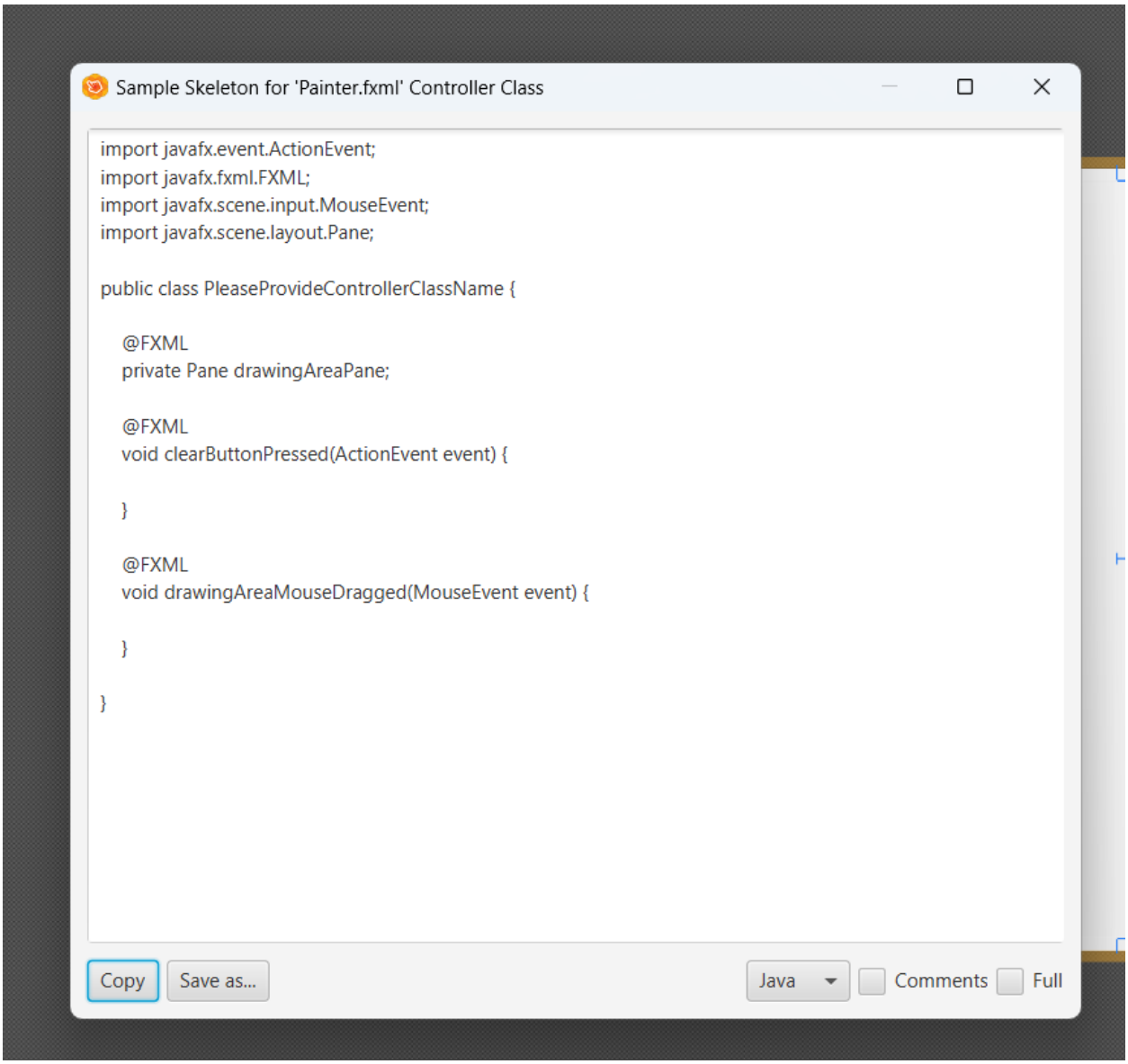

**Step 3**. Adding the Pane




**Step 4**. Adding the Button



## 4.2 Create the controller class



```java
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
 */
package application;

import javafx.event.ActionEvent;

public class PainterController {
    // Set color of pen
     private static Color typePen = Color.BLACK;
    @FXML
    private Pane drawingAreaPane;

    // Button to eraser
    @FXML
    private RadioButton radioEraserMode;

    // Button to draw
    @FXML
    private RadioButton radioPenDrawMode;

    // Color is Black
    private Color currentColor = Color.BLACK;
    @FXML

    // Action clear window
    void clearButtonPressed(ActionEvent event) {
        drawingAreaPane.getChildren().clear();
    }

    // Event click mouse to draw
    @FXML
    void drawingAreaMouseDragged(MouseEvent event) {
        Circle newCircle = new Circle(event.getX(),
                event.getY(), 4, typePen);
        drawingAreaPane.getChildren().add(newCircle);
    }

```

## 4.3 Create the application

Create a class named `Painter` in the same package as the FXML and the controller class.

```java
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
 */
package application;

import javafx.application.Application;

public class Painter extends Application {
    @Override

    public void start(Stage stage) throws Exception {
        // Create root
        Parent root = FXMLLoader.load(getClass()
                .getResource("/application/Painter.fxml")); // Resource from file Painter.fxml
        Scene scene = new Scene(root);
        // Set stage
        stage.setTitle("Painter");
        stage.setScene(scene);
        stage.show();
    }

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

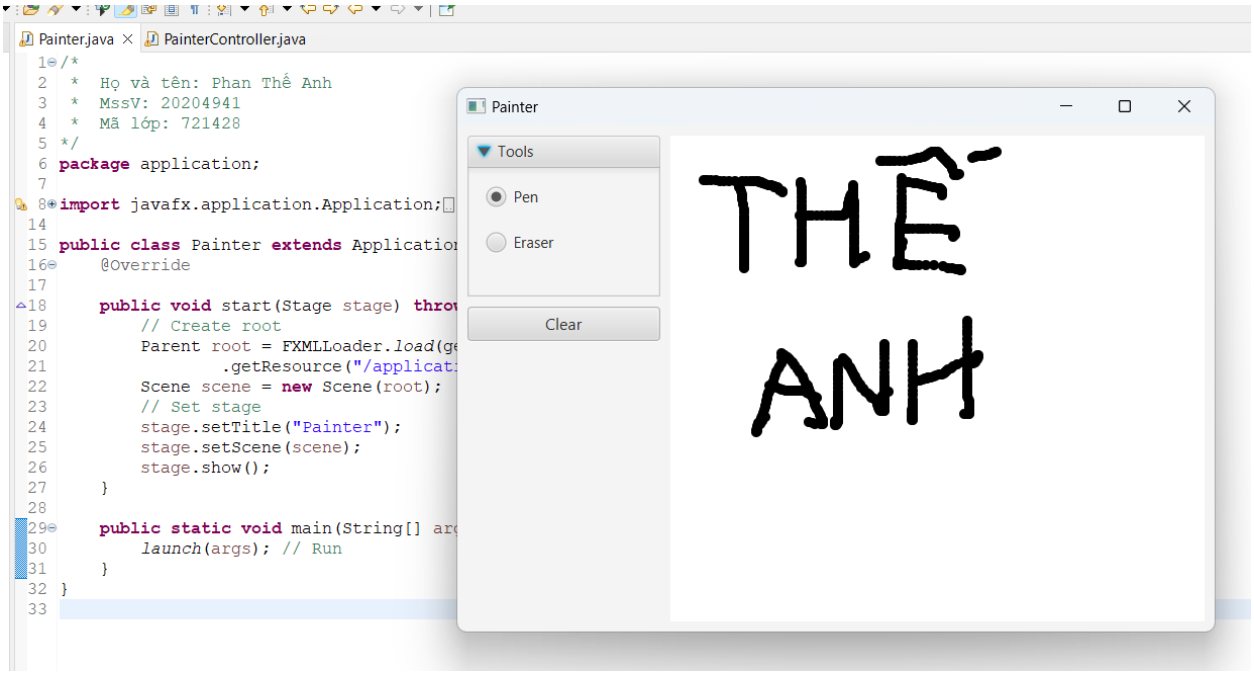
## 4.4 Practice exercise

```
29
30      // Color is Black
31      private Color currentColor = Color.BLACK;
32      @FXML
33
34      // Action clear window
35      void clearButtonPressed(ActionEvent event) {
36          drawingAreaPane.getChildren().clear();
37      }
38
39      /*
40       *  Họ và tên: Phan Thế Anh
41       *  MssV: 20204941
42       *  Mã lớp: 721428
43      */
44      // Event click mouse to draw
45      @FXML
46      void drawingAreaMouseDragged(MouseEvent event) {
47          Circle newCircle = new Circle(event.getX(),
48                  event.getY(), 4, typePen);
49          drawingAreaPane.getChildren().add(newCircle);
50      }
51      // Event eraser
52      @FXML
53      void eraserMode(ActionEvent event) {
54          setColor(Color.WHITE); // Set color is White to eraser
55      }
56
57      // Event draw
58      @FXML
59      void penMode(ActionEvent event) {
60          setColor(Color.BLACK); // Set color is BLACK to draw
61      }
62
63      // Set color to draw
64      void setColor(Color type) {
65          this.typePen = type;
66      }
67  }
68
```

- For the interface design: use `TitledPane` and `RadioButton`. Using Scene Builder, set the Toggle Group properties of the RadioButtons as identical, so only one of them can be selected at a time.
- For the implementation of Eraser: One approach is to implement an eraser just like a pen above, but use white ink color (canvas color) instead.

## 4.5 Demo

- Pen mode




- Eraser mode


# 5. Setting up the View Cart Screen with ScreenBuilder

## 5.1 Setting up the BorderPane

- Layout: Pref Width: 1024
- Layout: Pref Height: 768

## 5.2 Setting up the TOP area

Since the TOP area contains only the MenuBar on top of the header, we will use the Vbox layout.

Step 1. Drag a VBox into the BorderPane's NORTH area.

- Layout: Pref HEIGHT: USE_COMPUTED_SIZE

Step 2. Add a MenuBar into the VBox

Step 3. Add a label into the VBox

- Properties: Text: CART
- Properties: Font: 50px
- Properties: Text Fill: #00ffff (AQUA)
- Layout: Padding: 10 left

## 5.3 Setting up the CENTER area

Similar to the TOP area, we use the VBox layout for the CENTER areas to arrange components vertically. Inside the Vbox, we use a HBox to arrange the top row of components horizontally. We also use a TableView to display data in a tabular form.

Step 1. Drag a VBox into the CENTER area

- Layout: Padding: 10 left

Step 2. Add a HBox into the VBox

- Properties: Alignment: CENTER_LEFT
- Layout: Padding: 10 top & bottom
- Layout: Spacing: 10
- Layout: Pref Height: USE_COMPUTED_SIZE

Step 2.1. Add a Label into the HBox

- Properties: Text: Filter:

Step 2.2. Add a TextField into the HBox

Step 2.3. Add the first RadioButtons into the HBox

- Properties: Text: By ID
- Properties: Selected: ✓
- Properties: Toggle Group: filterCategory

Step 2.4. Add the second RadioButtons into the HBox

- Properties: Text: By Title
- Properties: Toggle Group: filterCategory

Step 3. Add a TableView into the VBox

- 3 TableColumns: Properties: Text: Title, Category & Cost.
- Column Resize Policy: constrained-resize
- Layout: Pref Width: USE_COMPUTED_SIZE
- Layout: Pref Height: USE_COMPUTED_SIZE

Step 4. Add a ButtonBar into the VBox

- 2 Buttons: Play, Remove

## 5.4 Setting up the RIGHT area

Step 1. Drag a VBox into the RIGHT area

- Properties: Alignment: TOP_CENTER
- Layout: Pref Width: USE_COMPUTED_SIZE

- Layout: Padding: 50 top

Step 2. Add an HBox into the VBox

- Properties: Alignment: CENTER
- Layout: Pref Width: USE_COMPUTED_SIZE
- Layout: Pref Height: USE_COMPUTED_SIZE

Step 3. Add a Label into the HBox

- Properties: Text: Total:
- Properties: Font: 24px
- Layout: Spacing: 10

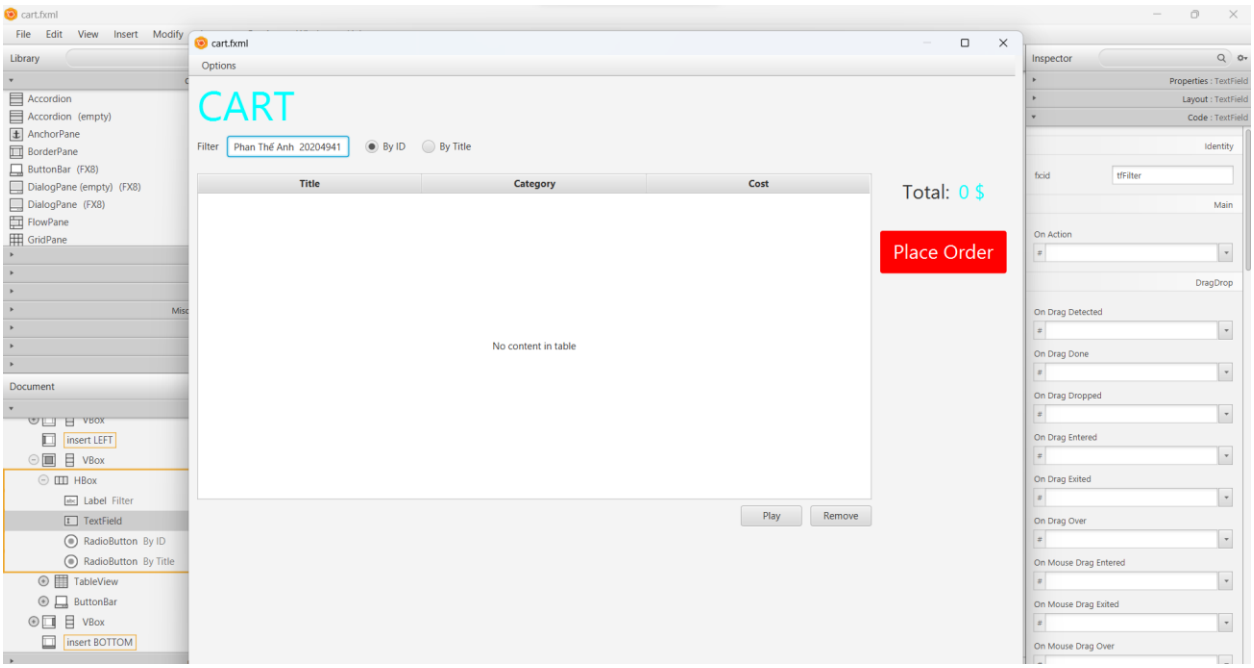
Step 4. Add another Label into the HBox

- Properties: Text: 0 $
- Properties: Font: 24px
- Properties: Text Fill: #00ffff (AQUA)

Step 5. Add a Button into the VBox

- Properties: Text: Place Order
- Properties: Font: 24px
- Properties: Text Fill: #ffffff (WHITE)
- Properties: Style: -fx-background-color: red

## 5.5 Demo




# 6. Integrating JavaFX into Swing application – The `JFXPanel` class

We create a `CartScreen` class that extends the `JFrame`, just like the `StoreScreen` class. Inside its constructor, we do the following steps:

- In line 29 – 30, we set up a `JFXPanel` in our `JFrame`.

- In line 37 – 43, we load the root `Node` from the FXML file and create its controller object (we will come to the actual implementation of the controller in the next section).
- In line 44, we create a new `Scene` with the root `Node` and add the `Scene` to the `JFXPanel`.

```
1  /*
2   *  Họ và tên: Phan Thế Anh
3   *  MssV: 20204941
4   *  Mã lớp: 721428
5   */
6  package hust.soict.dsai.aims.screen;
7
8  import hust.soict.dsai.aims.cart.Cart;
20
21 public class CartScreen extends JFrame {
22     private Cart cart;
23     // Create constructor
24     public CartScreen(Cart cart) {
25         super();
26
27         this.cart = cart;
28
29         JFXPanel fxPanel = new JFXPanel(); // Create new object JFXPanel
30         this.add(fxPanel);
31
32         this.setTitle("Cart"); // Set title
33         this.setVisible(true); // Set visible
34         Platform.runLater(new Runnable() {
35             @Override
36             public void run() {
37                 try {
38                     FXMLLoader loader = new FXMLLoader(getClass()
39                             .getResource("/screen/cart.fxml")); // Get resource from file cart.fxml
40                     CartScreenController controller =
41                             new CartScreenController(cart); // Create new object CartScreenController to control
42                     loader.setController(controller);
43                     Parent root = loader.load(); // Create a root
44                     fxPanel.setScene(new Scene(root)); // Set scene for fxPanel
45                 } catch(IOException e) { // Exception
46                     e.printStackTrace();
47                 }
48             }
49         });
50     }
51
```

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

## 7.1 Code

Set the fx:id property for the columns:

- The Title column: **colMediaTitle**
- The Category column: **colMediacategory**
- The Cost column: **colMediaCost**

Create the class `CartScreenController`

```java
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
 */
package hust.soict.dsai.aims.screen;
import hust.soict.dsai.aims.cart.Cart;
public class CartScreenController {
    private Cart cart;
    @FXML
    private TableColumn<Media, Float> colMediaCost; // Create table column

    @FXML
    private TableColumn<Media, String> colMediaTitle; // Create table column

    @FXML
    private TableColumn<Media, String> colMediaCategory; // Create table column

    @FXML
    private TableView<Media> tblMedia; // Create table
    // List filtered
    private FilteredList<Media> filteredList;

    public CartScreenController(Cart cart) {
        super();
        this.cart = cart;
        //this.filteredList = new FilteredList<Media>(cart.getItemsOrdered());
    }

    // Event
    @FXML
    private void initialize() {
        // Fill dữ liệu tương ứng với các cột trong bảng
        colMediaCategory.setCellValueFactory(
                new PropertyValueFactory<Media, String>("category"));
        colMediaTitle.setCellValueFactory(
                new PropertyValueFactory<Media, String>("title"));
        colMediaCost.setCellValueFactory(
                new PropertyValueFactory<Media, Float>("cost"));

        tblMedia.setItems(filteredList);
```

Recall earlier in the `CartScreen` constructor method, the constructor for `CartScreenController` is invoked, then the `load()` method of the FXMLLoader. In the `load()` method, any `@FXML` annotated fields are populated, then `initialize()` is called. That means we use can use the `initialize()` method to perform any post-processing on the content of the `JFrame` after it is loaded from FXML and before it is shown.

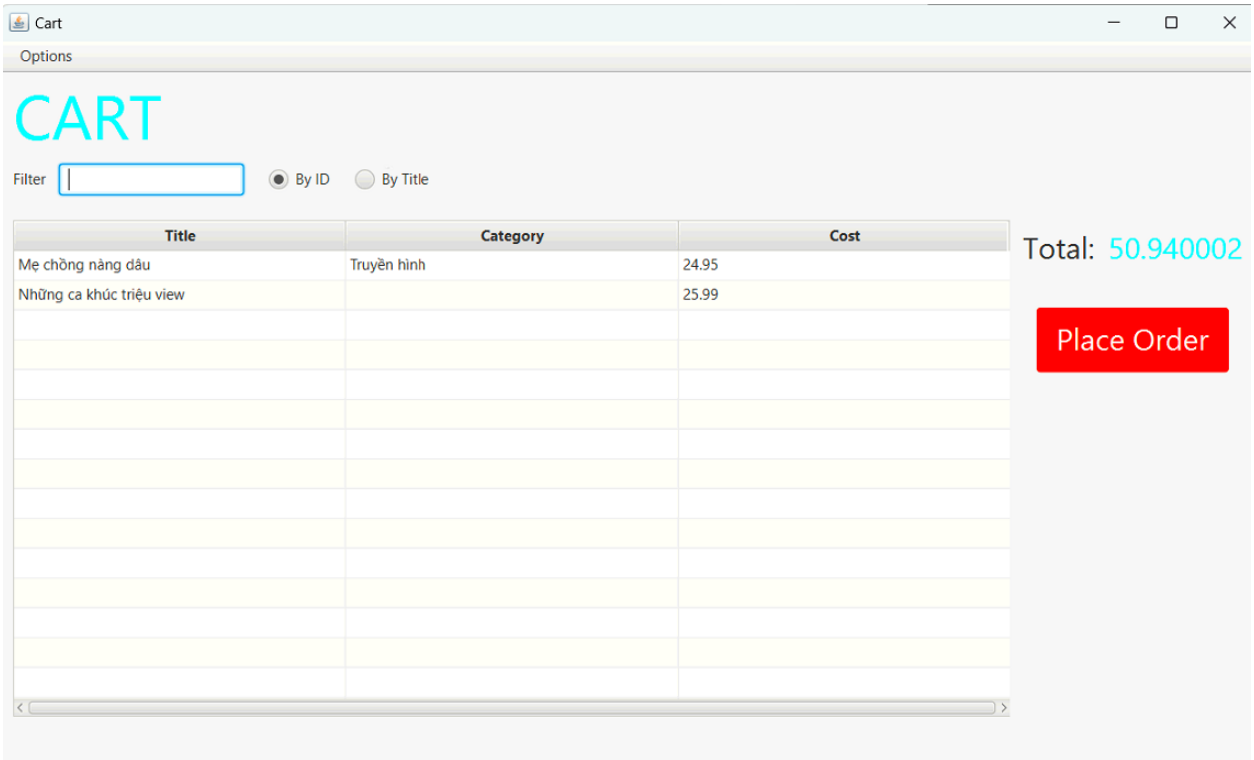
In line 54, we set the cart's list of items to the items of the `TableView`. Note that this will initially cause an error, because we cannot set a regular `List` as the items of a `TableView`. Instead, we have to use an `ObservableList`, so that any change on the data can be observed and reflected by the `TableView`. Please open the source code of `Cart` and change the `itemsOrdered` from `List<Media>` to `ObservableList<Media>`

```java
private ObservableList<Media> itemsOrdered =
        FXCollections.observableArrayList();
```

After setting the items of the `TableView`, the data still isn't showing up in the `TableView` yet, because we still have to set up the way the columns can retrieve data. This is done by setting the columns' `cellValueFactory`.

In line 47 – 52, we set the columns' `cellValueFactory` using the class `PropertyValueFactory<S, T>` (Read the Javadocs for more details). This class is a callback that will take in a `String <property>` and look for the method `get<property>()` in the `Source S` class. If a method matching this pattern exists, the value returned from this method is returned to the `TableCell`.

### 7.2 Demo

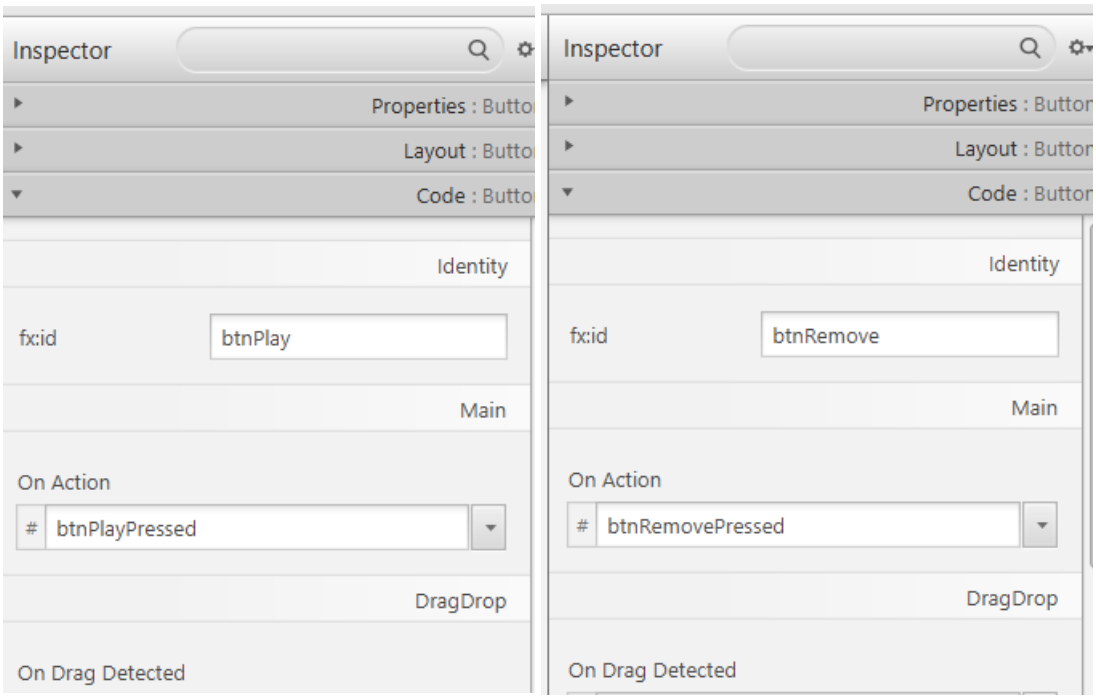



## 8. Updating buttons based on selected item in **`TableView`** - **`ChangeListener`**

### 8.1 Code

We start by adding the fx:id property for the components in SceneBuilder:

- The "Play" Button: **btnPlay**
- The "Remove" Button: **btnRemove**




We have to modify the `initialize()` method to make them invisible at first

```
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
*/
// Event
@FXML
private void initialize() {
    // Fill data to col
    colMediaCategory.setCellValueFactory(
            new PropertyValueFactory<Media, String>("category"));
    colMediaTitle.setCellValueFactory(
            new PropertyValueFactory<Media, String>("title"));
    colMediaCost.setCellValueFactory(
            new PropertyValueFactory<Media, Float>("cost"));

    tblMedia.setItems(this.cart.getItemsOrdered());

   // Set visible button
   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) {
                        // Update cart
                        updateButtonBar(newValue);
                    } else {
                        // Set visible button
                        btnPlay.setVisible(false);
                        btnRemove.setVisible(false);
                    }
                }
        }

    );
```

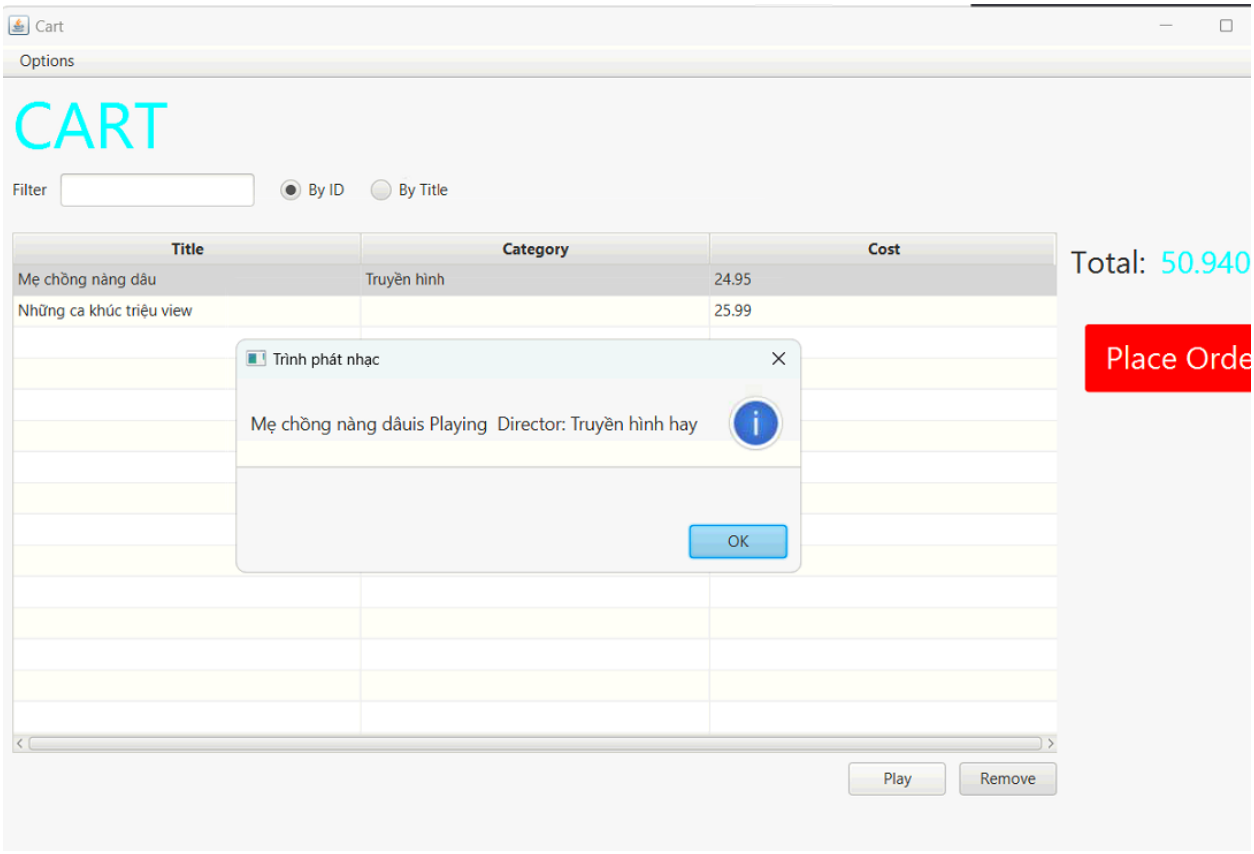
,

We check to make sure the `newValue` is not `null` (the user didn't just unselect) and call the `updateButtonBar()` method.

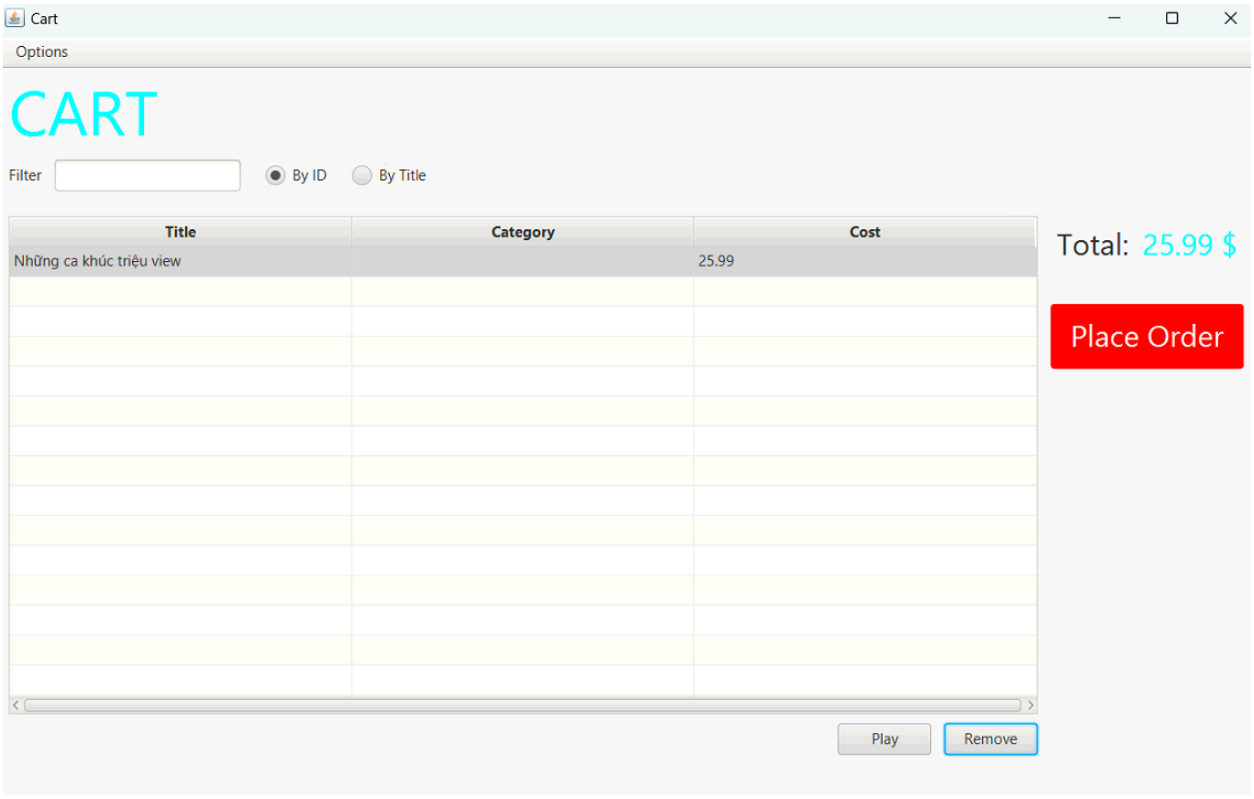
```
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
*/
void updateButtonBar(Media media) {
    btnRemove.setVisible(true); // Set visible button
    if(media instanceof Playable) {
        btnPlay.setVisible(true);
    } else {
        btnPlay.setVisible(false);
    }
}
```

## 8.2 Demo

Play button




Remove button

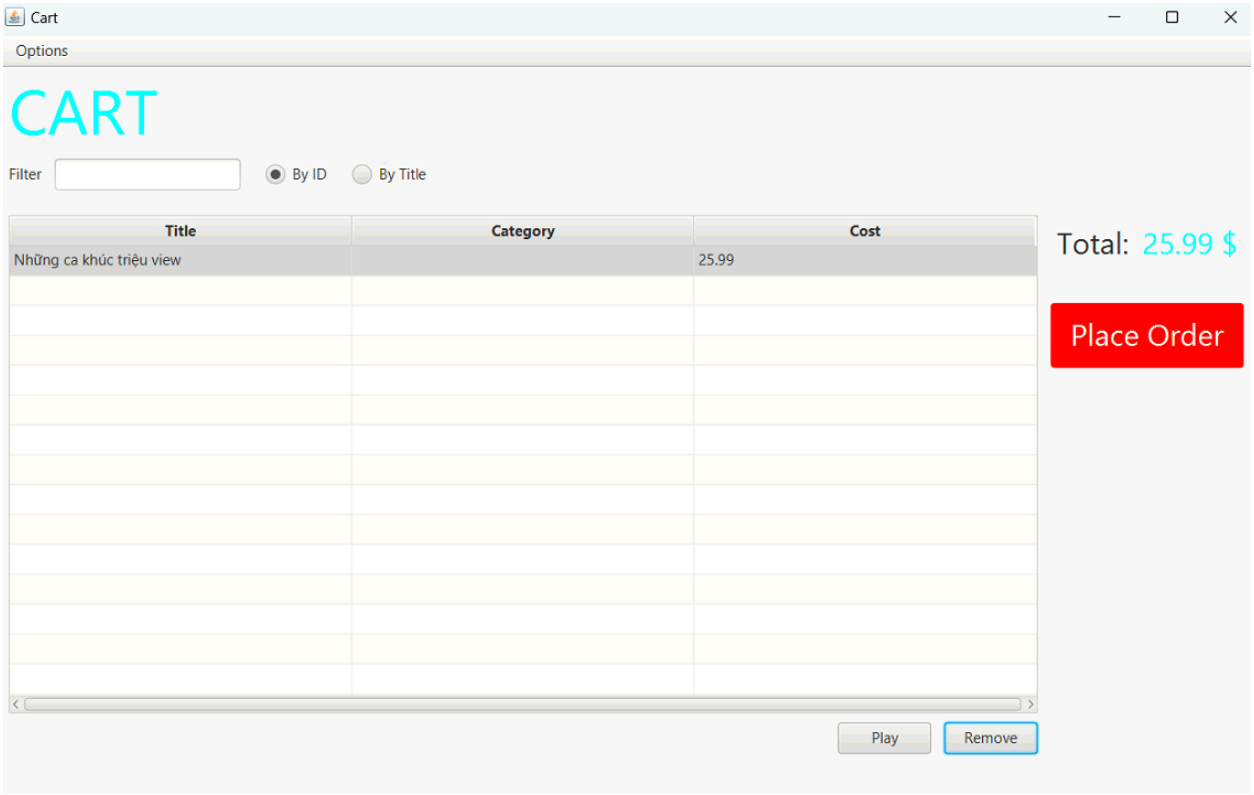


## 9. Deleting a media

We will implement the event-handling for the “Remove” button.

```
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
*/
// Create method for remove button
@FXML
void btnRemovePressed(ActionEvent event) {
    Media media = tblMedia.getSelectionModel().getSelectedItem();
    cart.removeMedia(media); // Remove
}
```

Result



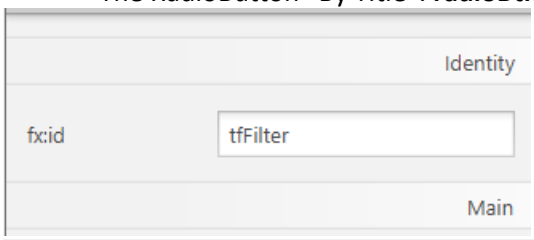

## 10. Filter items in cart – `FilteredList`

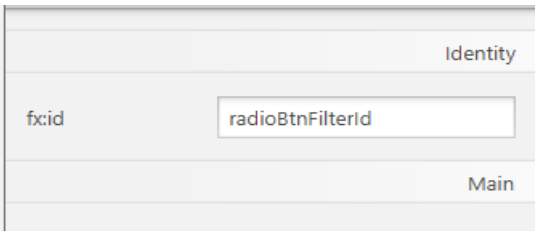
### 10.1 Code

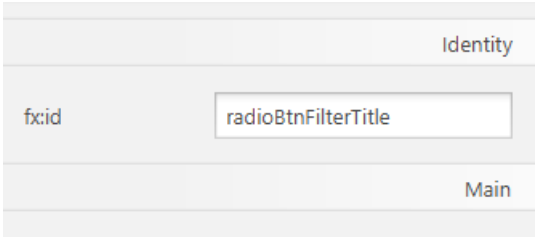
Add the fx:id property for the components in SceneBuilder and create three corresponding attributes in the controller:

- The TextField: **tfFilter**

- The RadioButton “By ID”: **radioBtnFilterId**
- The RadioButton “By Title”: **radioBtnFilterTitle**



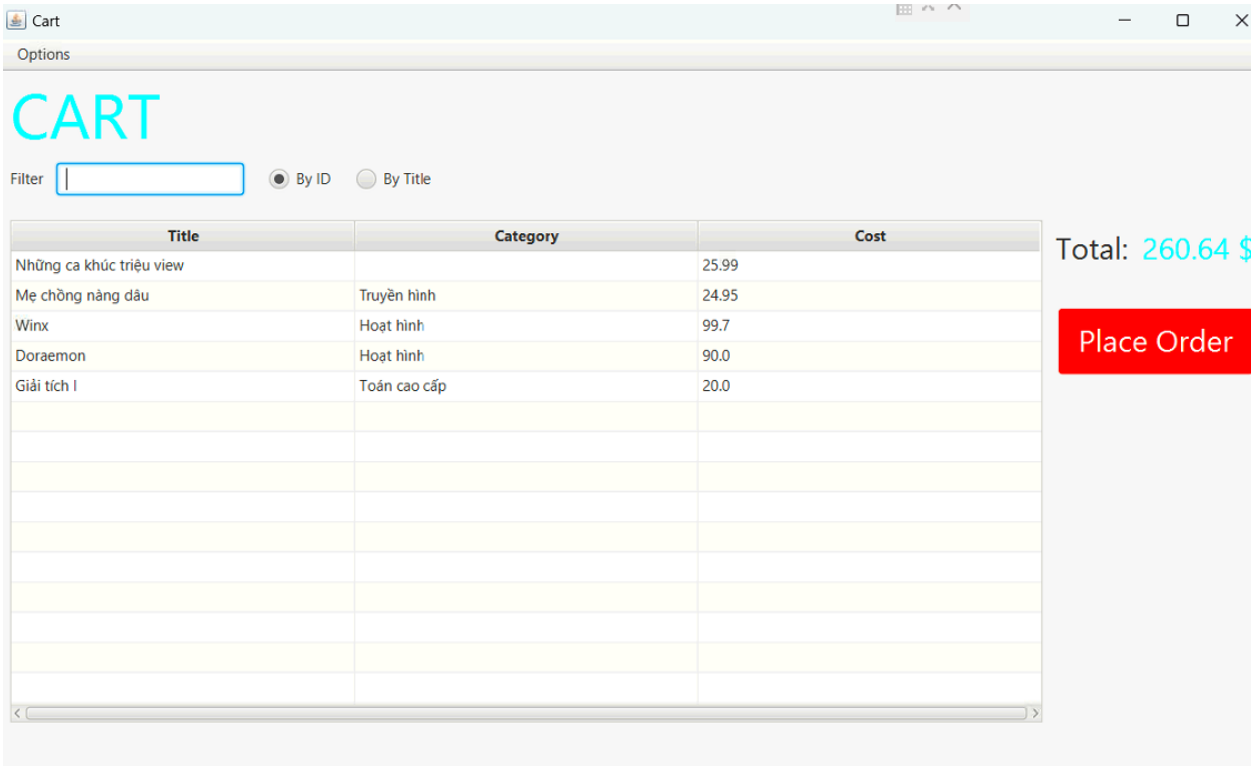

At the end of the `initialize()` method, put some code to add a `ChangeListener` to the `TextField`’s text property.

```
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
*/
 // Event enter input
 tfFilter.textProperty().addListener(
         new ChangeListener<String>() {
             @Override
             public void changed(ObservableValue<? extends String> observable, String oldValue,
                     String newValue) {
                 showFilterMedia(newValue);
             }
         }
 );
```

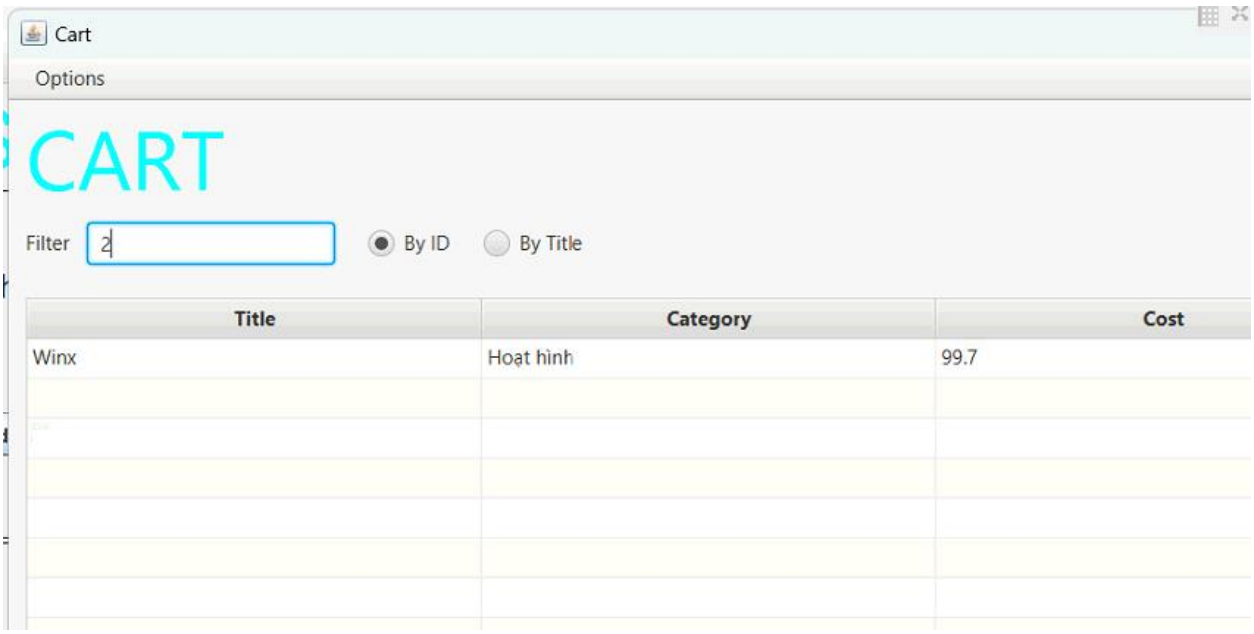
Create the `showFilteredMedia()` method.

```java
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
*/
// Filter data
void showFilterMedia(String filter) {
    filteredList.setPredicate(item -> {
        if(filter.isEmpty()) {
            return true;
        }
        try {
            // Filter by ID
            if(radioBtnFilterId.isSelected()) {
                return String.valueOf(item.getId()).equals(filter);
            } else if(radioBtnFilterTitle.isSelected()) {
                // Filter by title
                return item.getTitle().contains(filter);
            }
        } catch(Exception e) {
            e.printStackTrace();
        }

        return true;

    });

}


 private Label totalCostLabel;

 // List filtered
 private FilteredList<Media> filteredList;

 public CartScreenController(Cart cart) {
     super();
     this.cart = cart;
     this.filteredList = new FilteredList<Media>(cart.getItemsOrdered());
 }
```

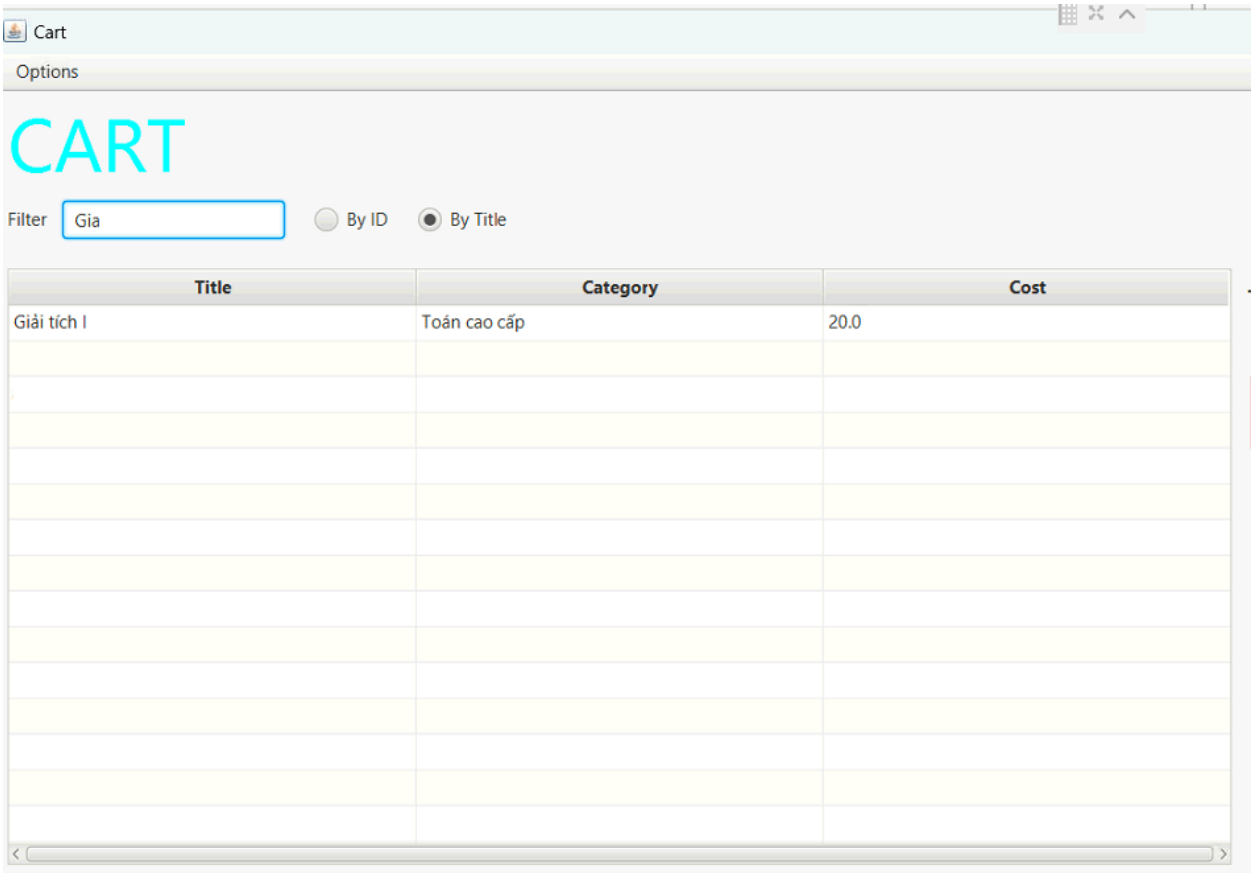
## 10.2 Demo




Filter by ID




Filter by Title




## 11. Complete the Aims GUI application

Complete the remaining UI of Aims to make a functioning GUI application

### 11.1 Cart Screen:

- "Place order" Button

```
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
*/
// Action when click button Place order
@FXML
void btnPlaceOrderPressed(ActionEvent event) {
    cart.placeOrder();
    Alert alert = new Alert(Alert.AlertType.INFORMATION);
    alert.setTitle("Notification");
    alert.setHeaderText("Order Success!!!");
    alert.show();
}
```

- "Play" Button

```
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
*/
// Action when click button Play
@FXML
void btnPlayPressed(ActionEvent event) {
    // Media selected
    Playable media = (Playable)tblMedia.getSelectionModel().getSelectedItem();
    // Get title
    String mediaTitle = tblMedia.getSelectionModel().getSelectedItem().getTitle();
    try {
        // Play
        media.play();
        // Notification
        Alert alert = new Alert(Alert.AlertType.INFORMATION);
        alert.setTitle("Play");
        Disc disc = (Disc)media;
        alert.setHeaderText(mediaTitle + " from " + disc.getDirector());
        alert.show();
    } catch (PlayerException e) {
        // Exception
        Alert alert = new Alert(Alert.AlertType.ERROR);
        alert.setTitle("Error");
        alert.setHeaderText(e.getLocalizedMessage());
        alert.show();
    }

}
```

- The total cost Label - should updated along with changes in the current cart (add/remove).

```
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
*/
// Calculate total cost
cart.getItemsOrdered().addListener((ListChangeListener<Media>) change -> {
    while (change.next()) {
        Platform.runLater(new Runnable() {
            @Override
            public void run() {
                totalCostLabel.setText(String.valueOf(cart.totalCost()) + " $"); // Update total cost
                System.out.println("Update total cost");
            }
        });

    }
});
```

- MenuBar

```java
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
*/
// Menubar
public static class MenuItemListener implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        String command = e.getActionCommand();
        // View store
        if(command.equals("view_store")) {
            currentView = new StoreScreen(myStore);
        }
        // View cart
        else if(command.equals("view_cart")) {
            currentView = new CartScreen(myCart);
        }
        else if(command.equals("add_dvd")) { // Add DVD
            new AddDigitalVideoDiscToStoreScreen();
        } else if(command.equals("add_cd")) { // Add CD
            new AddCompactDiscToStoreScreen();
        }  else if(command.equals("add_book")) { // Add Book
            new AddBookToStoreScreen();
        }
        System.out.println(e.getActionCommand() + " JMenuItem clicked.");
    }
}
```

## 11.2 Store Screen:

- "Add to cart" Button

```java
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
*/
// Event Add to Cart
private class AddToCartButtonListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        JButton button = (JButton) e.getSource();
        String type = (String) button.getClientProperty("type");
        Media selectedMedia = (Media)button.getClientProperty("media");
        try {
            // Add to cart
            InitData.myCart.addMedia(selectedMedia);
        } catch (LimitExceededException ex) {
            // Exception
            JDialog dialog = new JDialog(parentFrame, "Notification");
            JLabel label = new JLabel(ex.getLocalizedMessage(), SwingConstants.CENTER);
            dialog.add(label);
            dialog.setLocationRelativeTo(parentFrame);
            dialog.setSize(300, 200);
            dialog.setVisible(true);
        }
    }
}
```

## 11.3 Update Store Screen

When user clicks on one of the items of the “Update Store” menu on the menu bar (such as “Add Book”, “Add CD”, or “Add DVD”), the application should switch to an appropriate new screen for the user to add the new item. This screen should have the same menu bar as the View Store Screen, so the user can go back to view the store.

### 11.3.1 Create AddItemToStoreScreen class

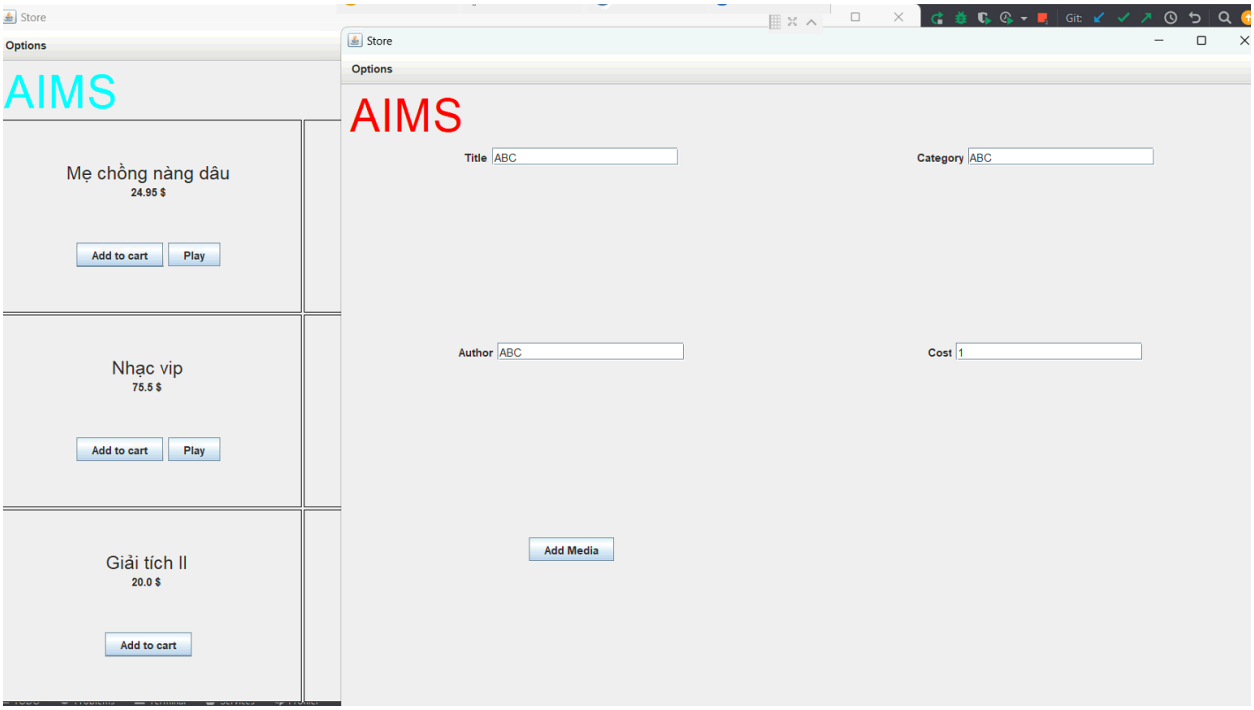
```
1  /*
2   *  Họ và tên: Phan Thế Anh
3   *  MssV: 20204941
4   *  Mã lớp: 721428
5   */
6  package hust.soict.dsai.aims.screen;
7  import hust.soict.dsai.aims.Aims;
11
12 public class AddItemToStoreScreen extends JFrame {
13     protected JPanel centerLayout;
14     public AddItemToStoreScreen() {
15         super();
16     }
17
18     JPanel createNorth() {
19         JPanel north = new JPanel();
20         north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
21         north.add(createMenuBar());
22         north.add(createHeader());
23         return north;
24     }
25
26     // Header
27     JPanel createHeader() {
28         JPanel header = new JPanel();
29         header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));
30
31         JLabel title = new JLabel("AIMS");
32         title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
33         title.setForeground(Color.RED);
34         header.add(Box.createRigidArea(new Dimension(10, 10)));
35         header.add(title);
36         header.add(Box.createHorizontalGlue());
37         header.add(Box.createRigidArea(new Dimension(10, 10)));
38         return header;
39     }
40
```

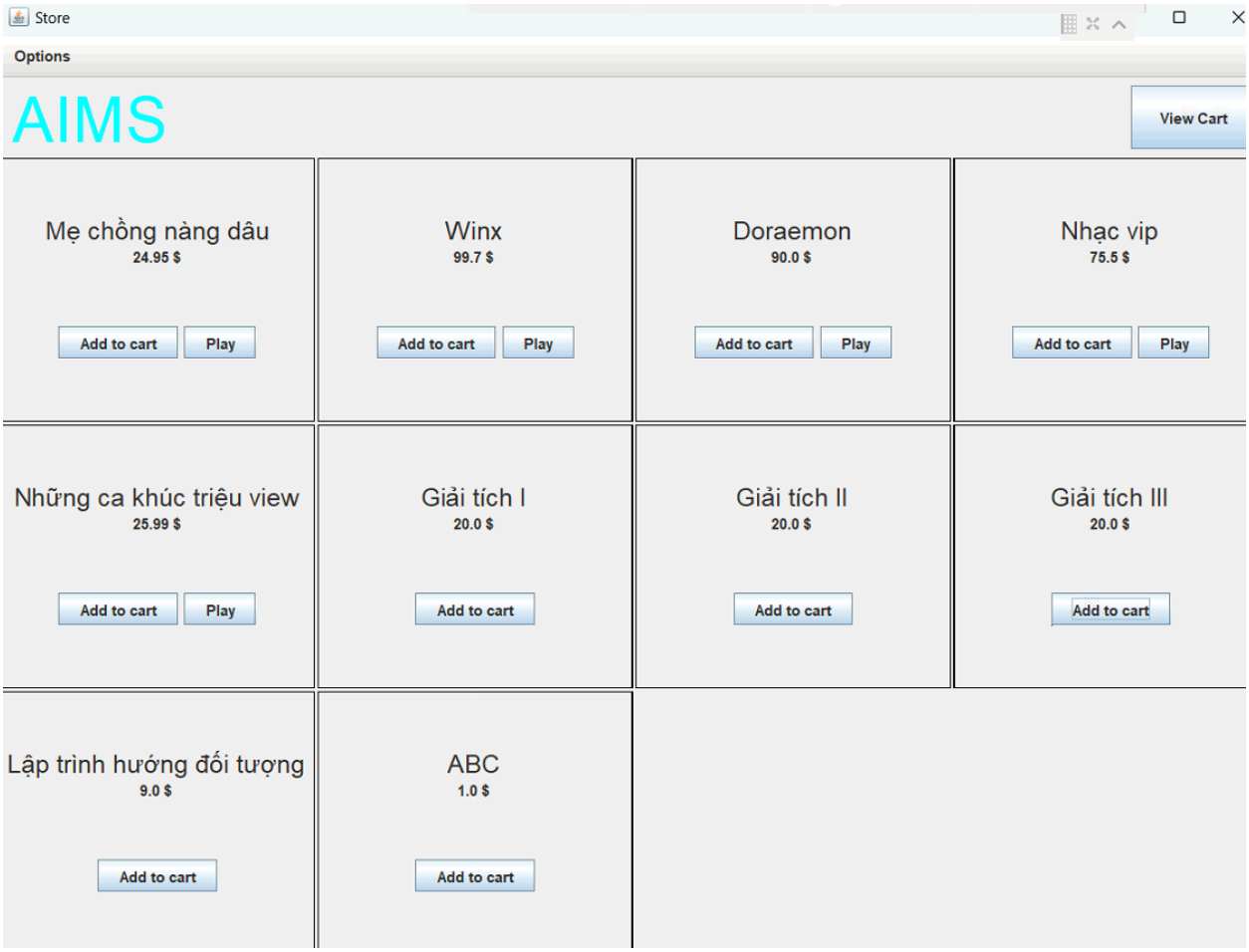
```
40
41      // Menu
42      JMenuBar createMenuBar() {
43          JMenu menu = new JMenu("Options");
44
45          JMenu smUpdateStore = new JMenu("Update Store");
46          smUpdateStore.add(new JMenuItem("Add Book"));
47          smUpdateStore.add(new JMenuItem("Add CD"));
48          smUpdateStore.add(new JMenuItem("Add DVD"));
49
50          menu.add(smUpdateStore);
51          Aims.MenuItemListener menuItemListener = new Aims.MenuItemListener();
52
53
54          JMenuItem viewStoreMenu = new JMenuItem("View Store");
55          viewStoreMenu.setActionCommand("view_store");
56          viewStoreMenu.addActionListener(menuItemListener);
57          menu.add(viewStoreMenu);
58
59          JMenuItem viewCartMenu = new JMenuItem("View Cart");
60          viewCartMenu.setActionCommand("view_cart");
61          viewCartMenu.addActionListener(menuItemListener);
62          menu.add(viewCartMenu);
63
64
65
66          JMenuBar menuBar = new JMenuBar();
67
68          menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
69          menuBar.add(menu);
70          return menuBar;
71      }
72
73
```

## 11.3.2 Create AddBookToStoreScreen class

```java
/*
 * Họ và tên: Phan Thế Anh
 * MssV: 20204941
 * Mã lớp: 721428
*/
package hust.soict.dsai.aims.screen;

import hust.soict.dsai.aims.data.InitData;

public class AddBookToStoreScreen extends AddItemToStoreScreen {
    private Store myStore = InitData.myStore;
    // Create constructor
    public AddBookToStoreScreen() {
        Container cp = getContentPane();
        cp.setLayout(new BorderLayout());
        cp.add(createNorth(), BorderLayout.NORTH);
        cp.add(renderAddBookScreen(), BorderLayout.CENTER);

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

    }

    JPanel renderAddBookScreen() {
        JPanel layout = new JPanel();
        layout.setLayout(new GridLayout(3, 3, 2, 2));
        // Input Title
        JPanel inputTitleGroup = this.inputGroup("Title", 20);
        layout.add(inputTitleGroup);
        // Input category
        JPanel inputCategoryGroup = this.inputGroup("Category", 20);
        layout.add(inputCategoryGroup);

        // Input author
        JPanel inputAuthorGroup = this.inputGroup("Author", 20);
        layout.add(inputAuthorGroup);

        // Input Cost
        JPanel inputCostGroup = this.inputGroup("Cost", 20);
        layout.add(inputCostGroup);

        // Submit button
        layout.add(submitButton(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // Create new book
                Book book = new Book(getValueTextField(inputTitleGroup),
                        getValueTextField(inputCategoryGroup), Float.valueOf(getValueTextField(inputCostGroup)));
                // Add author
                book.addAuthor(getValueTextField(inputAuthorGroup));
                // Add to store
                myStore.addMedia(book);
            }
        }));
        return layout;
    }
}
```

Result


Store
Options
AIMS
Mẹ chồng nàng dâu
24.95 $
Add to cart
Play
Nhạc vip
75.5 $
Add to cart
Play
Giải tích II
20.0 $
Add to cart
Store
Options
AIMS
Title ABC
Category ABC
Author ABC
Cost 1
Add Media

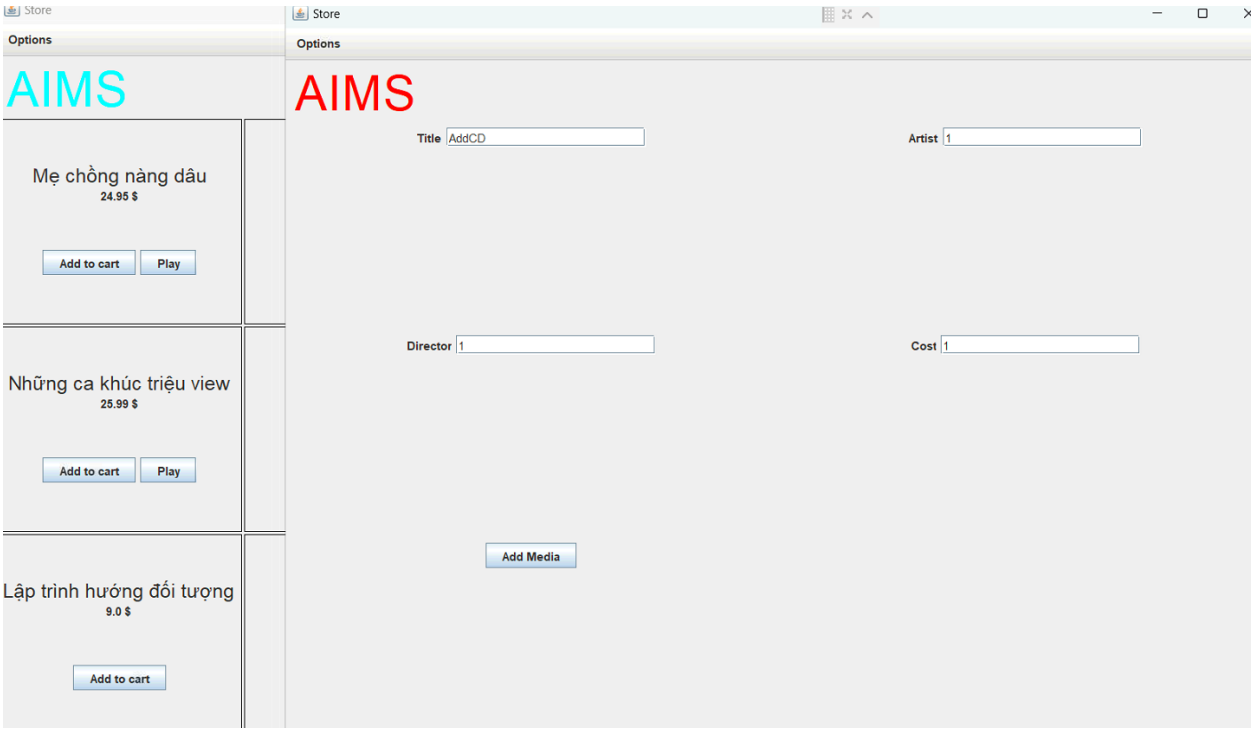

Store
Options
AIMS
View Cart
Mẹ chồng nàng dâu
24.95 $
Add to cart
Play
Winx
99.7 $
Add to cart
Play
Doraemon
90.0 $
Add to cart
Play
Nhạc vip
75.5 $
Add to cart
Play
Những ca khúc triệu view
25.99 $
Add to cart
Play
Giải tích I
20.0 $
Add to cart
Giải tích II
20.0 $
Add to cart
Giải tích III
20.0 $
Add to cart
Lập trình hướng đối tượng
9.0 $
Add to cart
ABC
1.0 $
Add to cart

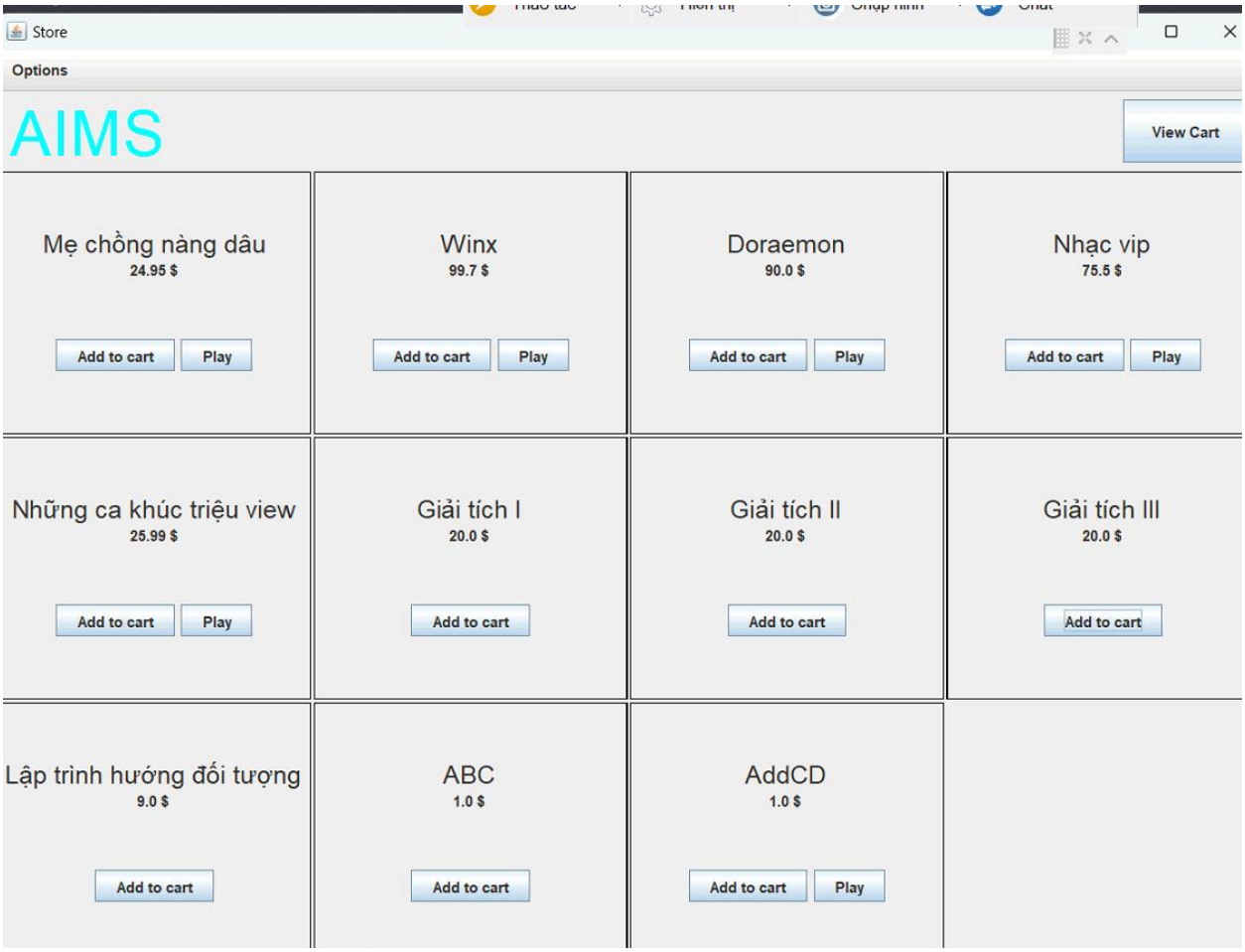
## 11.3.3 Create AddCDToStoreScreen class

```java
/*
 * Họ và tên: Phan Thế Anh
 * MssV: 20204941
 * Mã lớp: 721428
 */
package hust.soict.dsai.aims.screen;
import hust.soict.dsai.aims.data.InitData;

public class AddCDToStoreScreen extends AddItemToStoreScreen {
    private Store myStore = InitData.myStore;
    // Constructor
    public AddCDToStoreScreen() {
        Container cp = getContentPane();
        cp.setLayout(new BorderLayout());
        cp.add(createNorth(), BorderLayout.NORTH);
        cp.add(renderAddCDScreen(), BorderLayout.CENTER);

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

    }
```

```java
    // Add CD
    JPanel renderAddCDScreen() {
        JPanel layout = new JPanel();
        layout.setLayout(new GridLayout(3, 3, 2, 2));
        // Input Title
        JPanel inputTitleGroup = this.inputGroup("Title", 20);
        layout.add(inputTitleGroup);
        // Input category
        JPanel inputArtistGroup = this.inputGroup("Artist", 20);
        layout.add(inputArtistGroup);
        // Input director
        JPanel inputDirectorGroup = this.inputGroup("Director", 20);
        layout.add(inputDirectorGroup);
        // Input Cost
        JPanel inputCostGroup = this.inputGroup("Cost", 20);
        layout.add(inputCostGroup);

        // Submit button
        layout.add(submitButton(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // Create new CD
                CompactDisc cd = new CompactDisc(getValueTextField(inputArtistGroup),
                        getValueTextField(inputTitleGroup), getValueTextField(inputDirectorGroup),
                        Float.valueOf(getValueTextField(inputCostGroup)));
                // Add CD to store
                myStore.addMedia(cd);
            }
        }));
        return layout;
    }
```

Result


Store
Options
AIMS
Mẹ chồng nàng dâu
24.95 $
Add to cart
Play
Những ca khúc triệu view
25.99 $
Add to cart
Play
Lập trình hướng đối tượng
9.0 $
Add to cart
Store
Options
AIMS
Title AddCD
Artist 1
Director 1
Cost 1
Add Media

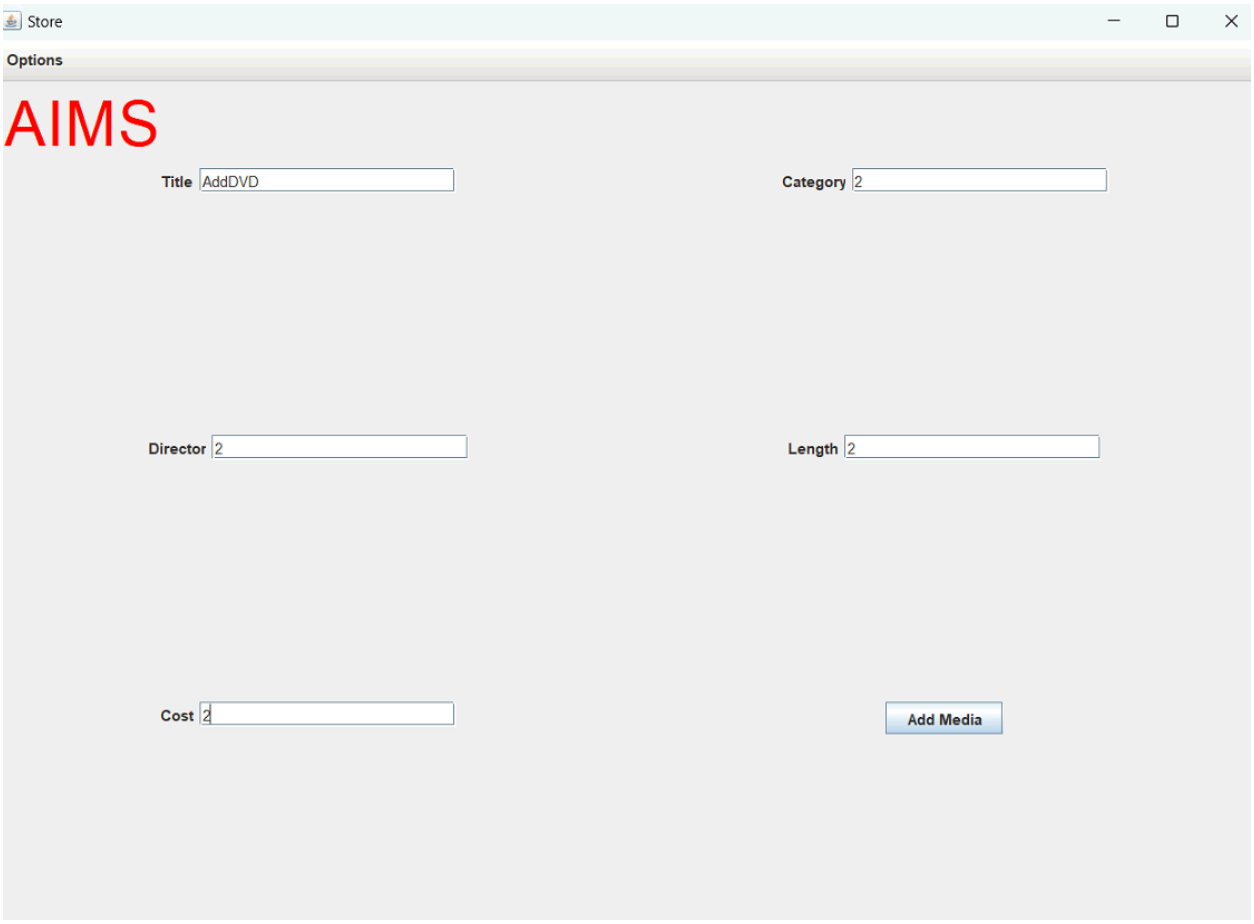

Store
Options
AIMS
View Cart
Mẹ chồng nàng dâu
24.95 $
Add to cart
Play
Winx
99.7 $
Add to cart
Play
Doraemon
90.0 $
Add to cart
Play
Nhạc vip
75.5 $
Add to cart
Play
Những ca khúc triệu view
25.99 $
Add to cart
Play
Giải tích I
20.0 $
Add to cart
Giải tích II
20.0 $
Add to cart
Giải tích III
20.0 $
Add to cart
Lập trình hướng đối tượng
9.0 $
Add to cart
ABC
1.0 $
Add to cart
AddCD
1.0 $
Add to cart
Play

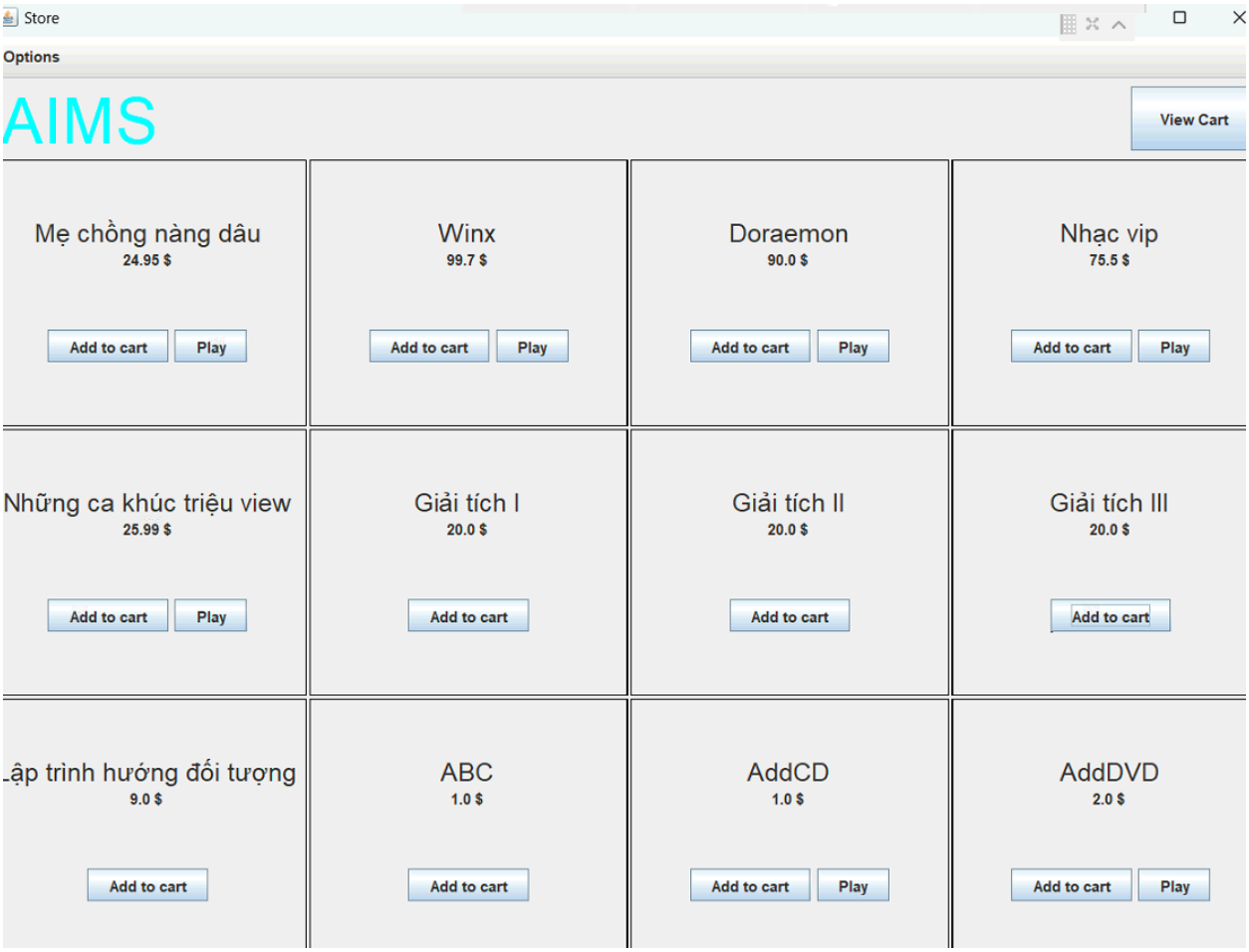
## 11.3.4 Create AddDVDToStoreScreen class

```java
/*
 * Họ và tên: Phan Thế Anh
 * MssV: 20204941
 * Mã lớp: 721428
*/
package hust.soict.dsai.aims.screen;
import hust.soict.dsai.aims.data.InitData;

public class AddDVDToStoreScreen extends AddItemToStoreScreen{
    private Store myStore = InitData.myStore;
    // Create constructor
    public AddDVDToStoreScreen() {
        Container cp = getContentPane();
        cp.setLayout(new BorderLayout());
        cp.add(createNorth(), BorderLayout.NORTH);
        cp.add(renderAddDVDScreen(), BorderLayout.CENTER);

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

    }

```

```java
    JPanel renderAddDVDScreen() {
        JPanel layout = new JPanel();
        layout.setLayout(new GridLayout(3, 3, 2, 2));
        // Input Title
        JPanel inputTitleGroup = this.inputGroup("Title", 20);
        layout.add(inputTitleGroup);
        // Input category
        JPanel inputCategoryGroup = this.inputGroup("Category", 20);
        layout.add(inputCategoryGroup);
        // Input director
        JPanel inputDirectorGroup = this.inputGroup("Director", 20);
        layout.add(inputDirectorGroup);
        // Input Length
        JPanel inputLengthGroup = this.inputGroup("Length", 20);
        layout.add(inputLengthGroup);
        // Input Cost
        JPanel inputCostGroup = this.inputGroup("Cost", 20);
        layout.add(inputCostGroup);

        // Submit button
        layout.add(submitButton(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // Create new DVD
                DigitalVideoDisc dvd = new DigitalVideoDisc(getValueTextField(inputTitleGroup),
                        getValueTextField(inputCategoryGroup), getValueTextField(inputDirectorGroup),
                        Integer.valueOf(getValueTextField(inputLengthGroup)) ,Float.valueOf(getValueTextField(inputCostGroup)));
                System.out.println(dvd);
                // Add DVD to store
                myStore.addMedia(dvd);
                System.out.println(myStore.getItemsInStore());
            }
        }));
        return layout;
    }
```

Result


Store
Options
AIMS
Title
AddDVD
Category
2
Director
2
Length
2
Cost
2
Add Media




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

Control the number of items in the cart with exception.

```java
/*
 * Họ và tên: Phan Thế Anh
 * MssV: 20204941
 * Mã lớp: 721428
 */
package hust.soict.dsai.aims.cart;
import hust.soict.dsai.aims.media.Media;

public class Cart {
    public static final int MAX_NUMBERS_ORDERD = 20;
    private ObservableList<Media> itemsOrdered =
            FXCollections.observableArrayList();

    // Add media to cart
    public boolean addMedia(Media ... medias) throws LimitExceededException {
        // Exception
        if(itemsOrdered.size() + medias.length > MAX_NUMBERS_ORDERD) {
            throw new LimitExceededException("Your cart was full!!");
        }
        for(Media media: medias) {
            if(!itemsOrdered.contains(media)) {
                itemsOrdered.add(media);
            } else {
                System.out.println(media.getTitle() + " already exists in your cart");
            }
        }

        return true;
    }

```

## 13. Create a class which inherits from **`Exception`**

### 13.1 Create new class named **`PlayerException`**

```java
/*
 * Họ và tên: Phan Thế Anh
 * MssV: 20204941
 * Mã lớp: 721428
 */
package hust.soict.dsai.aims.exception;

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

### 13.2 Raise the **`PlayerException`** in the **`play()`** method

**`DigitalVideoDisc`**

```java
/*
 * Họ và tên: Phan Thế Anh
 * MssV: 20204941
 * Mã lớp: 721428
*/

@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 {
        System.err.println("ERROR: DVD length is non-positive!");
        throw new PlayerException("ERROR: DVD length is non-positive!");
    }

}
```

**Track**

```java
/*
 * Họ và tên: Phan Thế Anh
 * MssV: 20204941
 * Mã lớp: 721428
*/
@Override
public void play() throws PlayerException {
    if(this.getLength() > 0) {
        System.out.println("Playing Track: " + this.getTitle());
        System.out.println("DVD length: " + this.getLength());
    } else {
        System.err.println("ERROR: Track length is non-positive!");
        throw new PlayerException("ERROR: Track length is non-positive!");
    }
}
```

## 13.3 Update `play()` in the `Playable` interface

Change the method signature for the **`Playable`** interface's **`play()`** method to include the throws **`PlayerException`** keywords.

InitData.java AddBookToSto... AddCDToStore... AddDVDToSto...

```java
/*
 * Họ và tên: Phan Thế Anh
 * MssV: 20204941
 * Mã lớp: 721428
*/
package hust.soict.dsai.aims.media;

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

public interface Playable {
    public void play() throws PlayerException;
}
```

### 13.4 Update `play()` in `CompactDisc`

```java
/*
 * Họ và tên: Phan Thế Anh
 * MssV: 20204941
 * Mã lớp: 721428
*/
@Override
public void play() throws PlayerException {
    for(int i = 0; i < tracks.size(); i++) {
        Track currentTrack = tracks.get(i);
        if(currentTrack.getLength() > 0) {
            tracks.get(i).play();
        } else {
            throw new PlayerException("ERROR: CD length is non-positive!");
        }

    }
}
```

## 14. Update the `Aims` class

The **`try-catch`** block is used in the main method of class **`Aims.java`** and in the **`play()`** method of the **`CompactDisc.java`**. Print all information of the exception object, e.g. **`getMessage()`**, **`toString()`**, **`printStackTrace()`**, display a dialog box to the user with the content of the exception.

```java
/*
 *  Họ và tên: Phan Thế Anh
 *  MssV: 20204941
 *  Mã lớp: 721428
 */
package hust.soict.dsai.aims;
import hust.soict.dsai.aims.cart.Cart;

public class Aims {
    private static JFrame currentView;
    private static Store myStore;
    private static Cart myCart;
    public static void main(String[] args) throws LimitExceededException {
        InitData.init();
        myStore = InitData.myStore;
        myCart = InitData.myCart;
        // Show windows
        currentView = new StoreScreen(myStore);
    }

    // Menubar
    public static class MenuItemListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            String command = e.getActionCommand();
            // View store
            if(command.equals("view_store")) {
                currentView = new StoreScreen(myStore);
            }
            // View cart
            else if(command.equals("view_cart")) {
                currentView = new CartScreen(myCart);
            }
            else if(command.equals("add_dvd")) { // Add DVD
                new AddDVDToStoreScreen();
            } else if(command.equals("add_cd")) { // Add CD
                new AddCDToStoreScreen();
            }  else if(command.equals("add_book")) { // Add Book
                new AddBookToStoreScreen();
            }
            System.out.println(e.getActionCommand() + " JMenuItem clicked.");
        }
    }

```

## 15. Modify the **`equals()`** method of **`Media`** class

- Two medias are equal if they have the same **`title`**
- Please remember to check for **`NullPointerException`** and **`ClassCastException`** if applicable.

```java
/*
 * Họ và tên: Phan Thế Anh
 * MssV: 20204941
 * Mã lớp: 721428
*/
@Override
public boolean equals(Object o) {
    if (this == o) return true;
    if (o == null || getClass() != o.getClass()) return false;
    // Kiểm tra instance xem có phải của class Media không
    if((o instanceof Media) == false) {
        return false;
    }
    Media media = (Media) o;
    return Objects.equals(title, media.title);
}
```

# 16. Summary

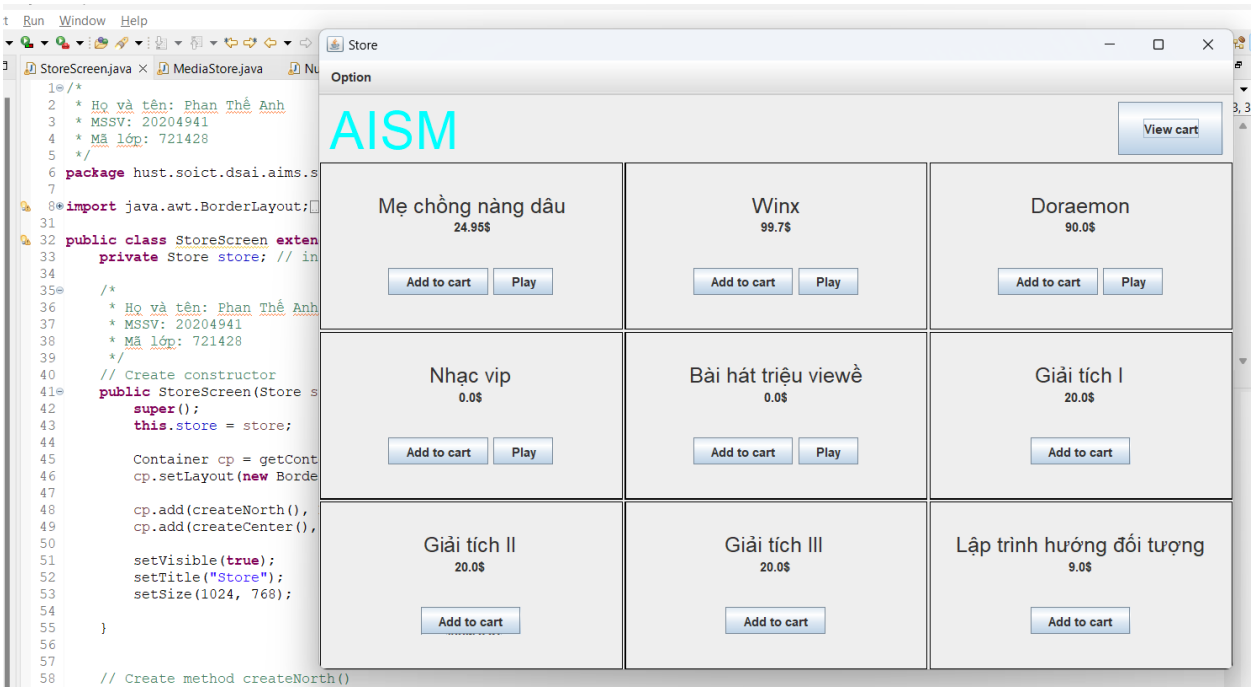
## 16.1 Functions of the program

- Store
  - Add to store
  - View store
  - Play media
- Cart
  - Add to cart
  - View cart
  - Remove to cart
  - Play media
  - Calculate total cost
  - Filter by ID
  - Filter by title
  - Place order

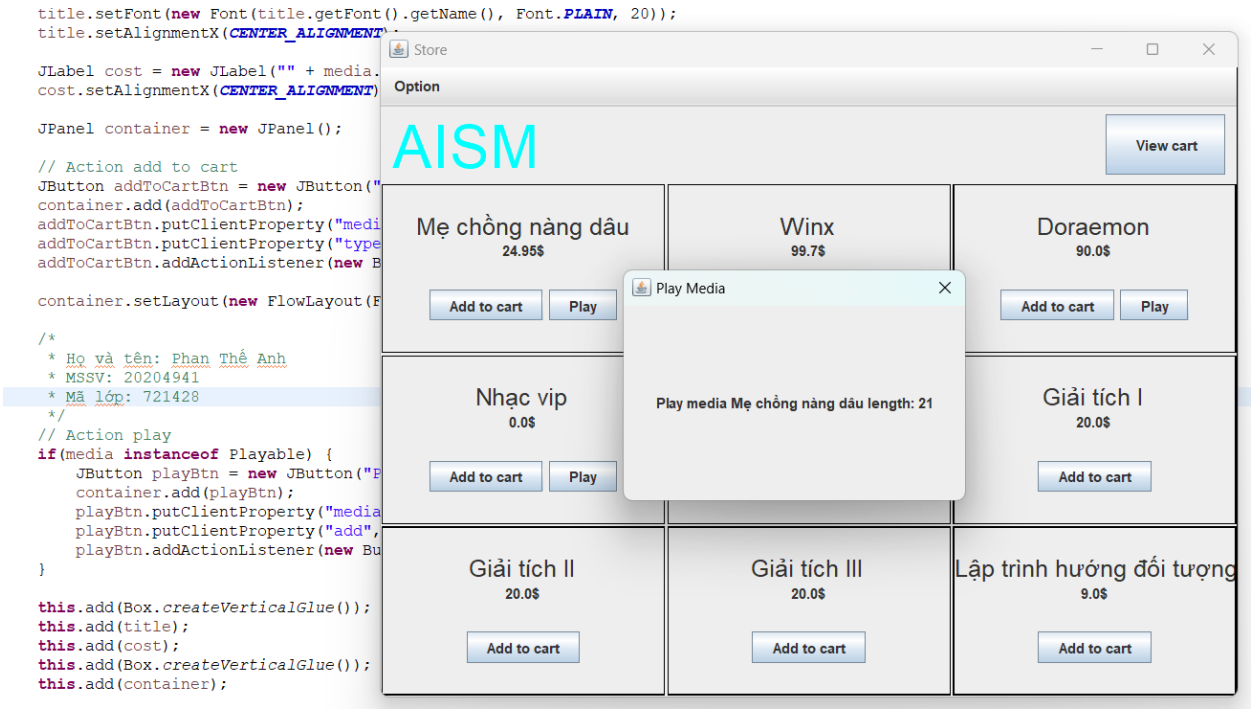
## 16.2 Demo

### 16.2.1 Store

View store



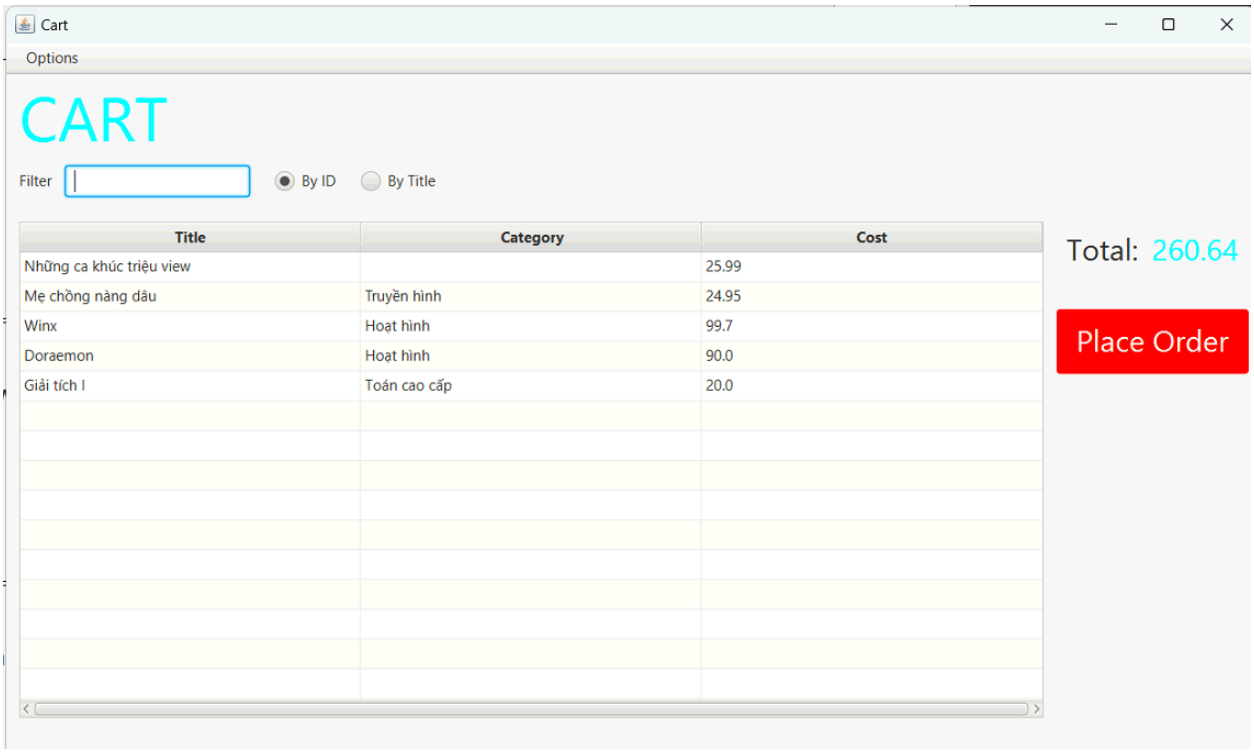

Play media



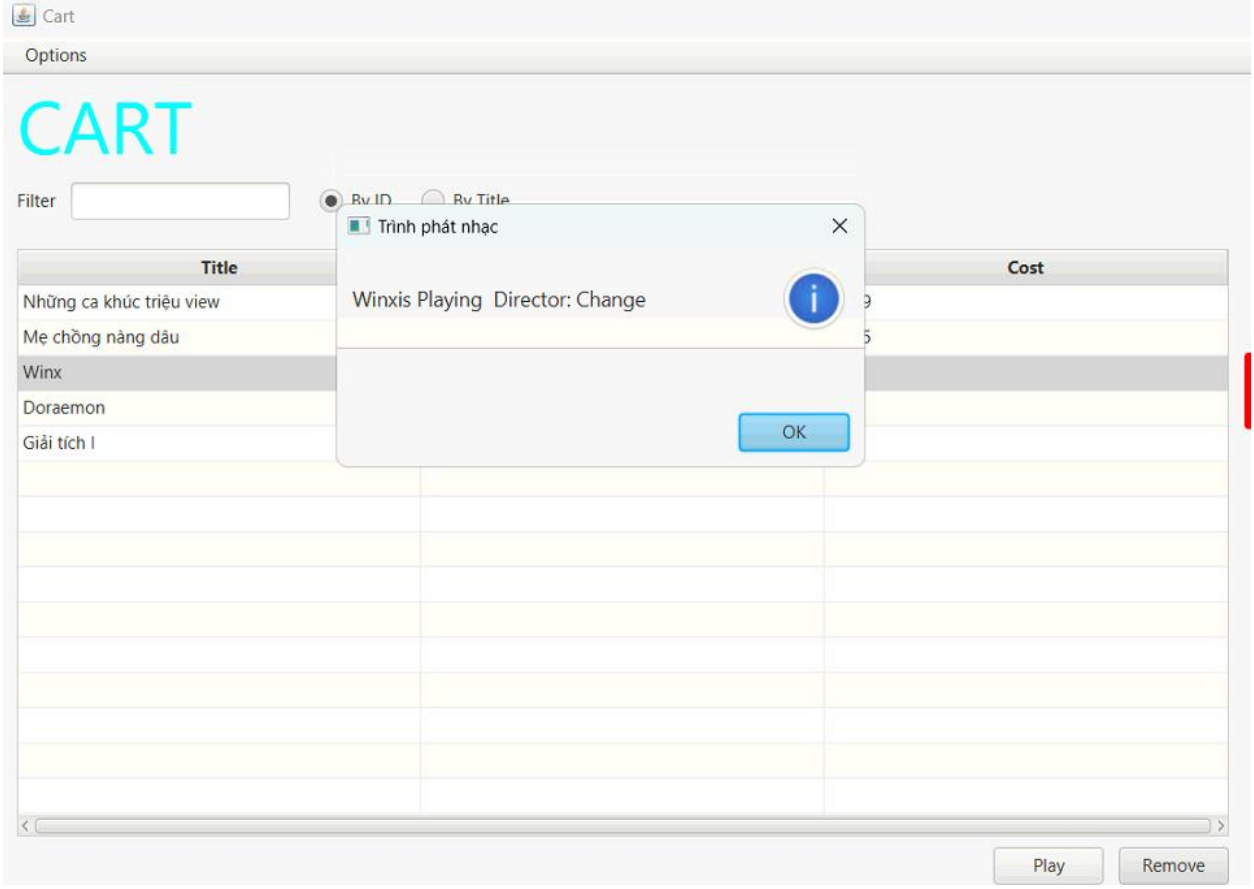

Add to store

## 16.2.2 Cart

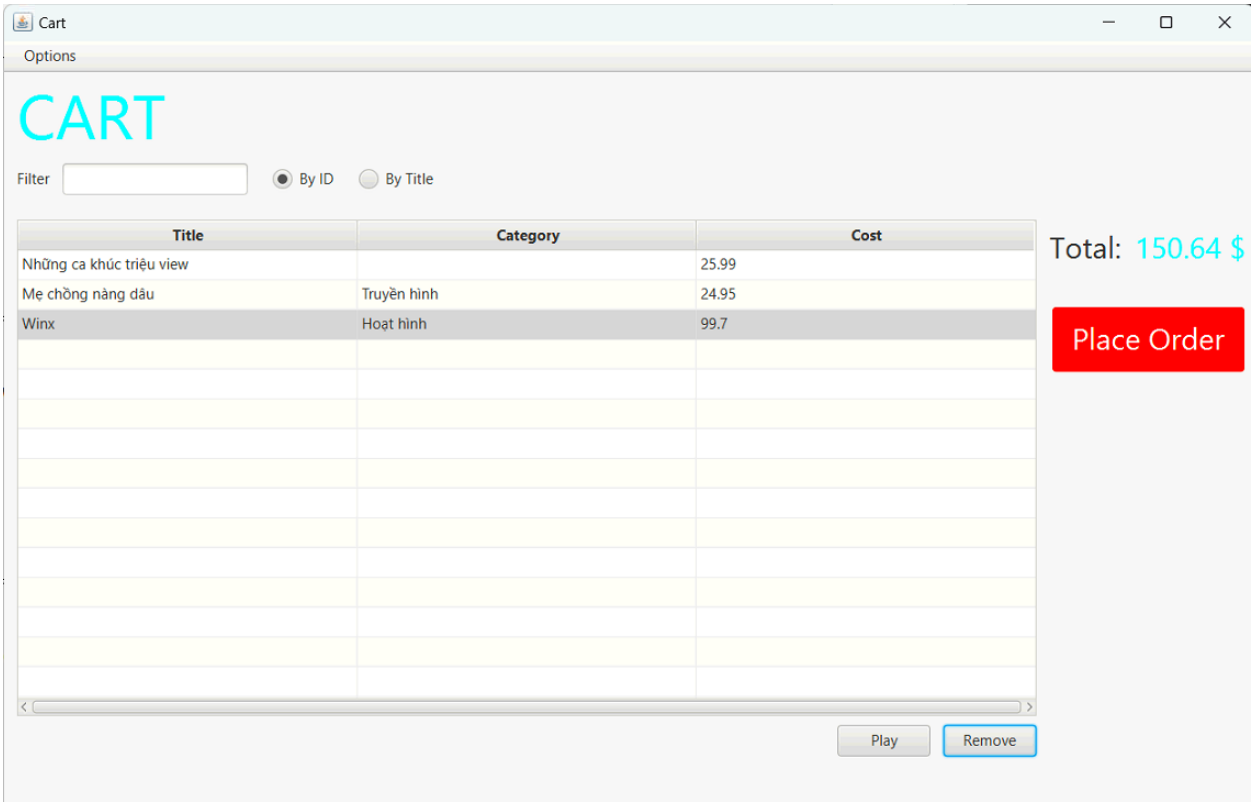
View cart



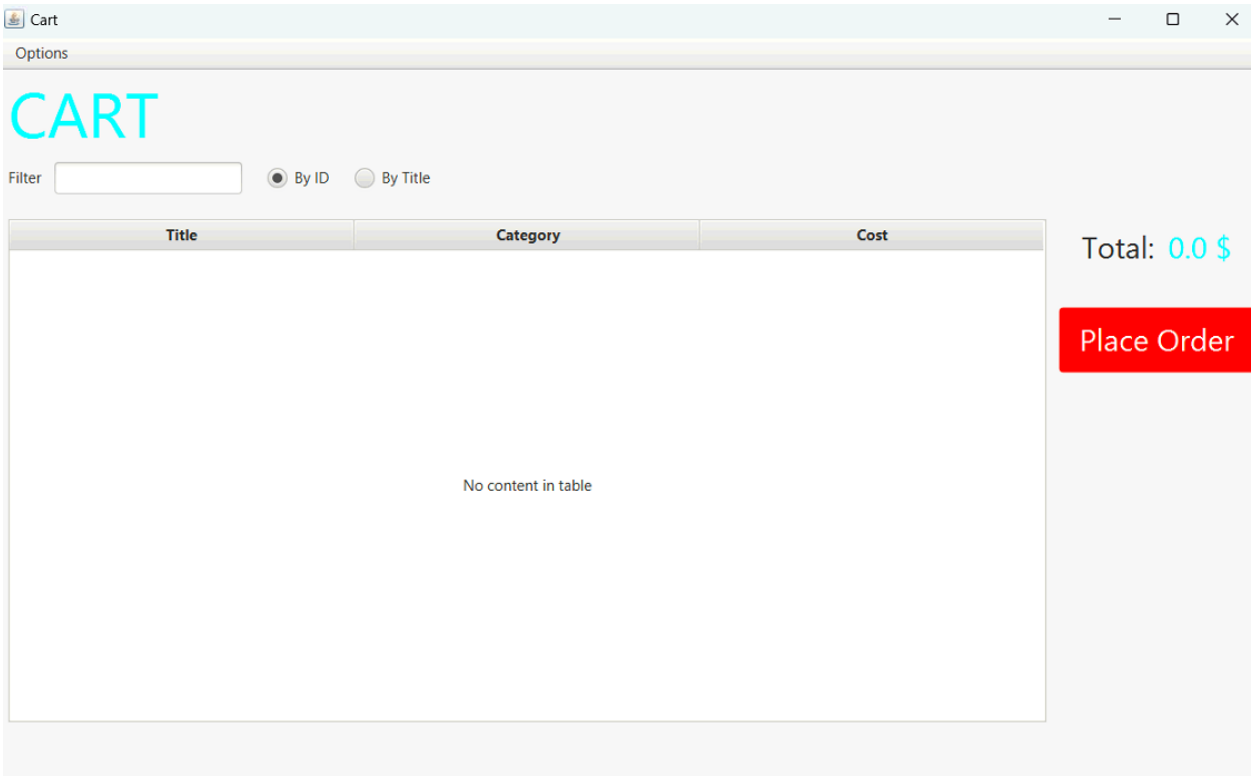

Play media

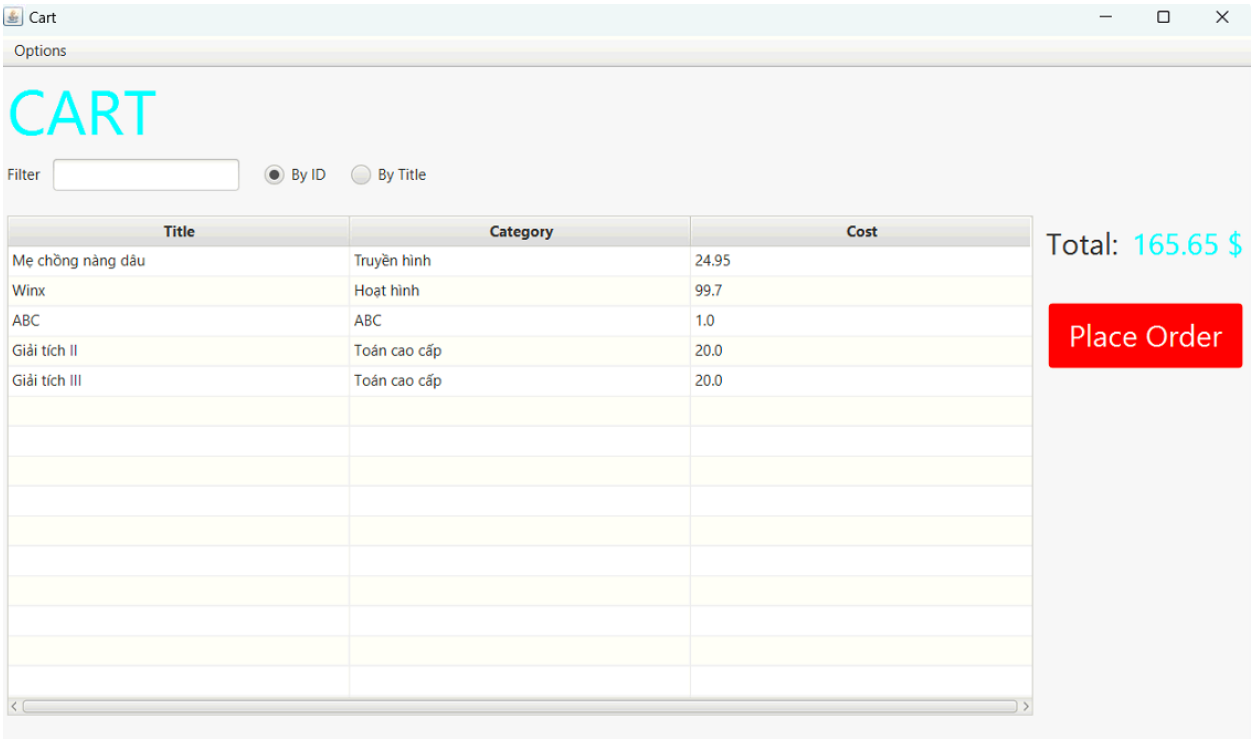



Remove media

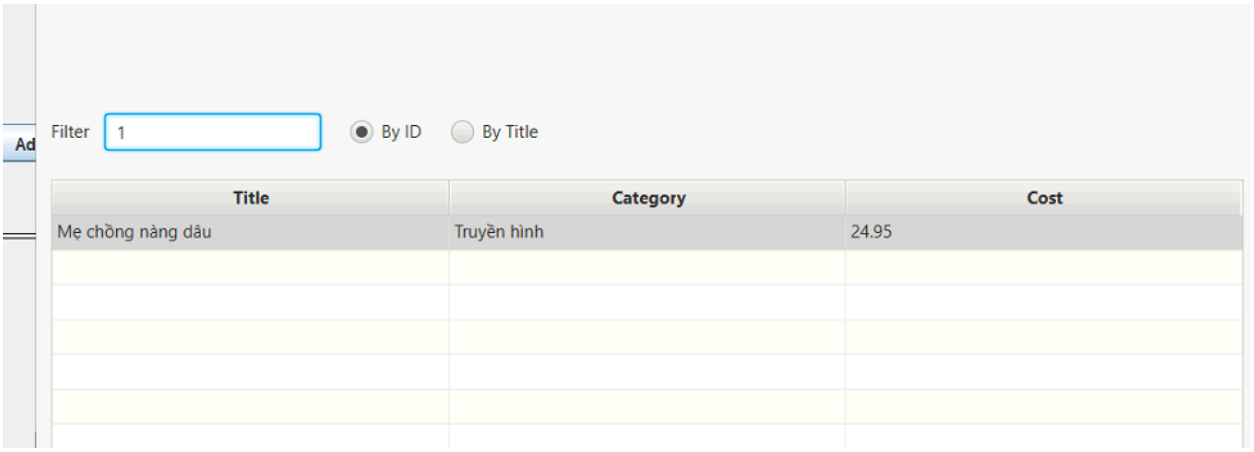

Cart
Options
CART
Filter
By ID
By Title
Title
Category
Cost
Những ca khúc triệu view
25.99
Mẹ chồng nàng dâu
Truyền hình
24.95
Winx
Hoạt hình
99.7
Total: 150.64 $
Place Order
Play
Remove

Add to cart

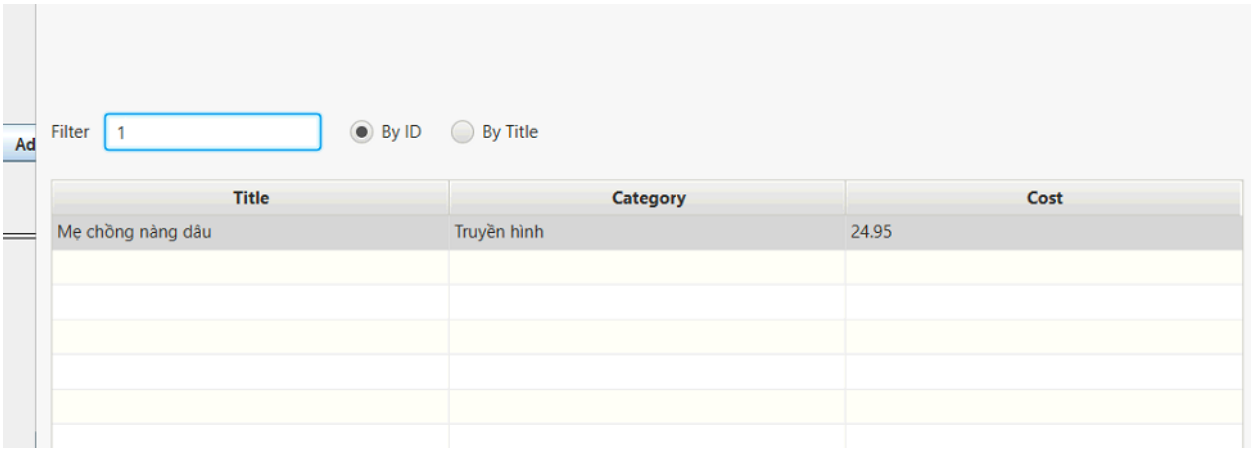

Cart
Options
CART
Filter
By ID
By Title
Title
Category
Cost
No content in table
Total: 0.0 $
Place Order



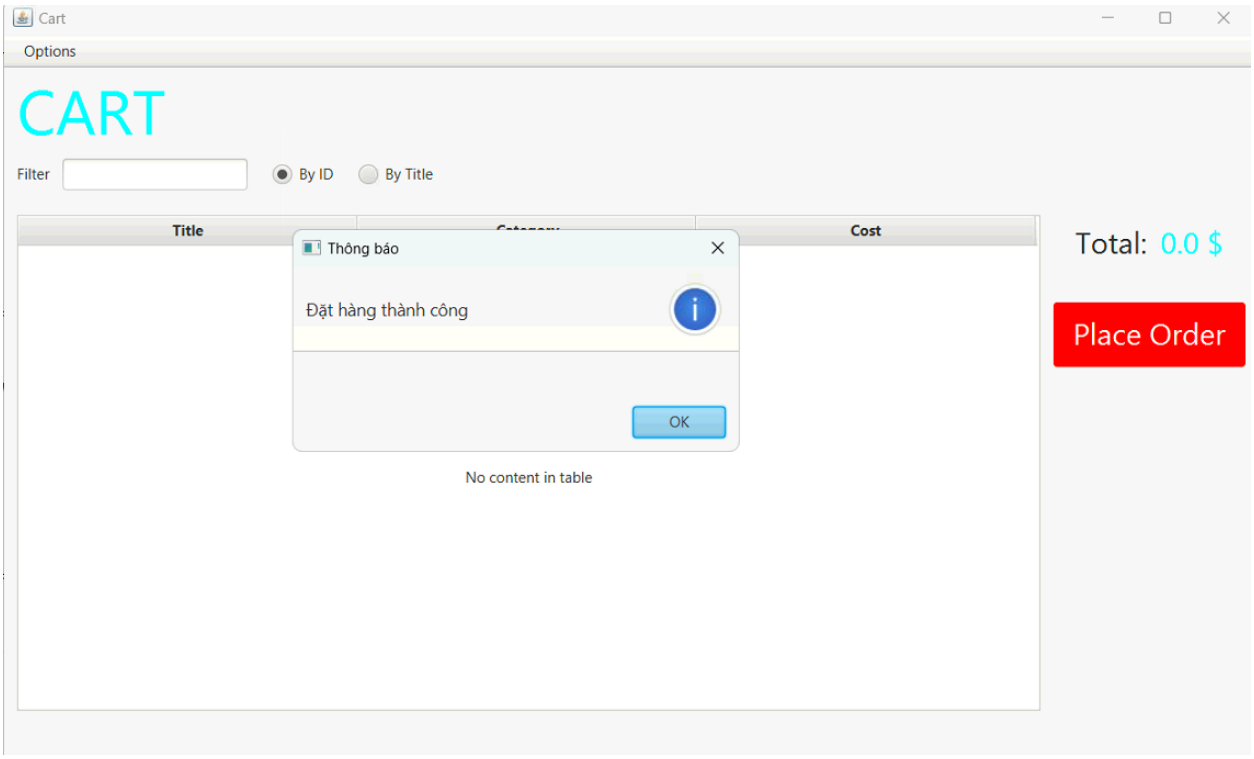

Fliter by ID




Filter by title


Filter
1
By ID
By Title
Title
Category
Cost
Mẹ chồng nàng dâu
Truyền hình
24.95

Place order


Cart
Options
CART
Filter
By ID
By Title
Title
Cost
Thông báo
Đặt hàng thành công
OK
No content in table
Total: 0.0 $
Place Order