

ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÁO CÁO THỰC HÀNH
IT3103-744528-2024.1

BÀI THỰC HÀNH LAB 5

Họ và tên sv: Nguyễn Trịnh
Hoàng Nguyên

Lớp: **K67-Việt Nhật 06**

GVHD: Lê Thị Hoa

TA: **Đặng Mạnh Cường**

Hà Nội 12/2024

Table of Contents

1.	Swing components.....	5
1.1	AWTAccumulator	5
1.2	SwingAccumulator	6
2.	Organizing Swing components with Layout Managers.....	8
2.1	Code	8
2.2	Demo.....	9
3.	Create a graphical user interface for AIMS with Swing	11
3.1	Create class StoreScreen.....	11
3.2	Create class MediaStore	15
3.3	Demo.....	18
4.	JavaFX API	21
4.1	Create class Painter.....	21
4.2	Create Painter.fxml	22
4.3	Create class PainterController	23
5.	View Cart Screen.....	27
5.1	Create cart.fxml	27
5.2	Create class CartScreen.....	29
5.3	Create class CartScreenController	30
5.4	Demo.....	35
6.	Updating buttons based on selected item in TableView – ChangeListener	36
6.1	Edit class CartScreenController.....	36
6.2	Demo.....	37
7.	Deleting a media	38
7.1	Code	38
7.2	Demo.....	39
8.	Complete the Aims GUI application.....	41
9.	equals() method in Media class	52
10.	Use case Diagram.....	53
11.	Class Diagram	54

Figure 1. Source code of AWTAccumulator 5

Figure 2.Demo of AWTAccumulator 1	5
Figure 3.Demo of AWTAccumulator 2	6
Figure 4.Demo of AWTAccumulator 3	6
Figure 5.Source code of SwingAccumulator 1	6
Figure 6.Source code of SwingAccumulator 2	7
Figure 7.Demo of SwingAccumulator 1	7
Figure 8.Demo of SwingAccumulator 2	7
Figure 9.Demo of SwingAccumulator 3	7
Figure 10.Source code of NumberGrid 1	8
Figure 11.Source code of NumberGrid 2	9
Figure 12.Demo buttons 0-9	9
Figure 13.Demo DEL button.....	10
Figure 14.Demo C button.....	10
Figure 15.Class StoreScreen 1.....	11
Figure 16.Class StoreScreen 2.....	12
Figure 17.Class StoreScreen 3.....	13
Figure 18.Class StoreScreen 4.....	14
Figure 19.Class MediaStore 1.....	15
Figure 20.Class MediaStore 2.....	16
Figure 21.Class MediaStore 3.....	17
Figure 22.Class MediaStore 4.....	17
Figure 23.StoreScreen.....	18
Figure 24.Demo Add to cart button.....	19
Figure 25.Demo Play button	19
Figure 26.Demo View cart button.....	20
Figure 27.Class Painter.....	21
Figure 28.Painter.fxml 1.....	22
Figure 29.Painter.fxml 2.....	22
Figure 30.Class PainterController	23
Figure 31.Use Pen	24
Figure 32.Use Eraser	25
Figure 33.Clear button	26
Figure 34.Cart.fxml 1.....	27
Figure 35.Cart.fxml 2.....	28
Figure 36.Cart.fxml 3.....	28
Figure 37.CartScreen class	29
Figure 38.CartScreenController 1	30
Figure 39.CartScreenController 2	31
Figure 40.CartScreenController 3	32
Figure 41.CartScreenController 4	33
Figure 42.CartScreenController 5	34
Figure 43.CartScreenController 6	35
Figure 44.Demo CartScreen	35
Figure 45.CartScreenController 1	36
Figure 46.CartScreenController 2	36
Figure 47.Demo media unplayable	37
Figure 48.Demo media playable	37
Figure 49.btnRemovePressed Method	38

Figure 50.button Remove 1	39
Figure 51.button Remove 2	40
Figure 52.Store before add book.....	41
Figure 53.Add book.....	42
Figure 54.Store after add book.....	43
Figure 55.Add CD	44
Figure 56.Store after add CD.....	45
Figure 57.Add DVDFigure 58.Store after add DVD	46
Figure 59.Test exception with DVD Length = 0.....	48
Figure 60.Add test to Cart.....	49
Figure 61.play test.....	50
Figure 62.Exception.....	50
Figure 63.Place Order	51
Figure 64.After Place Order	52
Figure 65.equals() method.....	52
Figure 66.UC Diagram 1	53
Figure 67..UC Diagram 2	53
Figure 68.Class Diagram.....	54

1. Swing components

1.1 AWTAccumulator

```

1 package hust.soict.dsai.swing;
2
3 import java.awt.*;
4 import java.awt.event.ActionEvent;
5 import java.awt.event.ActionListener;
6
7 public class AWTAccumulator extends Frame {
8     private TextField tfInput; 5 usages
9     private TextField tfOutput; 4 usages
10    private int sum = 0; 2 usages
11
12    public AWTAccumulator() { 1 usage
13        setLayout(new GridLayout( rows: 2, cols: 2));
14
15        add(new Label( text: "Enter an Integer: "));
16
17        tfInput = new TextField( columns: 10);
18        add(tfInput);
19        tfInput.addActionListener(new TFInputListener());
20
21        add(new Label( text: "The Accumulated Sum is: "));
22
23        tfOutput = new TextField( columns: 10);
24        tfOutput.setEditable(false);
25        add(tfOutput);
26
27        setTitle("AWT Accumulator");
28        setSize( width: 350, height: 120);
29        setVisible(true);
30    }
31
32    public static void main(String[] args) { new AWTAccumulator(); }
33
34    private class TFInputListener implements ActionListener { 1 usage
35        @Override
36        public void actionPerformed(ActionEvent evt) {
37            int numberIn = Integer.parseInt(tfInput.getText());
38            sum += numberIn;
39            tfInput.setText("");
40            tfOutput.setText(sum + "");
41        }
42    }
43
44}
45
46

```

Figure 1. Source code of AWTAccumulator

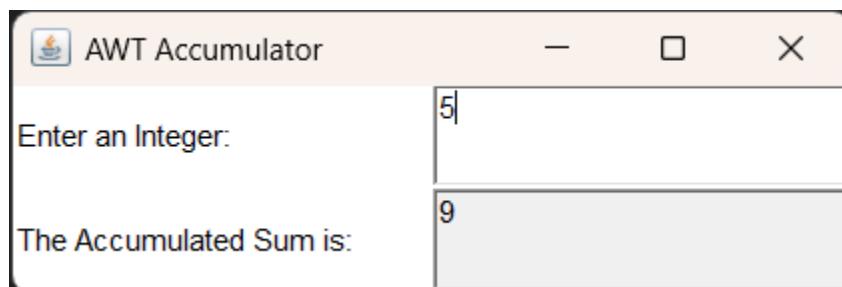


Figure 2.Demo of AWTAccumulator 1

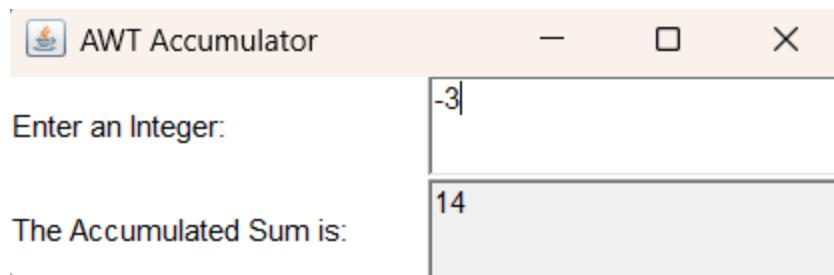


Figure 3.Demo of AWTAccumulator 2

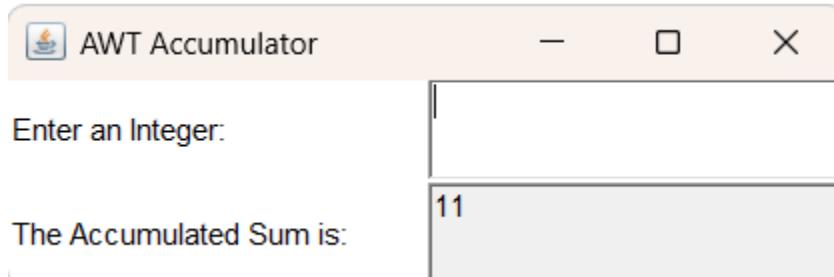


Figure 4.Demo of AWTAccumulator 3

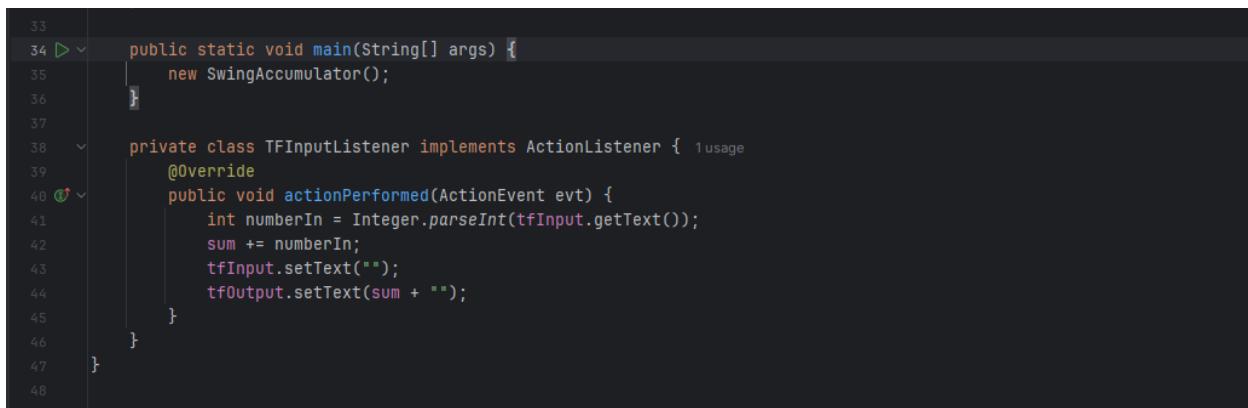
1.2 SwingAccumulator

```

1 package hust.soict.dsai.swing;
2
3 import javax.swing.*;
4 import java.awt.*;
5 import java.awt.event.ActionEvent;
6 import java.awt.event.ActionListener;
7
8 public class SwingAccumulator extends JFrame {
9     private JTextField tfInput; 5 usages
10    private JTextField tfOutput; 4 usages
11    private int sum = 0; 2 usages
12
13    public SwingAccumulator() { 1 usage
14        Container cp = getContentPane();
15        cp.setLayout(new GridLayout( rows: 2, cols: 2));
16
17        cp.add(new JLabel( text: "Enter an Integer: "));
18
19        tfInput = new JTextField( columns: 10);
20        cp.add(tfInput);
21        tfInput.addActionListener(new TFIinputListener());
22
23        cp.add(new JLabel( text: "The Accumulated Sum is: "));
24
25        tfOutput = new JTextField( columns: 10);
26        tfOutput.setEditable(false);
27        cp.add(tfOutput);
28
29        setTitle("Swing Accumulator");
30        setSize( width: 350, height: 120);
31        setVisible(true);
32    }

```

Figure 5.Source code of SwingAccumulator 1



```

33
34 public static void main(String[] args) {
35     new SwingAccumulator();
36 }
37
38 private class TFIInputListener implements ActionListener { 1 usage
39     @Override
40     public void actionPerformed(ActionEvent evt) {
41         int numberIn = Integer.parseInt(tfInput.getText());
42         sum += numberIn;
43         tfInput.setText("");
44         tfOutput.setText(sum + "");
45     }
46 }
47 }
48

```

Figure 6.Source code of SwingAccumulator 2

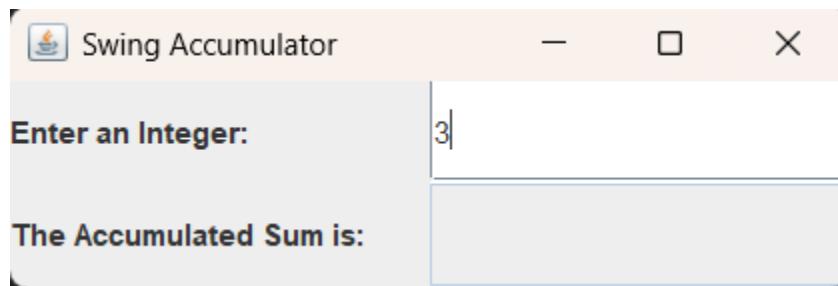


Figure 7.Demo of SwingAccumulator 1

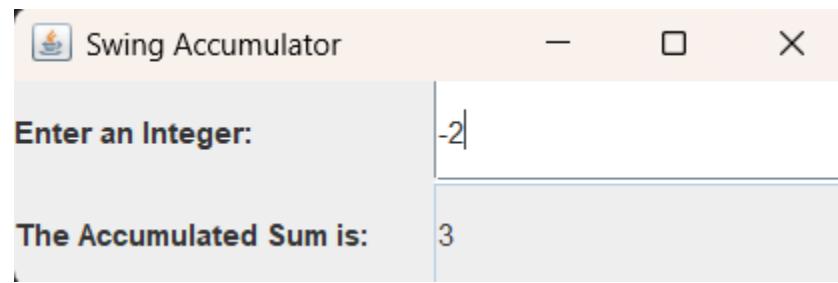


Figure 8.Demo of SwingAccumulator 2

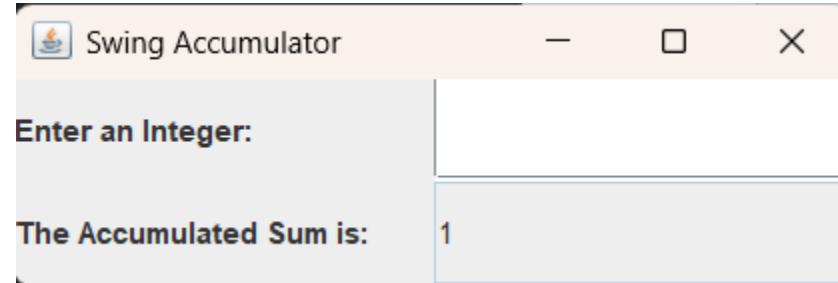


Figure 9.Demo of SwingAccumulator 3

2. Organizing Swing components with Layout Managers

2.1 Code

```

1 package hust.soict.dsai.swing;
2
3 import javax.swing.*;
4 import java.awt.*;
5 import java.awt.event.ActionEvent;
6 import java.awt.event.ActionListener;
7
8 public class NumberGrid extends JFrame {
9     public static void main(String[] args) {
10         new NumberGrid();
11     }
12
13     private JButton[] btnNumbers = new JButton[10];  6 usages
14     private JButton btnDelete, btnReset;  3 usages
15     private JTextField tfDisplay;  8 usages
16
17     public NumberGrid() {  1 usage
18
19         tfDisplay = new JTextField();
20         tfDisplay.setComponentOrientation(
21             ComponentOrientation.RIGHT_TO_LEFT);
22
23         JPanel panelButtons = new JPanel(new GridLayout( rows: 4, cols: 3));
24         addButtons(panelButtons);
25
26         Container cp = getContentPane();
27         cp.setLayout(new BorderLayout());
28         cp.add(tfDisplay, BorderLayout.NORTH);
29         cp.add(panelButtons, BorderLayout.CENTER);
30
31         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
32         setTitle("Number Grid");
33         setSize( width: 200, height: 200);
34         setVisible(true);
35     }
36
37     void addButtons(JPanel panelButtons) {  1 usage
38         ButtonListener btnListener = new ButtonListener();
39         for (int i = 1; i <= 9; i++) {
40             btnNumbers[i] = new JButton( text: "" + i);
41             panelButtons.add(btnNumbers[i]);
42             btnNumbers[i].addActionListener(btnListener);
43         }
44     }

```

Figure 10. Source code of NumberGrid 1

```

44
45     btnDelete = new JButton( text: "DEL");
46     panelButtons.add(btnDelete);
47     btnDelete.addActionListener(btnListener);
48
49     btnNumbers[0] = new JButton( text: "0");
50     panelButtons.add(btnNumbers[0]);
51     btnNumbers[0].addActionListener(btnListener);
52
53     btnReset = new JButton( text: "C");
54     panelButtons.add(btnReset);
55     btnReset.addActionListener(btnListener);
56 }
57
58 private class ButtonListener implements ActionListener { 2 usages
59     @Override
60     public void actionPerformed(ActionEvent e) {
61         String button = e.getActionCommand();
62         if (button.charAt(0) >= '0' && button.charAt(0) <= '9') {
63             tfDisplay.setText(tfDisplay.getText() + button);
64         } else if (button.equals("DEL")) {
65             String currentText = tfDisplay.getText();
66             if (!currentText.isEmpty()) {
67                 tfDisplay.setText(currentText.substring(0, currentText.length() - 1));
68             }
69         } else {
70             tfDisplay.setText("");
71         }
72     }
73 }
74 }
75

```

Figure 11. Source code of NumberGrid 2

2.2 Demo

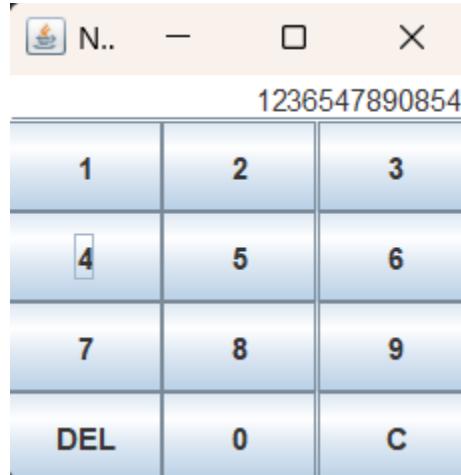


Figure 12. Demo buttons 0-9

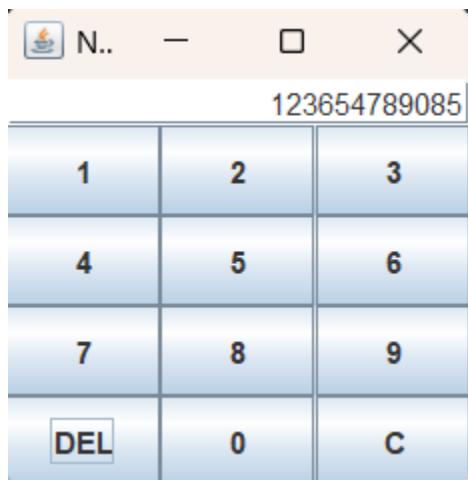


Figure 13.Demo DEL button

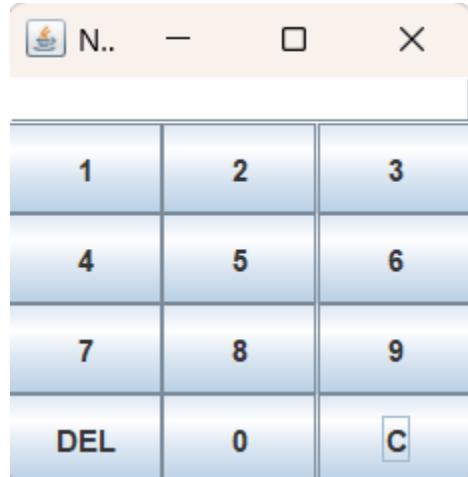


Figure 14.Demo C button

3. Create a graphical user interface for AIMS with Swing

3.1 Create class StoreScreen

```

1 package hust.soict.dsai.aims.screen;
2
3 import hust.soict.dsai.aims.screen.*;
4 import hust.soict.dsai.aims.store.*;
5 import hust.soict.dsai.aims.cart.*;
6 import hust.soict.dsai.aims.exception.*;
7 import hust.soict.dsai.aims.media.*;
8
9 import java.awt.BorderLayout;
10 import java.awt.Color;
11 import java.awt.Container;
12 import java.awt.Dimension;
13 import java.awt.FlowLayout;
14 import java.awt.Font;
15 import java.awt.GridLayout;
16 import java.awt.Toolkit;
17 import java.awt.event.ActionEvent;
18 import java.awt.event.ActionListener;
19 import java.util.ArrayList;
20 import java.util.Arrays;
21
22 import javax.swing.Box;
23 import javax.swing.BoxLayout;
24 import javax.swing.JButton;
25 import javax.swing.JFrame;
26 import javax.swing.JLabel;
27 import javax.swing.JMenu;
28 import javax.swing.JMenuBar;
29 import javax.swing.JMenuItem;
30 import javax.swing.JPanel;
31
32 public class StoreScreen extends JFrame {    ▲ Hoàng Nguyễn
33     private Store store;  6 usages
34     private Cart cart;  6 usages
35
36     public StoreScreen(Store store, Cart cart) {  5 usages ▲ Hoàng Nguyễn
37         this.store = store;
38         this.cart = cart;
39         Container cp = getContentPane();
40         cp.setLayout(new BorderLayout());
41         cp.add(createNorth(), BorderLayout.NORTH);
42         cp.add(createCenter(), BorderLayout.CENTER);
43         setVisible(true);
44         setTitle("Store");
45         setSize( width: 1024, height: 768 );

```

Figure 15. Class StoreScreen 1

```

42     cp.add(createCenter(), BorderLayout.CENTER);
43     setVisible(true);
44     setTitle("Store");
45     setSize( width: 1024, height: 768 );
46     Dimension dim = Toolkit.getDefaultToolkit().getScreenSize();
47     int w = getSize().width;
48     int h = getSize().height;
49     int x = (dim.width - w) / 2;
50     int y = (dim.height - h) / 2;
51     setLocation(x, y);
52 }
53
54 JPanel createNorth() { 1 usage ± Hoàng Nguyễn
55     JPanel north = new JPanel();
56     north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
57     north.add(createMenuBar());
58     north.add(createHeader());
59     return north;
60 }
61
62 JPanel createCenter() { 1 usage ± Hoàng Nguyễn
63     JPanel center = new JPanel();
64     center.setLayout(new GridLayout( rows: 5, cols: 5, hgap: 2, vgap: 2));
65     ArrayList<Media> mediaInStore = store.getItemsInStore();
66     for (Media media : mediaInStore) {
67         MediaStore cell = new MediaStore(media, cart);
68         center.add(cell);
69     }
70     return center;
71 }
72
73 JMenuBar createMenuBar() { 1 usage ± Hoàng Nguyễn
74     JMenu menu = new JMenu( s: "Options");
75
76     JMenu smUpdateStore = new JMenu( s: "Update Store");
77     JMenuItem addBook = new JMenuItem( text: "Add Book");
78     addBook.addActionListener(new AddBookListener());
79     smUpdateStore.add(addBook);
80     JMenuItem addCD = new JMenuItem( text: "Add CD");
81     addCD.addActionListener(new AddCDListener());
82     smUpdateStore.add(addCD);
83     JMenuItem addDVD = new JMenuItem( text: "Add DVD");
84     addDVD.addActionListener(new AddDVDListener());

```

Figure 16. Class StoreScreen 2

```

84     addDVD.addActionListener(new AddDVDListener());
85     smUpdateStore.add(addDVD);
86
87     menu.add(smUpdateStore);
88     menu.add(new JMenuItem( text: "View store"));
89     JMenuItem cart = new JMenuItem( text: "View cart");
90     cart.addActionListener(new ViewCartListener());
91     menu.add(cart);
92
93     JMenuBar menuBar = new JMenuBar();
94     menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
95     menuBar.add(menu);
96
97     return menuBar;
98 }
99
100 JPanel createHeader() { 1 usage ▾ Hoàng Nguyễn
101     JPanel header = new JPanel();
102     header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));
103
104     JLabel title = new JLabel( text: "AIMS:Nguyen Trinh Hoang Nguyen 20225656");
105     title.setFont(new Font(title.getFont().getName(), Font.PLAIN, size: 50));
106     title.setForeground(Color.CYAN);
107
108     JButton cart = new JButton( text: "View cart");
109     cart.setPreferredSize(new Dimension( width: 100, height: 50));
110     cart.setMaximumSize(new Dimension( width: 100, height: 50));
111     cart.addActionListener(new ViewCartListener());
112
113     header.add(Box.createRigidArea(new Dimension( width: 10, height: 10)));
114     header.add(title);
115     header.add(Box.createHorizontalGlue());
116     header.add(cart);
117     header.add(Box.createRigidArea(new Dimension( width: 10, height: 10)));
118
119     return header;
120 }
121
122 private class ViewCartListener implements ActionListener { 2 usages ▾ Hoàng Nguyễn
123     @Override
124     public void actionPerformed(ActionEvent e) {
125         new CartScreen(store, cart);
126         dispose();
127     }

```

Figure 17. Class StoreScreen 3

```
122     private class ViewCartListener implements ActionListener { 2 usages ▾ Hoàng Nguyễn
123         @Override ▾ Hoàng Nguyễn
124         public void actionPerformed(ActionEvent e) {
125             new CartScreen(store, cart);
126             dispose();
127         }
128     }
129
130
131     private class AddDVDListener implements ActionListener { 1 usage ▾ Hoàng Nguyễn
132         @Override ▾ Hoàng Nguyễn
133         public void actionPerformed(ActionEvent e) {
134             new AddDVDToStoreScreen(store, cart);
135             dispose();
136         }
137     }
138
139
140     private class AddBookListener implements ActionListener { 1 usage ▾ Hoàng Nguyễn
141         @Override ▾ Hoàng Nguyễn
142         public void actionPerformed(ActionEvent e) {
143             new AddBookToStoreScreen(store, cart);
144             dispose();
145         }
146     }
147
148
149     private class AddCDListener implements ActionListener { 1 usage ▾ Hoàng Nguyễn
150         @Override ▾ Hoàng Nguyễn
151         public void actionPerformed(ActionEvent e) {
152             new AddCDToStoreScreen(store, cart);
153             dispose();
154         }
155     }
156
157
158 }
```

Figure 18. Class StoreScreen 4

3.2 Create class MediaStore

```

1 package hust.soict.dsai.aims.screen;
2
3 import hust.soict.dsai.aims.media.*;
4 import hust.soict.dsai.aims.cart.Cart;
5 import hust.soict.dsai.aims.exception.*;
6
7 import java.awt.Color;
8 import java.awt.Component;
9 import java.awt.Dimension;
10 import java.awt.FlowLayout;
11 import java.awt.Font;
12 import java.awt.Toolkit;
13 import java.awt.event.ActionEvent;
14 import java.awt.event.ActionListener;
15
16 import javax.swing.BorderFactory;
17 import javax.swing.Box;
18 import javax.swing.BoxLayout;
19 import javax.swing.JButton;
20 import javax.swing.JDialog;
21 import javax.swing.JLabel;
22 import javax.swing.JPanel;
23 import javax.swing.SwingConstantsConstants;
24
25 public class MediaStore extends JPanel { ▲ Hoàng Nguyễn
26     private Media media; 6 usages
27     private Cart cart; 2 usages
28
29     @
30     public MediaStore(Media media, Cart cart) { 1 usage ▲ Hoàng Nguyễn
31         this.media = media;
32         this.cart = cart;
33         this.setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));
34
35         JLabel title = new JLabel(media.getTitle());
36         title.setFont(new Font(title.getFont().getName(), Font.PLAIN, size: 20));
37         title.setAlignmentX(CENTER_ALIGNMENT);
38
39         JLabel cost = new JLabel( text: "" + media.getCost() + " $");
40         cost.setAlignmentX(CENTER_ALIGNMENT);
41
42         JPanel container = new JPanel();
43         container.setLayout(new FlowLayout(FlowLayout.CENTER));
44
45         JButton addToCartButton = new JButton( text: "Add to cart");

```

Figure 19. Class MediaStore 1

```
42
43
44     JButton addToCartButton = new JButton( text: "Add to cart");
45     addToCartButton.addActionListener(new AddToCartListener());
46     container.add(addToCartButton);
47
48     if (media instanceof Playable) {
49         JButton playButton = new JButton( text: "Play");
50         playButton.addActionListener(new PlayButtonListener());
51         container.add(playButton);
52     }
53
54     this.add(Box.createVerticalGlue());
55     this.add(title);
56     this.add(cost);
57     this.add(Box.createVerticalGlue());
58     this.add(container);
59
60     this.setBorder(BorderFactory.createLineBorder(Color.BLACK));
61 }
62
63 private class PlayButtonListener implements ActionListener { 1usage * Hoàng Nguyên
64
65     @Override * Hoàng Nguyên
66     public void actionPerformed(ActionEvent e) {
67         try {
68             ((Playable) media).play();
69         } catch (PlayerException ex) {
70             JPanel p = new JPanel();
71             JDialog d = new JDialog();
72             JLabel j1 = new JLabel( text: media.getTitle() + " cannot be played");
73             JLabel j2 = new JLabel( text: "Media length is non-positive");
74             p.setLayout(new BoxLayout(p, BoxLayout.Y_AXIS));
75             j1.setAlignmentX(Component.CENTER_ALIGNMENT);
76             j2.setAlignmentX(Component.CENTER_ALIGNMENT);
77             p.add(Box.createVerticalGlue());
78             p.add(j1);
79             p.add(j2);
80             p.add(Box.createVerticalGlue());
```

Figure 20. Class MediaStore 2

```

80
81         p.add(Box.createVerticalGlue());
82         d.add(p);
83         d.setSize( width: 250, height: 100);
84         Dimension dim = Toolkit.getDefaultToolkit().getScreenSize();
85         int w = d.getSize().width;
86         int h = d.getSize().height;
87         int x = (dim.width - w) / 2;
88         int y = (dim.height - h) / 2;
89         d.setLocation(x, y);
90         d.setVisible(true);
91     }
92 }
93
94 private class AddToCartListener implements ActionListener { 1 usage ± Hoàng Nguyễn
95
96     @Override ± Hoàng Nguyễn
97     public void actionPerformed(ActionEvent e) {
98         JPanel p = new JPanel();
99         JDialog d = new JDialog();
100        JLabel l = new JLabel();
101        try {
102            cart.addMedia(media);
103            l.setText(media.getTitle() + " added to cart");
104        } catch (CartFullException ex) {
105            l.setText("The cart is full");
106        } finally {
107            p.setLayout(new BoxLayout(p, BoxLayout.Y_AXIS));
108            l.setAlignmentX(Component.CENTER_ALIGNMENT);
109            p.add(Box.createVerticalGlue());
110            p.add(l);
111            p.add(Box.createVerticalGlue());
112            d.add(p);

```

Figure 21. Class MediaStore 3

```

113         d.add(p);
114         d.setSize( width: 200, height: 100);
115         Dimension dim = Toolkit.getDefaultToolkit().getScreenSize();
116         int w = d.getSize().width;
117         int h = d.getSize().height;
118         int x = (dim.width - w) / 2;
119         int y = (dim.height - h) / 2;
120         d.setLocation(x, y);
121         d.setVisible(true);
122     }
123 }
124

```

Figure 22. Class MediaStore 4

3.3 Demo

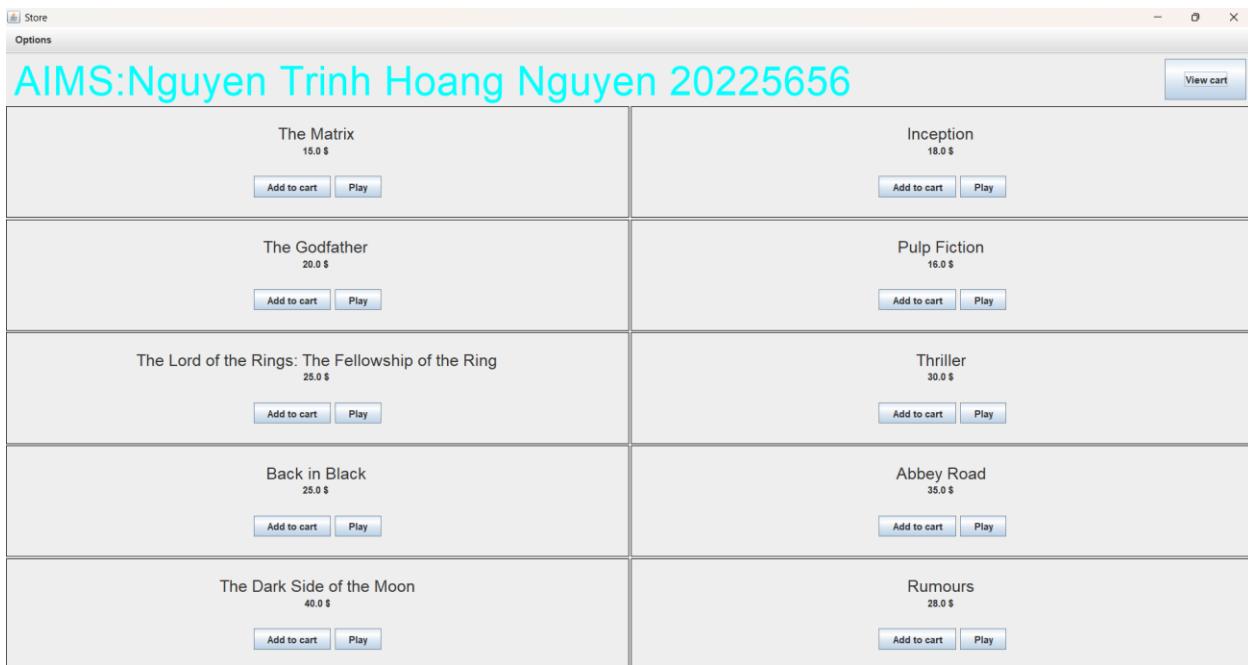


Figure 23. StoreScreen



Figure 24.Demo Add to cart button

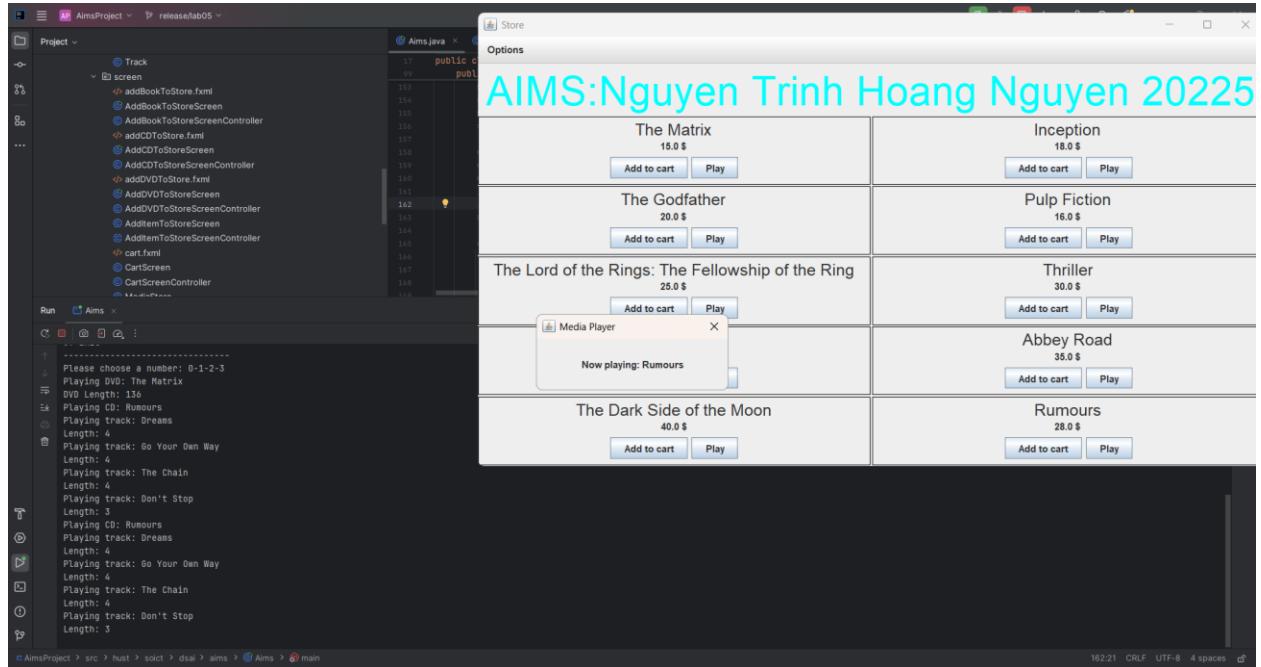


Figure 25.Demo Play button

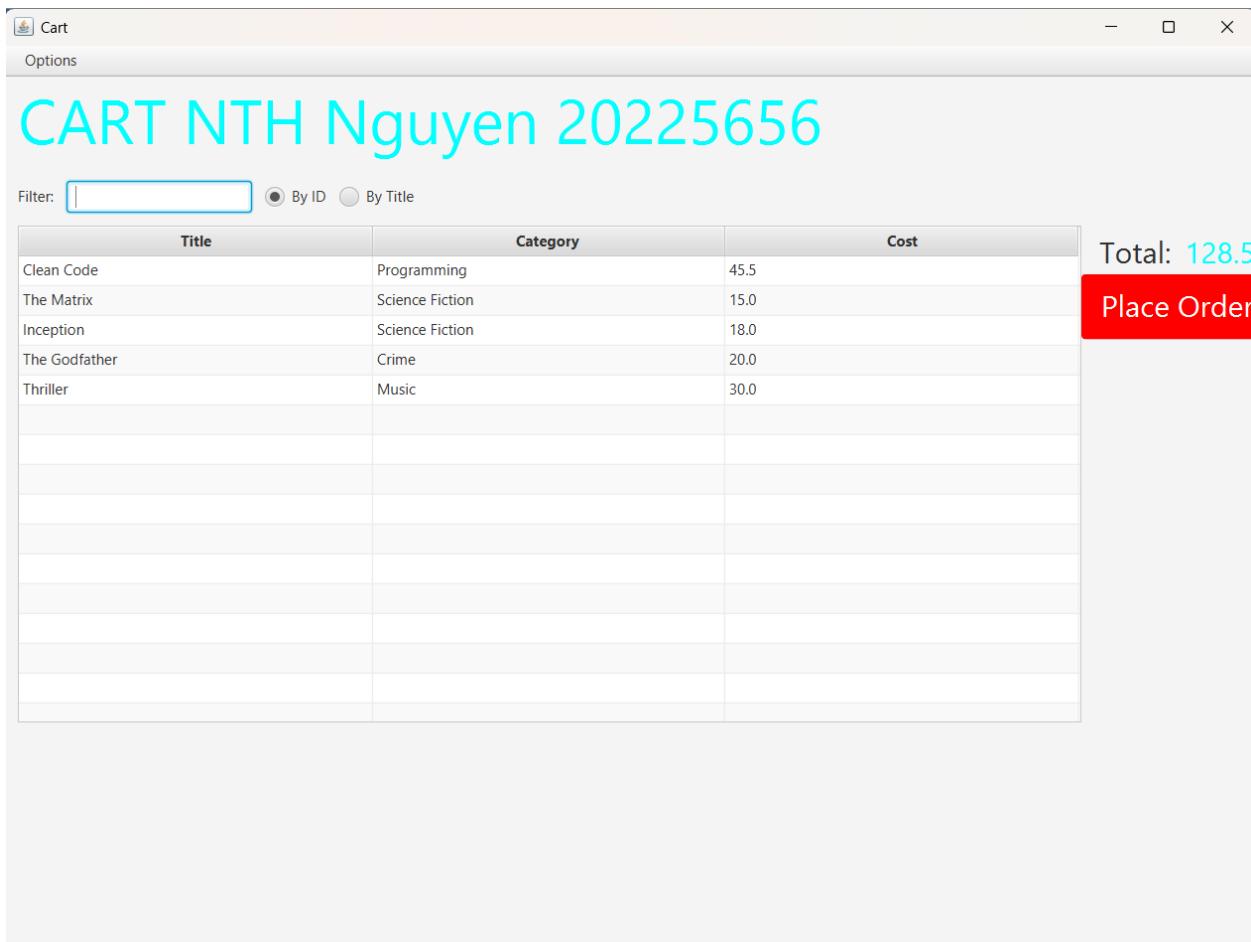
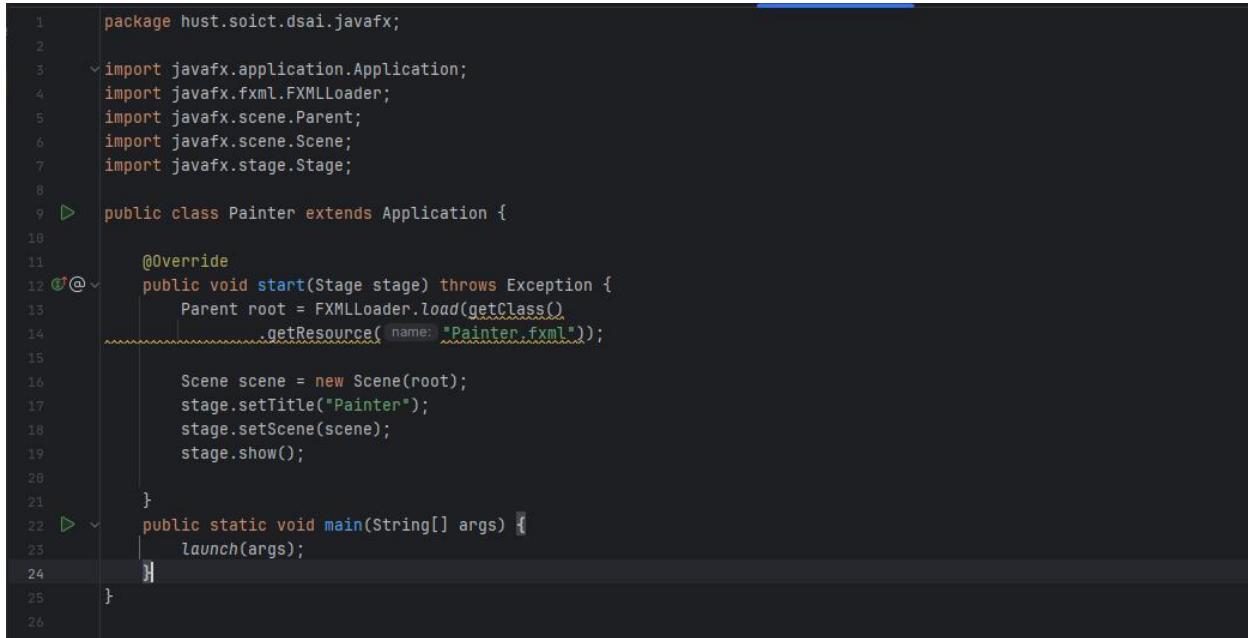


Figure 26.Demo View cart button

4. JavaFX API

4.1 Create class Painter



```
1 package hust.soict.dsai.javafx;
2
3 import javafx.application.Application;
4 import javafx.fxml.FXMLLoader;
5 import javafx.scene.Parent;
6 import javafx.scene.Scene;
7 import javafx.stage.Stage;
8
9 public class Painter extends Application {
10
11     @Override
12     public void start(Stage stage) throws Exception {
13         Parent root = FXMLLoader.load(getClass()
14             .getResource("Painter.fxml"));
15
16         Scene scene = new Scene(root);
17         stage.setTitle("Painter");
18         stage.setScene(scene);
19         stage.show();
20     }
21
22     public static void main(String[] args) {
23         launch(args);
24     }
25 }
```

Figure 27. Class Painter

4.2 Create Painter.fxml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <?import javafx.geometry.Insets?>
4  <?import javafx.scene.control.Button?>
5  <?import javafx.scene.control.RadioButton?>
6  <?import javafx.scene.control.TitledPane?>
7  <?import javafx.scene.control.ToggleGroup?>
8  <?import javafx.scene.layout.BorderPane?>
9  <?import javafx.scene.layout.Pane?>
10 <?import javafx.scene.layout.VBox?>
11 <?import javafx.scene.text.Font?>
12
13 <BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="480.0" prefWidth="640.0"
14     xmlns="http://javafx.com/javafx/23.0.1" xmlns:fx="http://javafx.com/fxml/1" fx:controller="hust_soict_dsa1 javafx.PainterController">
15     <padding>
16         <Insets bottom="8.0" left="8.0" right="8.0" top="8.0" />
17     </padding>
18     <left>
19         <VBox maxHeight="1.7976931348623157E308" spacing="8.0" BorderPane.alignment="CENTER">
20             <BorderPane.margin>
21                 <Insets right="8.0" />
22             </BorderPane.margin>
23             <children>
24                 <titledPane animated="false" maxWidth="1.7976931348623157E308" text="Tools">
25                     <font>
26                         <Font size="13.0" />
27                     </font>
28                     <content>
29                         <VBox>
30                             <children>
31                                 <RadioButton mnemonicParsing="false" onAction="#penButtonPressed" text="Pen">
32                                     <font>
33                                         <Font size="13.0" />
34                                     </font>
35                                     <toggleGroup>
36                                         <ToggleGroup fx:id="chose" />
37                                     </toggleGroup>
38                                 </RadioButton>
39                                 <RadioButton mnemonicParsing="false" onAction="#eraserButtonPressed" text="Eraser" toggleGroup="$chose">
40                                     <font>
41                                         <Font size="13.0" />
42                                     </font>
43                                 </RadioButton>
44                             </children>
45                         </VBox>
46                     </content>
47                 </titledPane>
48                 <Button maxWidth="1.7976931348623157E308" mnemonicParsing="false" onAction="#clearButtonPressed" text="Clear">
49                     <font>
50                         <Font size="13.0" />
51                     </font>
52                 </Button>
53             </children>
54         </VBox>
55     </left>
56     <center>
57         <Pane fx:id="drawingAreaPane" onMouseDragged="#drawingAreaMouseDragged" prefHeight="200.0" prefWidth="200.0" style="fx-background-color: white;" BorderPane.alignment="CENTER" />
58     </center>
59 </BorderPane>

```

Figure 28.Painter.fxml 1

```

31             <RadioButton mnemonicParsing="false" onAction="#penButtonPressed" text="Pen">
32                 <font>
33                     <Font size="13.0" />
34                 </font>
35                 <toggleGroup>
36                     <ToggleGroup fx:id="chose" />
37                 </toggleGroup>
38             </RadioButton>
39             <RadioButton mnemonicParsing="false" onAction="#eraserButtonPressed" text="Eraser" toggleGroup="$chose">
40                 <font>
41                     <Font size="13.0" />
42                 </font>
43             </RadioButton>
44         </children>
45     </VBox>
46     <content>
47     </titledPane>
48     <Button maxWidth="1.7976931348623157E308" mnemonicParsing="false" onAction="#clearButtonPressed" text="Clear">
49         <font>
50             <Font size="13.0" />
51         </font>
52     </Button>
53 </children>
54 </VBox>
55 </left>
56 <center>
57     <Pane fx:id="drawingAreaPane" onMouseDragged="#drawingAreaMouseDragged" prefHeight="200.0" prefWidth="200.0" style="fx-background-color: white;" BorderPane.alignment="CENTER" />
58 </center>
59 </BorderPane>

```

Figure 29.Painter.fxml 2

4.3 Create class PainterController

```
1 package hust.soict.dsai.javafx;
2 Runnable class
3     import javafx.event.ActionEvent;
4     import javafx.fxml.FXML;
5     import javafx.scene.control.ToggleGroup;
6     import javafx.scene.input.MouseEvent;
7     import javafx.scene.layout.Pane;
8     import javafx.scene.paint.Color;
9     import javafx.scene.paint.Paint;
10    import javafx.scene.shape.Circle;
11
12   public class PainterController {
13       private Paint penColor;  3 usages
14
15       @FXML
16   </>     private ToggleGroup chose;
17
18       @FXML
19   </>     private Pane drawingAreaPane;
20
21       @FXML
22   void clearButtonPressed(ActionEvent event) {
23       drawingAreaPane.getChildren().clear();
24   }
25
26
27   @
28   void drawingAreaMouseDragged(MouseEvent event) {
29       Circle newCircle = new Circle(event.getX(), event.getY(),  v2: 4, penColor);
30       drawingAreaPane.getChildren().add(newCircle);
31   }
32
33   @FXML
34   void eraserButtonPressed(ActionEvent event) {
35       penColor = Color.WHITE;
36   }
37
38   @FXML
39   void penButtonPressed(ActionEvent event) {
40       penColor = Color.BLACK;
41   }
42 }
```

Figure 30. Class PainterController

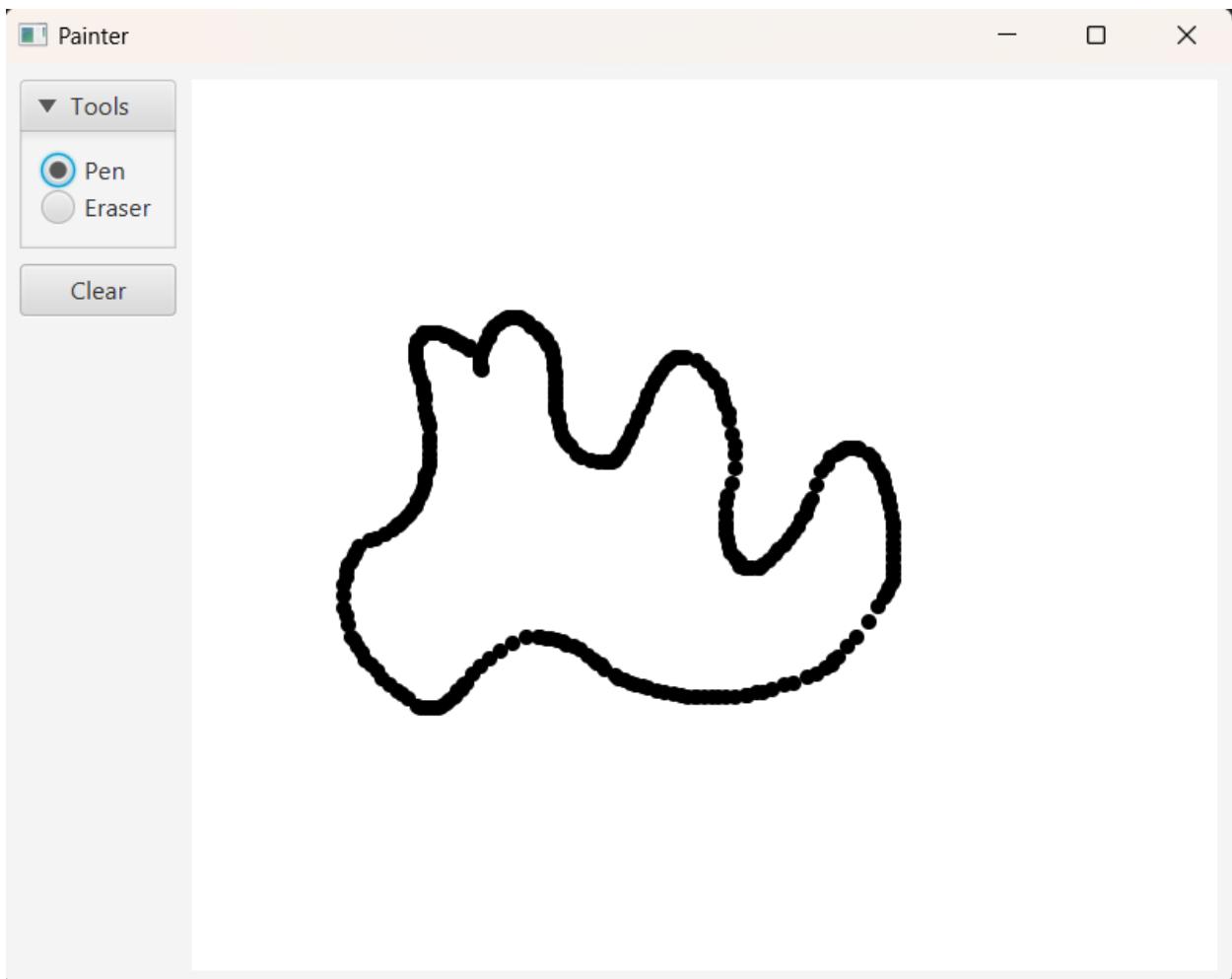


Figure 31. Use Pen

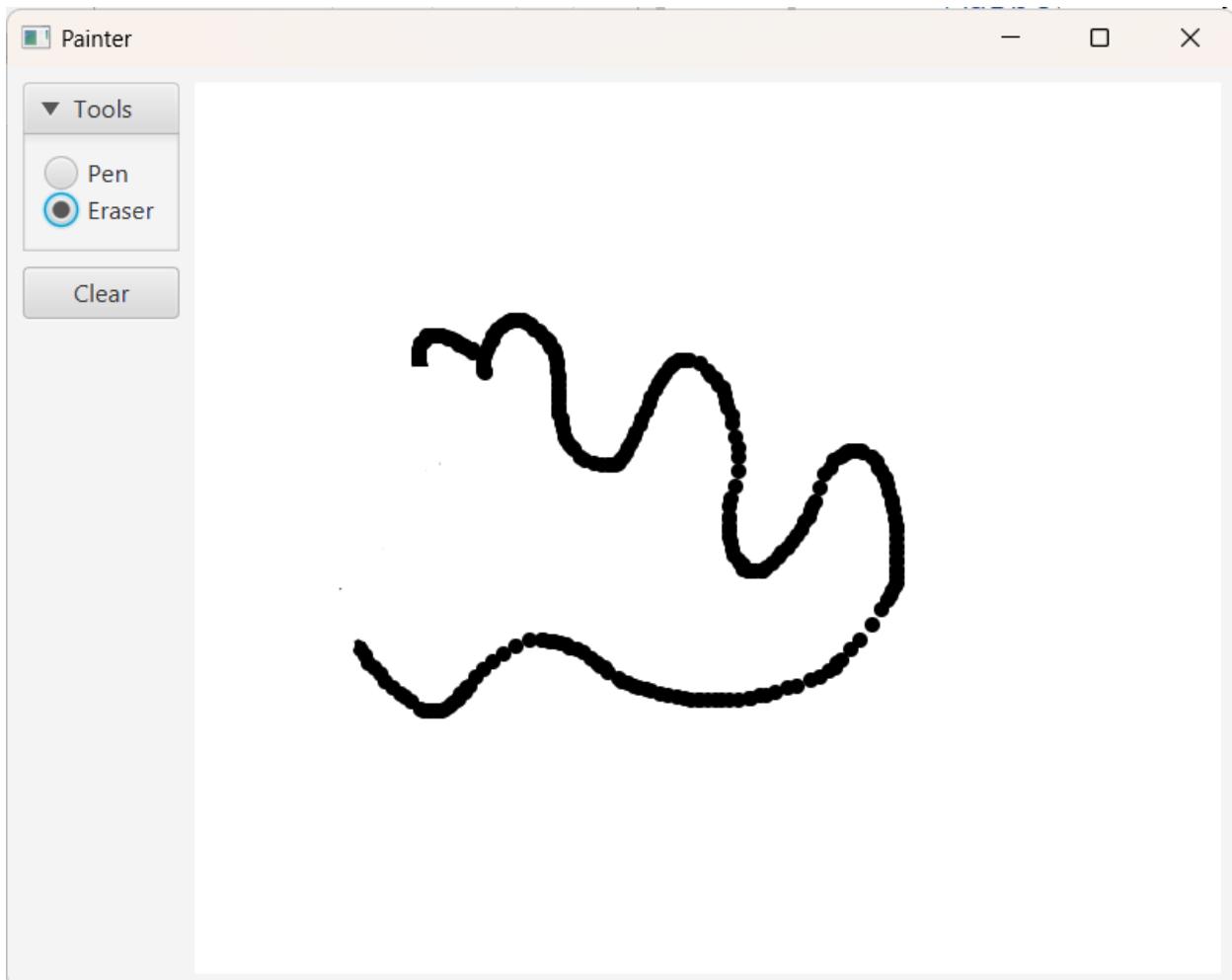


Figure 32. Use Eraser

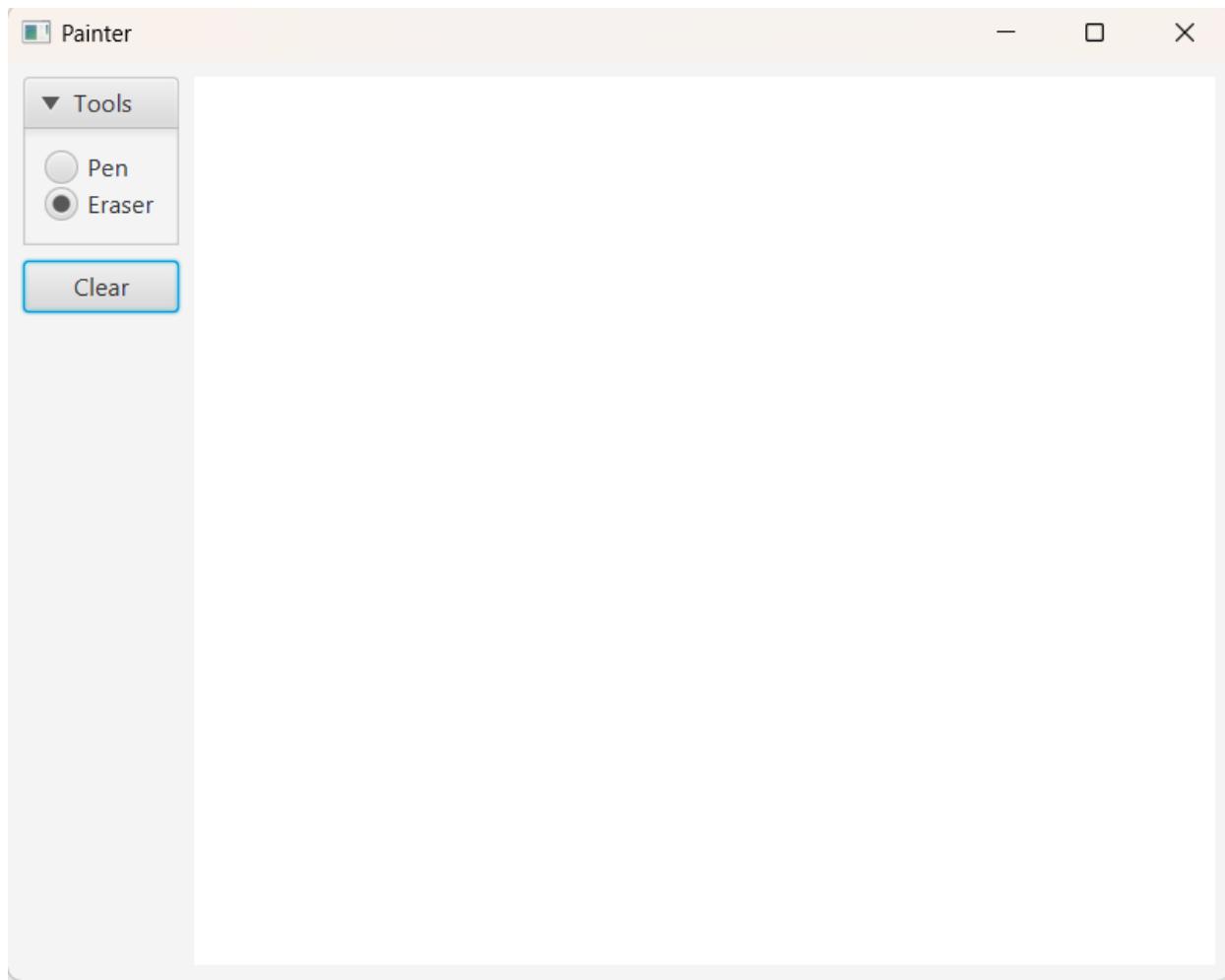
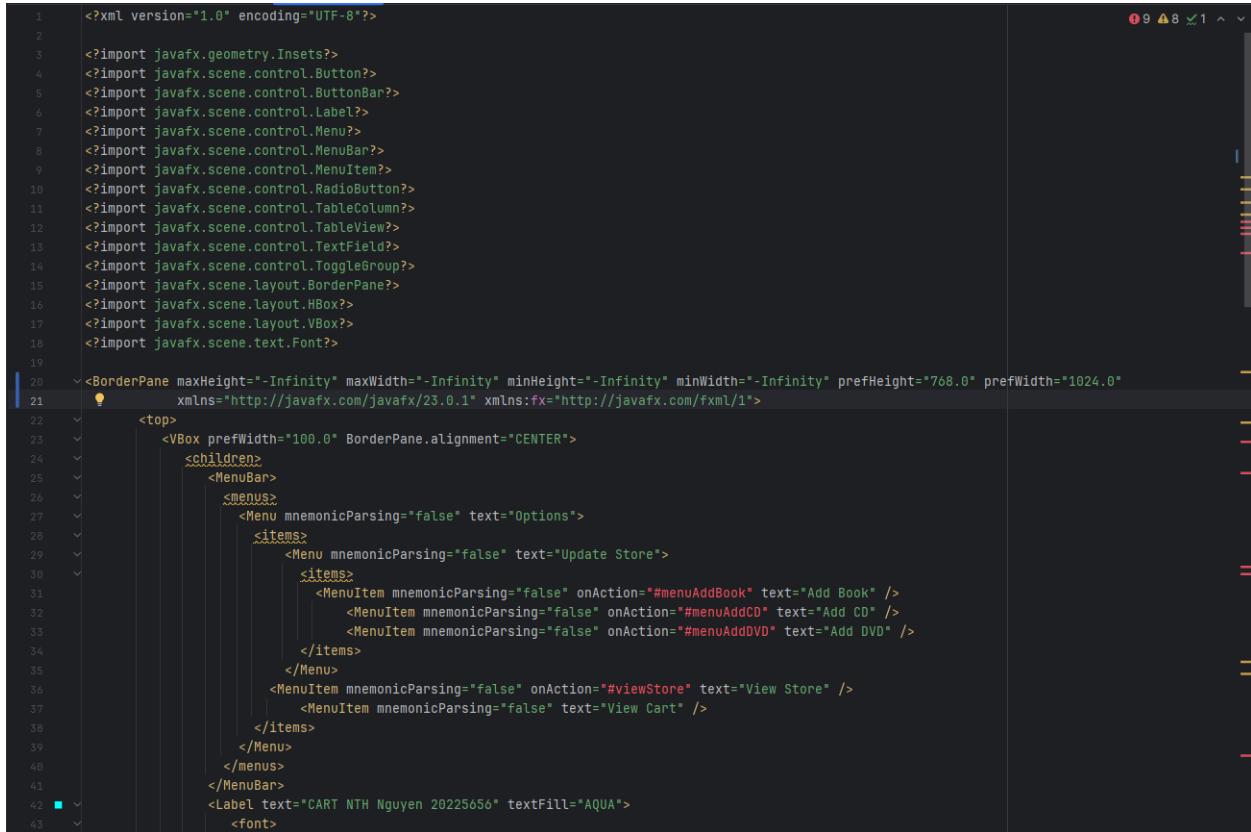


Figure 33. Clear button

5. View Cart Screen

5.1 Create cart.fxml



```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <?import javafx.geometry.Insets?>
4  <?import javafx.scene.control.Button?>
5  <?import javafx.scene.control.ButtonBar?>
6  <?import javafx.scene.control.Label?>
7  <?import javafx.scene.control.Menu?>
8  <?import javafx.scene.control.MenuBar?>
9  <?import javafx.scene.control.MenuItem?>
10 <?import javafx.scene.control.RadioButton?>
11 <?import javafx.scene.control.TableColumn?>
12 <?import javafx.scene.control.TableView?>
13 <?import javafx.scene.control.TextField?>
14 <?import javafx.scene.control.ToggleGroup?>
15 <?import javafx.scene.layout.BorderPane?>
16 <?import javafx.scene.layout.HBox?>
17 <?import javafx.scene.layout.VBox?>
18 <?import javafx.scene.text.Font?>
19
20 <BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="768.0" prefWidth="1024.0" xmlns="http://javafx.com/javafx/23.0.1" xmlns:fx="http://javafx.com/fxml/1">
21   <top>
22     <VBox prefWidth="100.0" BorderPane.alignment="CENTER">
23       <children>
24         <MenuBar>
25           <menus>
26             <menu mnemonicParsing="false" text="Options">
27               <items>
28                 <menu mnemonicParsing="false" text="Update Store">
29                   <items>
30                     <menuItem mnemonicParsing="false" onAction="#menuAddBook" text="Add Book" />
31                     <menuItem mnemonicParsing="false" onAction="#menuAddCD" text="Add CD" />
32                     <menuItem mnemonicParsing="false" onAction="#menuAddDVD" text="Add DVD" />
33                   </items>
34                 </menu>
35               <menuItem mnemonicParsing="false" onAction="#viewStore" text="View Store" />
36                 <menuItem mnemonicParsing="false" text="View Cart" />
37               </items>
38             </menu>
39           </menus>
40         </MenuBar>
41         <Label text="CART NTH Nguyen 20225656" textFill="AQUA">
42           <font>
43

```

Figure 34. Cart.fxml 1

```
44         <Font size="50.0" />
45     </font>
46     <padding>
47         <Insets left="10.0" />
48     </padding>
49     </Label>
50   </children>
51 </VBox>
52 </top>
53 <center>
54   <VBox prefHeight="200.0" prefWidth="100.0" BorderPane.alignment="CENTER">
55     <children>
56       <HBox alignment="CENTER_LEFT" prefWidth="200.0" spacing="10.0">
57         <opaqueInsets>
58             <Insets />
59         </opaqueInsets>
60         <padding>
61             <Insets bottom="10.0" top="10.0" />
62         </padding>
63       <children>
64         <Label text="Filter:>" />
65         <TextField fx:id="tfFilter" />
66         <RadioButton fx:id="radioBtnFilterId" mnemonicParsing="false" onAction="#setFilterByID" selected="true" text="By ID">
67             <toggleGroup>
68                 <ToggleGroup fx:id="filterCategory" />
69             </toggleGroup>
70         </RadioButton>
71         <RadioButton fx:id="radioBtnFilterTitle" mnemonicParsing="false" onAction="#setFilterByTitle" text="By Title" toggleGroup="$filterCategory" />
72     </children>
73   </HBox>
74 <TableView fx:id="tblMedia">
75   <columns>
76     <TableColumn fx:id="colMediaTitle" prefWidth="75.0" text="Title" />
77     <TableColumn fx:id="colMediaCategory" prefWidth="75.0" text="Category" />
78     <TableColumn fx:id="colMediaCost" prefWidth="75.0" text="Cost" />
79   </columns>
80   <columnResizePolicy>
81     <Tableview fx:constant="CONSTRAINED_RESIZE_POLICY" />
82   </columnResizePolicy>
```

Figure 35. Cart.fxml 2

```
83 <ButtonBar prefHeight="40.0" prefWidth="200.0">
84     <buttons>
85         <Button fx:id="btnPlay" mnemonicParsing="false" onAction="#btnPlayPressed" text="Play" />
86         <Button fx:id="btnRemove" mnemonicParsing="false" onAction="#btnRemovePressed" text="Remove" />
87     </buttons>
88 </ButtonBar>
89 </children>
90 <padding>
91     <Insets left="10.0" />
92 </padding>
93 </VBox>
94 </center>
95 <right>
96     <VBox alignment="TOP_CENTER" prefHeight="200.0" BorderPane.alignment="CENTER">
97         <padding>
98             <Insets top="50.0" />
99         </padding>
100        <children>
101            <HBox alignment="CENTER" spacing="10.0">
102                <children>
103                    <Label text="Total:>
104                        <font>
105                            <Font size="24.0" />
106                        </font>
107                    </Label>
108                    <Label fx:id="totalCost" text="0 $" textFill="AQUA">
109                        <font>
110                            <Font size="24.0" />
111                        </font>
112                    </Label>
113                </children>
114            </HBox>
115            <Button mnemonicParsing="false" onAction="#placeOrderPressed" style="-fx-background-color: red;" text="Place Order" textFill="WHITE">
116                <font>
117                    <Font size="24.0" />
118                </font>
119            </Button>
120        <children>
121    </VBox>
122 </right>
```

Figure 36. Cart.fxml 3

5.2 Create class CartScreen

```

1 package hust.soict.dsai.aims.screen;
2
3 > import ...
4
5
6 public class CartScreen extends JFrame { ± Hoàng Nguyễn
7     private Cart cart; 1 usage
8     private Store store; 1 usage
9
10
11     public CartScreen(Store store,Cart cart) { 2 usages ± Hoàng Nguyễn
12         super();
13         this.store = store;
14         this.cart = cart;
15         this.setSize( width: 1024, height: 768);
16         JFXPanel fxPanel = new JFXPanel();
17         this.add(fxPanel);
18
19         this.setTitle("Cart");
20         this.setVisible(true);
21         JFrame frame = this;
22
23         this.addWindowListener((WindowAdapter) windowClosing(e) → {
24             new StoreScreen(store, cart);
25             dispose(); // Giải phóng tài nguyên của sổ hiện tại
26         });
27
28
29
30
31
32
33 < > Platform.runLater(new Runnable{ ± Hoàng Nguyễn
34     @Override ± Hoàng Nguyễn
35     public void run() {
36         try{
37             FXMLLoader loader = new FXMLLoader(getClass()
38                 .getResource( name: "cart.fxml"));
39             CartScreenController controller =
40                 new CartScreenController(store, cart,frame);
41             loader.setController(controller);
42             Parent root = loader.load();
43             fxPanel.setScene(new Scene(root));
44         }catch (IOException e){
45             e.printStackTrace();
46         }
47     }
48
49     });
50
51
52
53
54
55
56
57     });
58
59 }

```

Figure 37. CartScreen class

5.3 Create class CartScreenController

```
1 package hust.soict.dsai.aims.screen;
2
3 > import ...;
4
5
6 public class CartScreenController { ± Hoàng Nguyễn
7
8     private Store store; 5 usages
9     private Cart cart; 14 usages
10    private JFrame stage; 5 usages
11    private FilteredList<Media> filteredCart; 4 usages
12    private boolean filterByID = true; 3 usages
13
14    @FXML 3 usages
15    private Label totalCost;
16
17    @FXML no usages
18    private ToggleGroup filterCategory;
19
20    @FXML 4 usages
21    private Button btnPlay;
22
23    @FXML 3 usages
24    private Button btnRemove;
25
26    @FXML 1 usage
27    private TextField tfFilter;
28
29    @FXML no usages
30    private RadioButton radioBtnFilterId;
31
32    @FXML no usages
33    private RadioButton radioBtnFilterTitle;
34
35    @FXML 4 usages
36    private TableView<Media> tblMedia;
37
38    @FXML 1 usage
39    private TableColumn<Media, String> colMediaTitle;
40
41    @FXML 1 usage
42    private TableColumn<Media, String> colMediacategory;
43
44    @FXML 1 usage
45    private TableColumn<Media, Float> colMediaCost;
```

Figure 38. CartScreenController 1

```

63     public CartScreenController(Store store, Cart cart, JFrame stage) { 1 usage ▲ Hoàng Nguyễn
64         super();
65         this.cart = cart;
66         this.store = store;
67         this.stage = stage;
68     }
69
70     @FXML  no usages ▲ Hoàng Nguyễn
71     private void initialize() {
72         filteredCart = new FilteredList<Media>(this.cart.getItemsOrdered(), s -> true);
73
74         colMediaTitle.setCellValueFactory(
75             new PropertyValueFactory<Media, String>( s: "Title"));
76         colMediacategory.setCellValueFactory(
77             new PropertyValueFactory<Media, String>( s: "Category"));
78         colMediaCost.setCellValueFactory(
79             new PropertyValueFactory<Media, Float>( s: "Cost"));
80         tblMedia.setItems(this.cart.getItemsOrdered());
81
82         btnPlay.setVisible(false);
83         btnRemove.setVisible(false);
84
85         totalCost.setText(String.valueOf(this.cart.totalCost()));
86
87         tblMedia.getSelectionModel().selectedItemProperty().addListener(new ChangeListener<Media>() { ▲ Hoàng Nguyễn
88
89             @Override ▲ Hoàng Nguyễn
90             public void changed(ObservableValue<? extends Media> observable, Media oldValue, Media newValue) {
91                 if (newValue != null) {
92                     updateButtonBar(newValue);
93                 }
94             }
95         });
96
97         tfFilter.textProperty().addListener(new ChangeListener<String>() { ▲ Hoàng Nguyễn
98             @Override ▲ Hoàng Nguyễn
99             public void changed(ObservableValue<? extends String> observable, String oldValue, String newValue) {
100                 showFilteredMedia(newValue);
101             }
102         });
103     }
104 }
105

```

Figure 39. CartScreenController 2

```

105
106 @FXML no usages ▲ Hoàng Nguyễn
107 < private void btnRemovePressed(ActionEvent event) throws NonExistingItemException {
108     // Lấy đối tượng media được chọn
109     Media media = tblMedia.getSelectionModel().getSelectedItem();
110
111     // Kiểm tra xem có item nào được chọn hay không
112     if (media == null) {
113         // Nếu không có item nào được chọn, hiển thị thông báo lỗi
114         Alert alert = new Alert(Alert.AlertType.WARNING);
115         alert.setTitle("Selection Error");
116         alert.setHeaderText("No item selected");
117         alert.setContentText("Please select an item to remove.");
118         alert.showAndWait();
119         return; // Dừng phương thức nếu không có item được chọn
120     }
121
122     // Thử xóa media khỏi giỏ hàng
123     try {
124         cart.removeMedia(media);
125         // Cập nhật lại tổng chi phí sau khi xóa
126         totalCost.setText(String.valueOf(this.cart.totalCost()));
127         // Hiển thị thông báo thành công
128         Alert alert = new Alert(Alert.AlertType.INFORMATION);
129         alert.setTitle("Success");
130         alert.setHeaderText("Item Removed");
131         alert.setContentText(media.getTitle() + " has been removed from the cart.");
132         alert.showAndWait();
133     } catch (NonExistingItemException e) {
134         // Nếu media không tồn tại trong giỏ hàng, ném ngoại lệ
135         throw new NonExistingItemException("Item does not exist in the cart.");
136     }
137 }
138
139
140 @FXML no usages ▲ Hoàng Nguyễn
141 > private void setFilterByID() { this.filterByID = true; }
142
143 @FXML no usages ▲ Hoàng Nguyễn
144 > private void setFilterByTitle() { this.filterByID = false; }
145
146 > void updateButtonBar(Media media) { 1 usage ▲ Hoàng Nguyễn
147     if (media == null) {
148         btnPlay.setVisible(false);
149     }

```

Figure 40. CartScreenController 3

```

150     void updateButtonBar(Media media) { 1 usage ▲ Hoàng Nguyễn
151         if (media == null) {
152             btnPlay.setVisible(false);
153             btnRemove.setVisible(false);
154         } else {
155             btnRemove.setVisible(true);
156             if (media instanceof Playable) {
157                 btnPlay.setVisible(true);
158             } else {
159                 btnPlay.setVisible(false);
160             }
161         }
162     }
163
164     private void showFilteredMedia(String filter) { 1 usage ▲ Hoàng Nguyễn
165         if (filter == null || filter.length() == 0) {
166             filteredCart.setPredicate(s -> true);
167         } else {
168             // Nếu bộ lọc là ID, chỉ hiển thị sản phẩm có ID khớp.
169             if (filterByID) {
170                 try {
171                     filteredCart.setPredicate(s -> s.getId() == Integer.parseInt(filter));
172                 } catch (NumberFormatException e) {
173                 }
174             } else // Nếu bộ lọc là tên, hiển thị sản phẩm có tên chứa từ khóa (bỏ qua chữ hoa/chữ thường).
175             {
176                 filteredCart.setPredicate(s -> s.getTitle().toLowerCase().contains(filter));
177             }
178         }
179     }
180
181     @FXML no usages ▲ Hoàng Nguyễn
182     private void btnPlayPressed(ActionEvent event) throws PlayerException {
183         // Lấy đối tượng media được chọn
184         Media media = this.tblMedia.getSelectionModel().getSelectedItem();
185
186         // Kiểm tra xem media có phải là đối tượng Playable không
187         if (media instanceof Playable) {
188             try {
189                 // Ép kiểu media thành Playable và gọi phương thức play
190                 ((Playable) media).play();
191             } catch (Exception e) {
192                 // Xử lý lỗi phát video hoặc âm thanh (nếu có)
193                 throw new PlayerException("Error while trying to play the media: " + e.getMessage(), e);
194             }
195         }
196     }

```

Figure 41. CartScreenController 4

```

193             }
194         } else {
195             // Nếu media không phải là Playable, hiển thị thông báo lỗi
196             throw new PlayerException("Selected media is not playable.");
197         }
198     }
199 }
200
201
202 @FXML no usages ▾ Hoàng Nguyễn
203 private void placeOrderPressed(ActionEvent event) {
204     // Kiểm tra nếu giỏ hàng trống
205     if (cart.getItemsOrdered().isEmpty()) {
206         // Hiển thị thông báo lỗi
207         Alert emptyCartAlert = new Alert(Alert.AlertType.ERROR);
208         emptyCartAlert.setTitle("Empty Cart");
209         emptyCartAlert.setHeaderText(null);
210         emptyCartAlert.setContentText("Your cart is empty. Please add items before placing an order.");
211         emptyCartAlert.showAndWait();
212         return;
213     }
214
215     // Tính tổng chi phí
216     float totalCost1 = cart.totalCost();
217
218     // Hiển thị thông báo thành công
219     Alert successAlert = new Alert(Alert.AlertType.INFORMATION);
220     successAlert.setTitle("Order Placed");
221     successAlert.setHeaderText(null);
222     successAlert.setContentText("Your order has been placed successfully!\nTotal cost: " + totalCost1);
223     successAlert.showAndWait();
224     // Xóa giỏ hàng
225     cart.clear();
226     totalCost.setText(String.valueOf(this.cart.totalCost()));
227 }
228
229
230 @FXML no usages ▾ Hoàng Nguyễn
231 void menuAddBook(ActionEvent event) {
232     new AddBookToStoreScreen(store, cart); // Hiển thị giao diện thêm Book
233     stage.hide();
234 }
235

```

Figure 42. CartScreenController 5

```
230 @FXML no usages ▲ Hoàng Nguyễn
231 void menuAddBook(ActionEvent event) {
232     new AddBookToStoreScreen(store, cart); // Hiển thị giao diện thêm Book
233     stage.hide();
234 }
235
236 @FXML no usages ▲ Hoàng Nguyễn
237 void menuAddCD(ActionEvent event) {
238     new AddCDToStoreScreen(store, cart);
239     stage.hide();
240 }
241
242 @FXML no usages ▲ Hoàng Nguyễn
243 void menuAddDVD(ActionEvent event) {
244     new AddDVDToStoreScreen(store, cart);
245     stage.hide();
246 }
247
248 @FXML no usages ▲ Hoàng Nguyễn
249 void viewStore(ActionEvent event) {
250     new StoreScreen(store, cart);
251     stage.hide();
252 }
253
254 }
```

Figure 43. CartScreenController 6

5.4 Demo



Figure 44.Demo CartScreen

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

6.1 Edit class CartScreenController

```

@FXML no usages ± Hoàng Nguyễn
private void initialize() {
    filteredCart = new FilteredList<Media>(this.cart.getItemsOrdered(), s -> true);

    colMediaTitle.setCellValueFactory(
        new PropertyValueFactory<Media, String>(s: "Title"));
    colMediaCategory.setCellValueFactory(
        new PropertyValueFactory<Media, String>(s: "Category"));
    colMediaCost.setCellValueFactory(
        new PropertyValueFactory<Media, Float>(s: "Cost"));
    tblMedia.setItems(this.cart.getItemsOrdered());

    btnPlay.setVisible(false);
    btnRemove.setVisible(false);

    totalCost.setText(String.valueOf(this.cart.totalCost()));

    tblMedia.getSelectionModel().selectedItemProperty().addListener(new ChangeListener<Media>() { ± Hoàng Nguyễn
        @Override ± Hoàng Nguyễn
        public void changed(ObservableValue<? extends Media> observable, Media oldValue, Media newValue) {
            if (newValue != null) {
                updateButtonBar(newValue);
            }
        }
    });
}

tfFilter.textProperty().addListener(new ChangeListener<String>() { ± Hoàng Nguyễn
    @Override ± Hoàng Nguyễn
    public void changed(ObservableValue<? extends String> observable, String oldValue, String newValue) {
        showFilteredMedia(newValue);
    }
});
}

```

Figure 45. CartScreenController 1

```

@FXML no usages ± Hoàng Nguyễn
private void btnPlayPressed(ActionEvent event) throws PlayerException {
    // Lấy đối tượng media được chọn
    Media media = this.tblMedia.getSelectionModel().getSelectedItem();

    // Kiểm tra xem media có phải là đối tượng Playable không
    if (media instanceof Playable) {
        try {
            // Ép kiểu media thành Playable và gọi phương thức play
            ((Playable) media).play();
        } catch (Exception e) {
            // Xử lý lỗi phát video hoặc âm thanh (nếu có)
            throw new PlayerException("Error while trying to play the media: " + e.getMessage(), e);
        }
    } else {
        // Nếu media không phải là Playable, hiển thị thông báo lỗi
        throw new PlayerException("Selected media is not playable.");
    }
}

```

Figure 46. CartScreenController 2

6.2 Demo

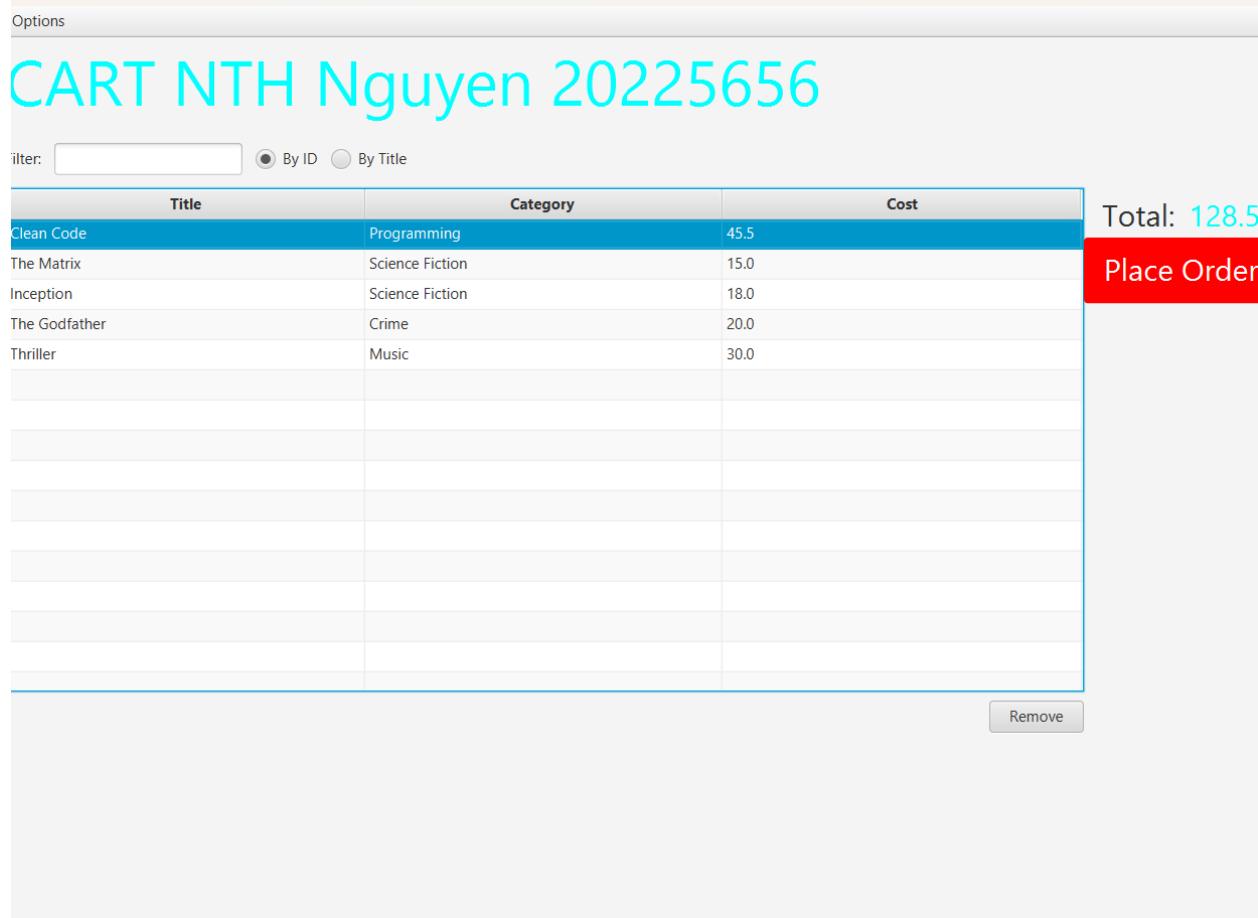


Figure 47.Demo media unplayable

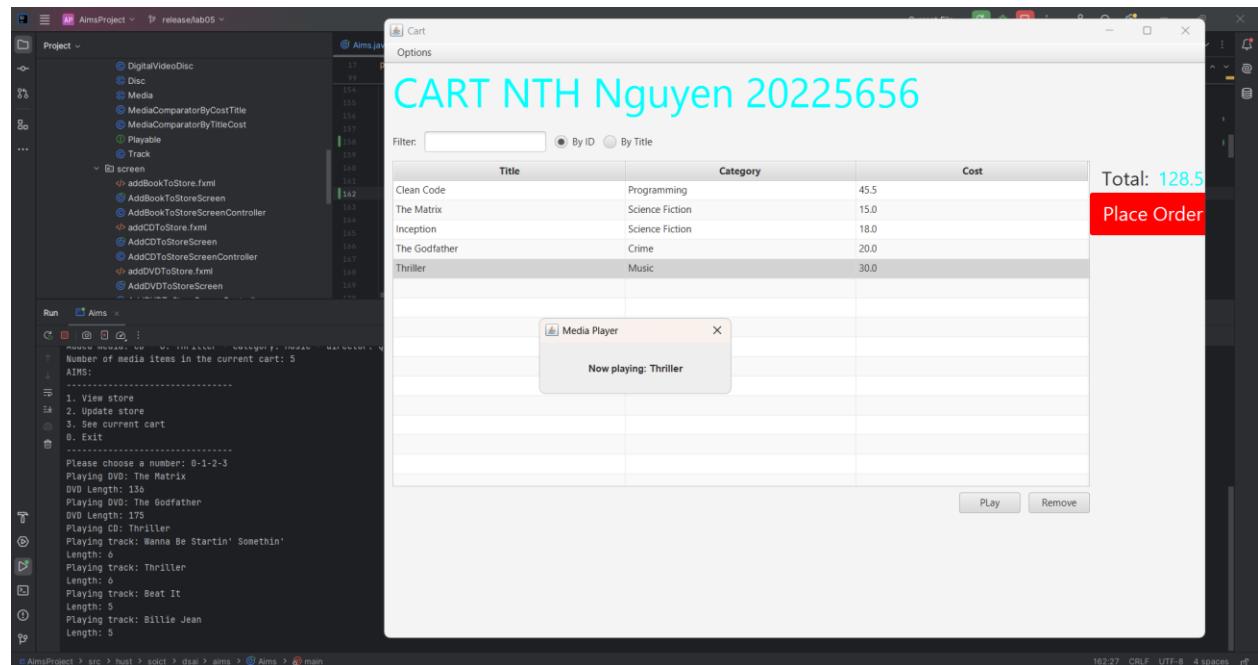


Figure 48.Demo media playable

7. Deleting a media

7.1 Code

```

@FXML no usages + Hoàng Nguyễn
private void btnRemovePressed(ActionEvent event) throws NonExistingItemException {
    // Lấy đối tượng media được chọn
    Media media = tblMedia.getSelectionModel().getSelectedItem();

    // Kiểm tra xem có item nào được chọn hay không
    if (media == null) {
        // Nếu không có item nào được chọn, hiển thị thông báo lỗi
        Alert alert = new Alert(Alert.AlertType.WARNING);
        alert.setTitle("Selection Error");
        alert.setHeaderText("No item selected");
        alert.setContentText("Please select an item to remove.");
        alert.showAndWait();
        return; // Dừng phương thức nếu không có item được chọn
    }

    // Thử xóa media khỏi giỏ hàng
    try {
        cart.removeMedia(media);
        // Cập nhật lại tổng chi phí sau khi xóa
        totalCost.setText(String.valueOf(this.cart.totalCost()));
        // Hiển thị thông báo thành công
        Alert alert = new Alert(Alert.AlertType.INFORMATION);
        alert.setTitle("Success");
        alert.setHeaderText("Item Removed");
        alert.setContentText(media.getTitle() + " has been removed from the cart.");
        alert.showAndWait();
    } catch (NonExistingItemException e) {
        // Nếu media không tồn tại trong giỏ hàng, ném ngoại lệ
        throw new NonExistingItemException("Item does not exist in the cart.");
    }
}

```

Figure 49. btnRemovePressed Method

7.2 Demo

The screenshot shows a Windows application window titled "CART NTH Nguyen 20225656". The window has a standard title bar with icons for minimize, maximize, and close. Below the title bar is a menu bar with "Cart" and "Options". The main area contains a grid table with the following data:

Title	Category	Cost
Clean Code	Programming	45.5
The Matrix	Science Fiction	15.0
Inception	Science Fiction	18.0
The Godfather	Crime	20.0
Thriller	Music	30.0

Below the table is a large empty space. On the right side, there is a red button with the text "Place Order" and above it, the text "Total: 128.5". At the bottom right of the main area are two buttons: "Play" and "Remove".

Figure 50.button Remove 1

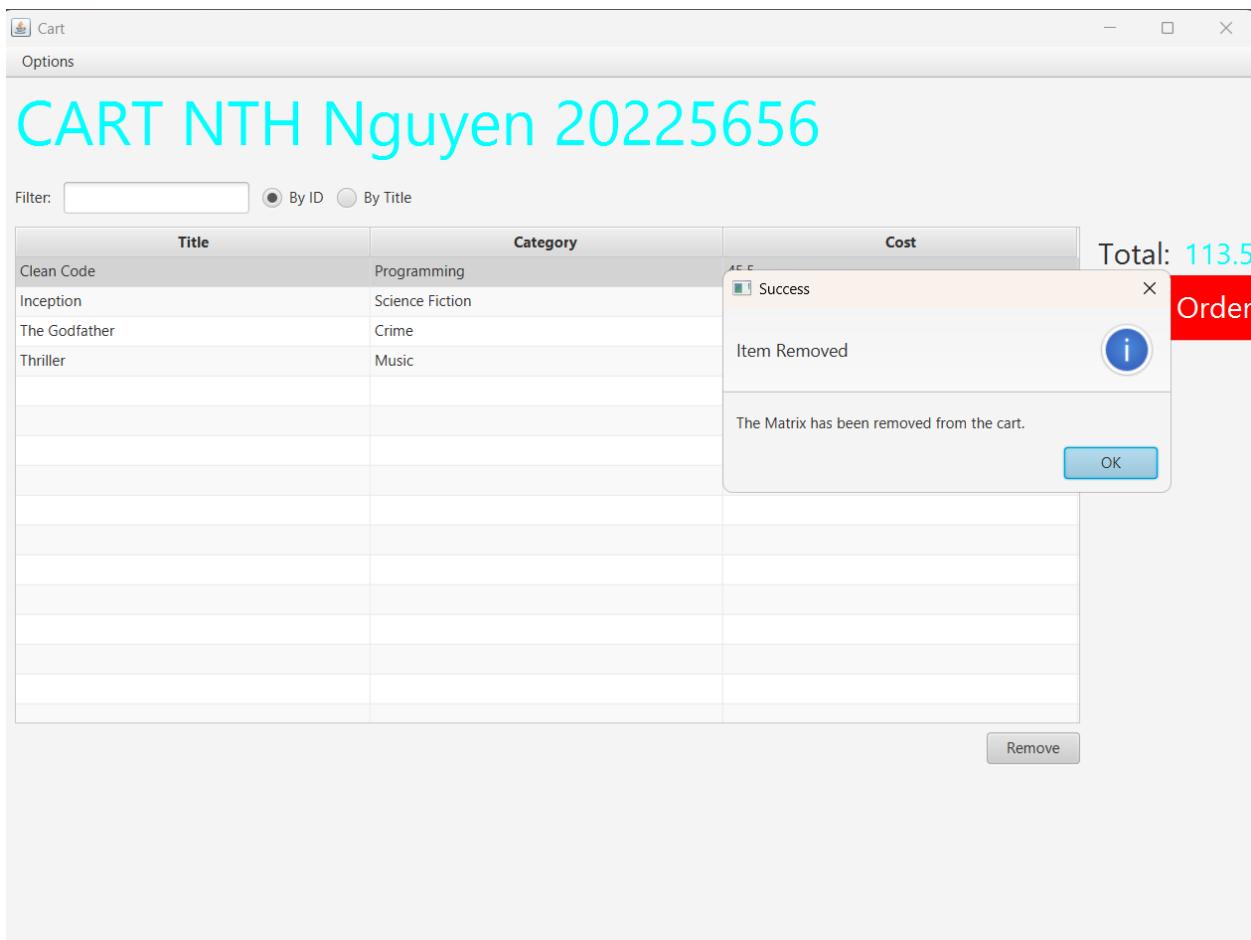


Figure 51.button Remove 2

8. Complete the Aims GUI application



Figure 52.Store before add book

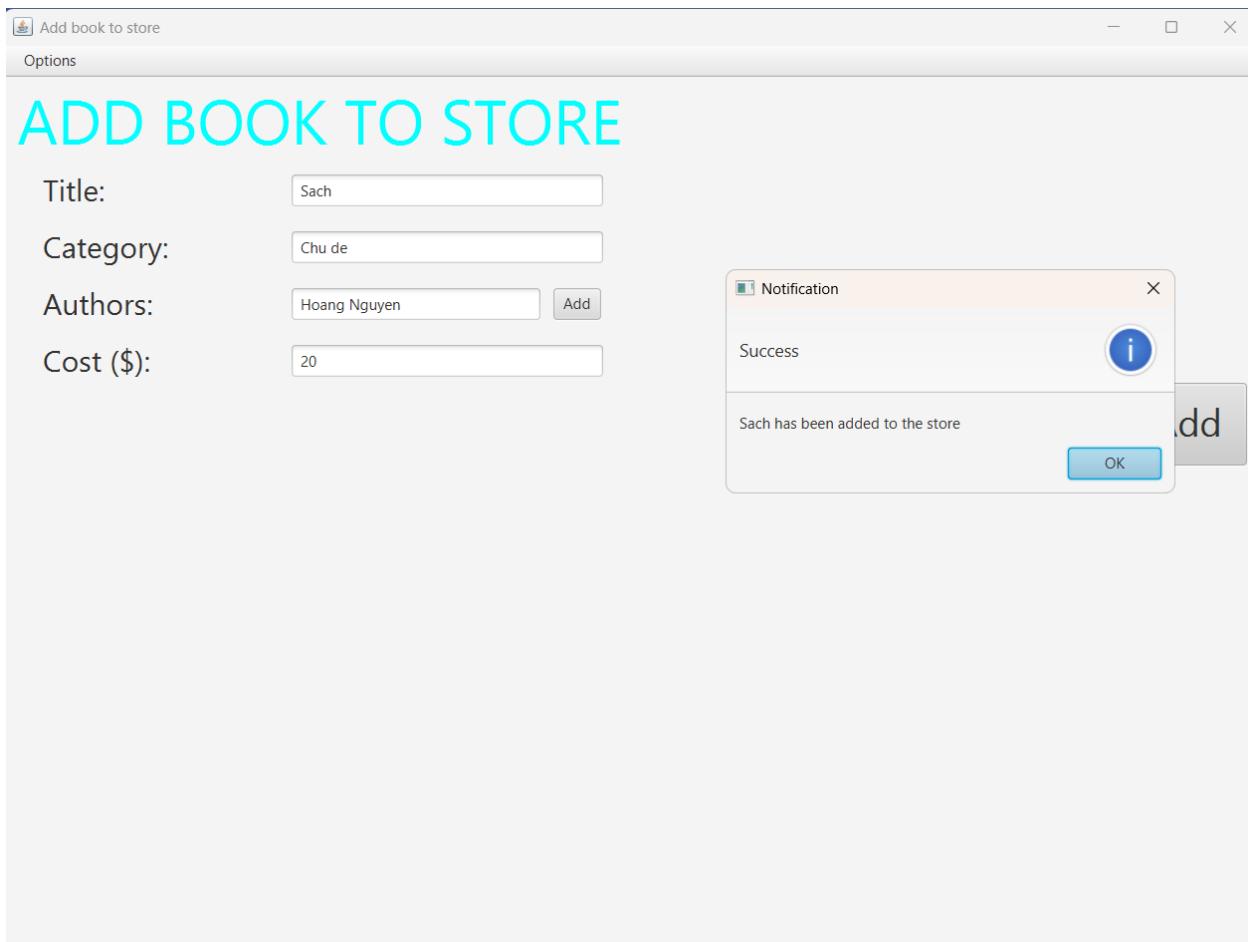


Figure 53.Add book



Figure 54. Store after add book

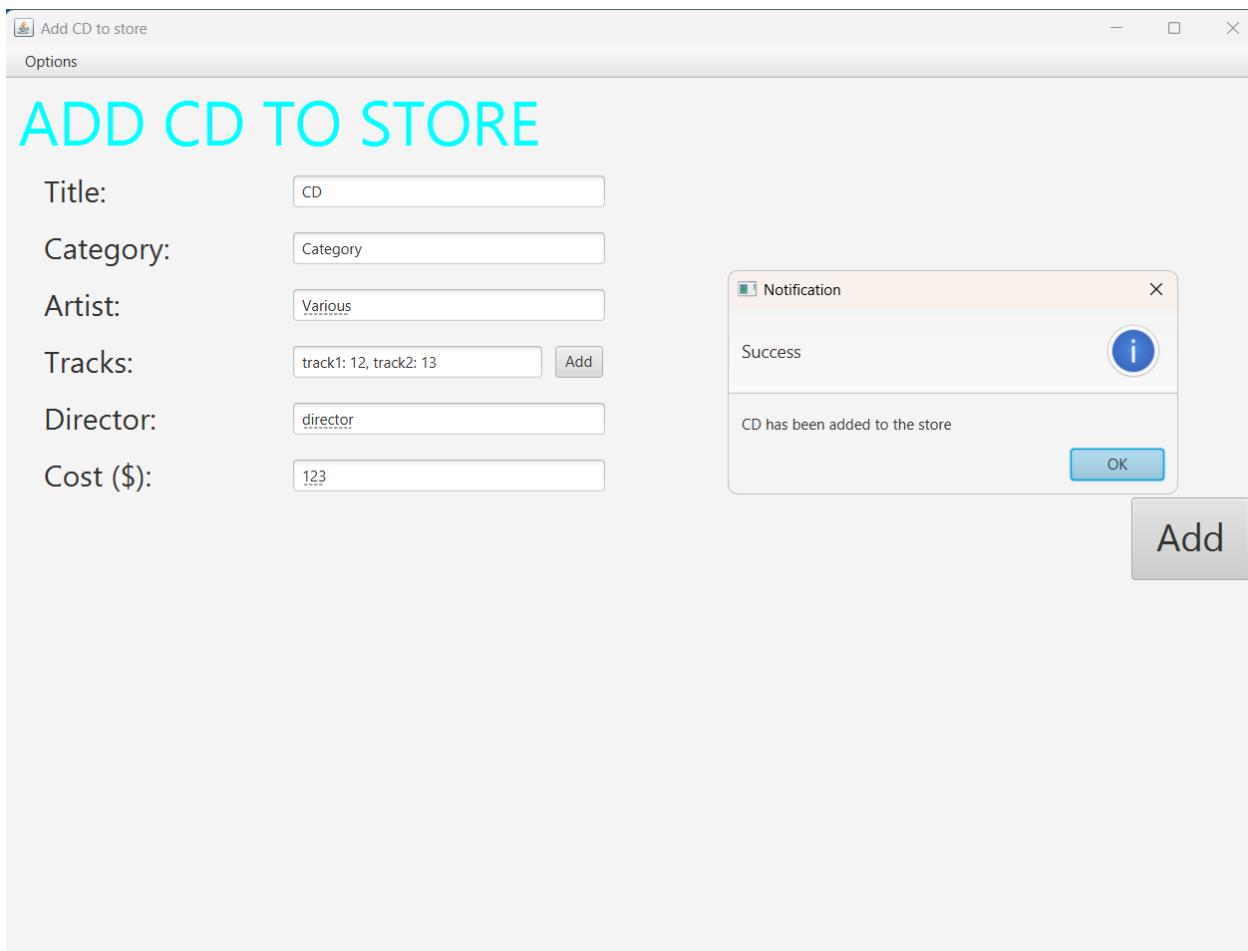


Figure 55.Add CD



Figure 56. Store after add CD

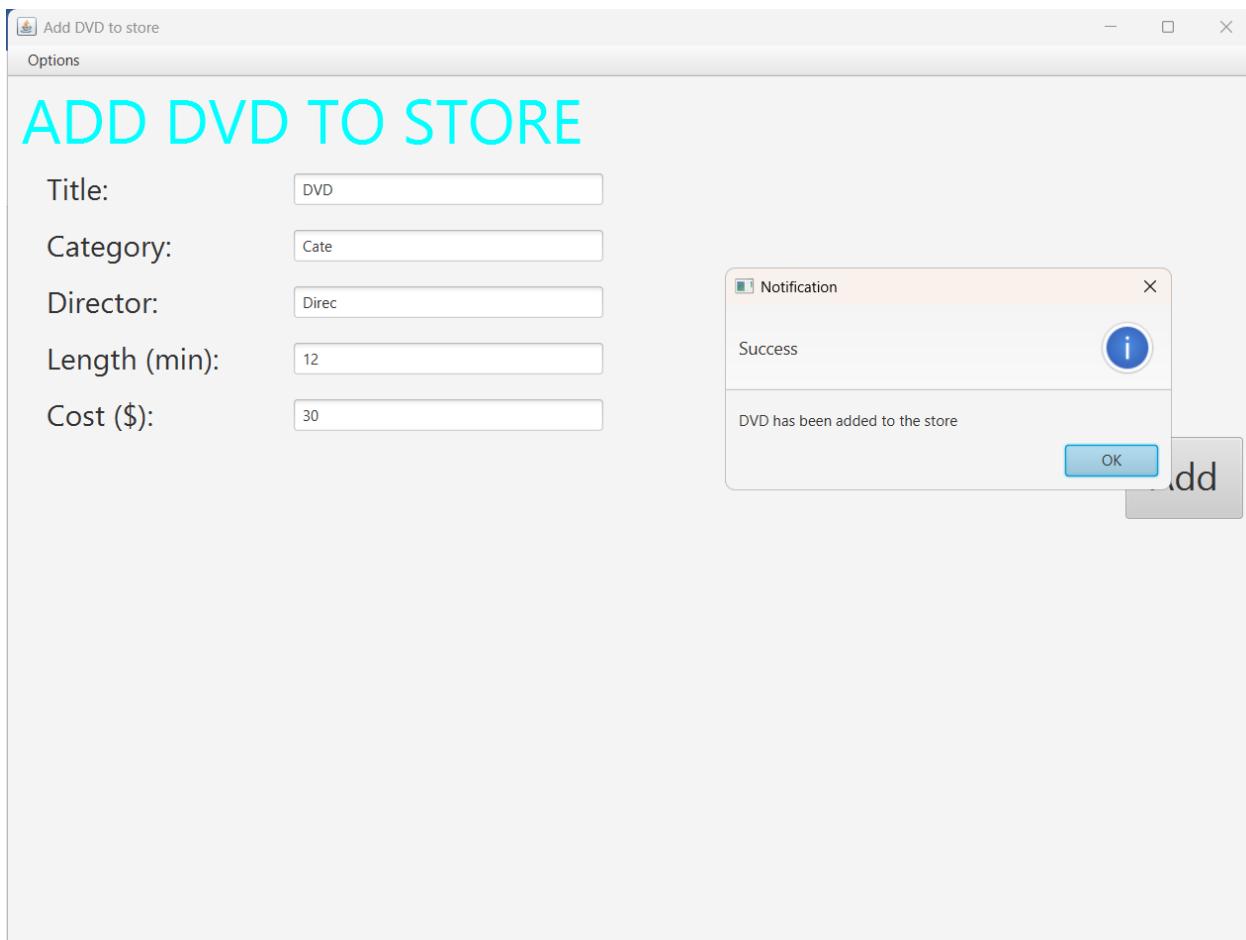


Figure 57.Add DVDFigure 58.Store after add DVD

AIMS:Nguyen Trinh Hoang Nguyen 2022565		
The Matrix 15.0 \$ Add to cart Play	Inception 18.0 \$ Add to cart Play	The Godfather 20.0 \$ Add to cart Play
Pulp Fiction 16.0 \$ Add to cart Play	The Lord of the Rings: The Fellows... 25.0 \$ Add to cart Play	Thriller 30.0 \$ Add to cart Play
Back in Black 25.0 \$ Add to cart Play	Abbey Road 35.0 \$ Add to cart Play	The Dark Side of the Moon 40.0 \$ Add to cart Play
Rumours 28.0 \$ Add to cart Play	Sach 20.0 \$ Add to cart	CD 123.0 \$ Add to cart Play
DVD 30.0 \$ Add to cart Play		

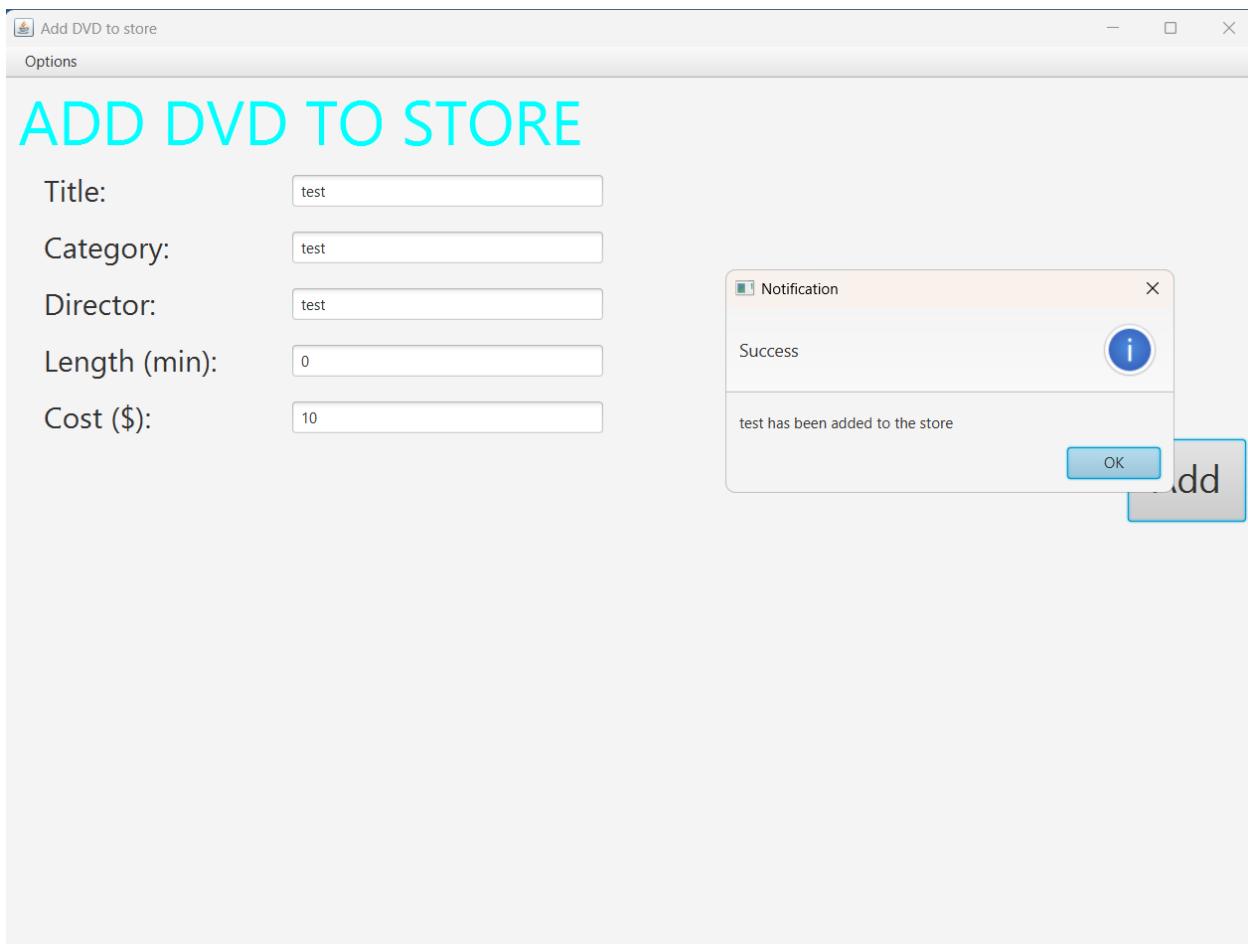


Figure 59. Test exception with DVD Length = 0

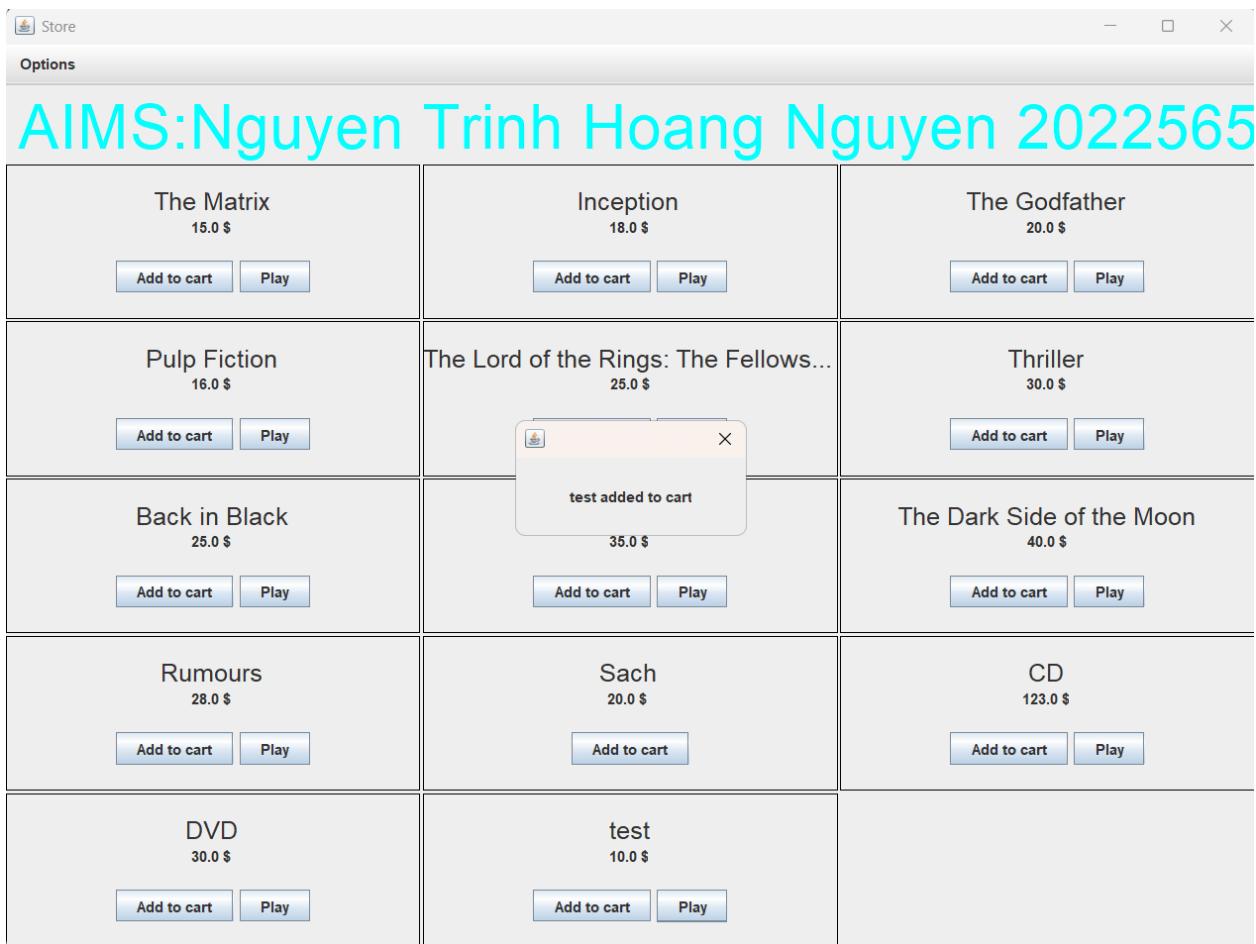


Figure 60.Add test to Cart

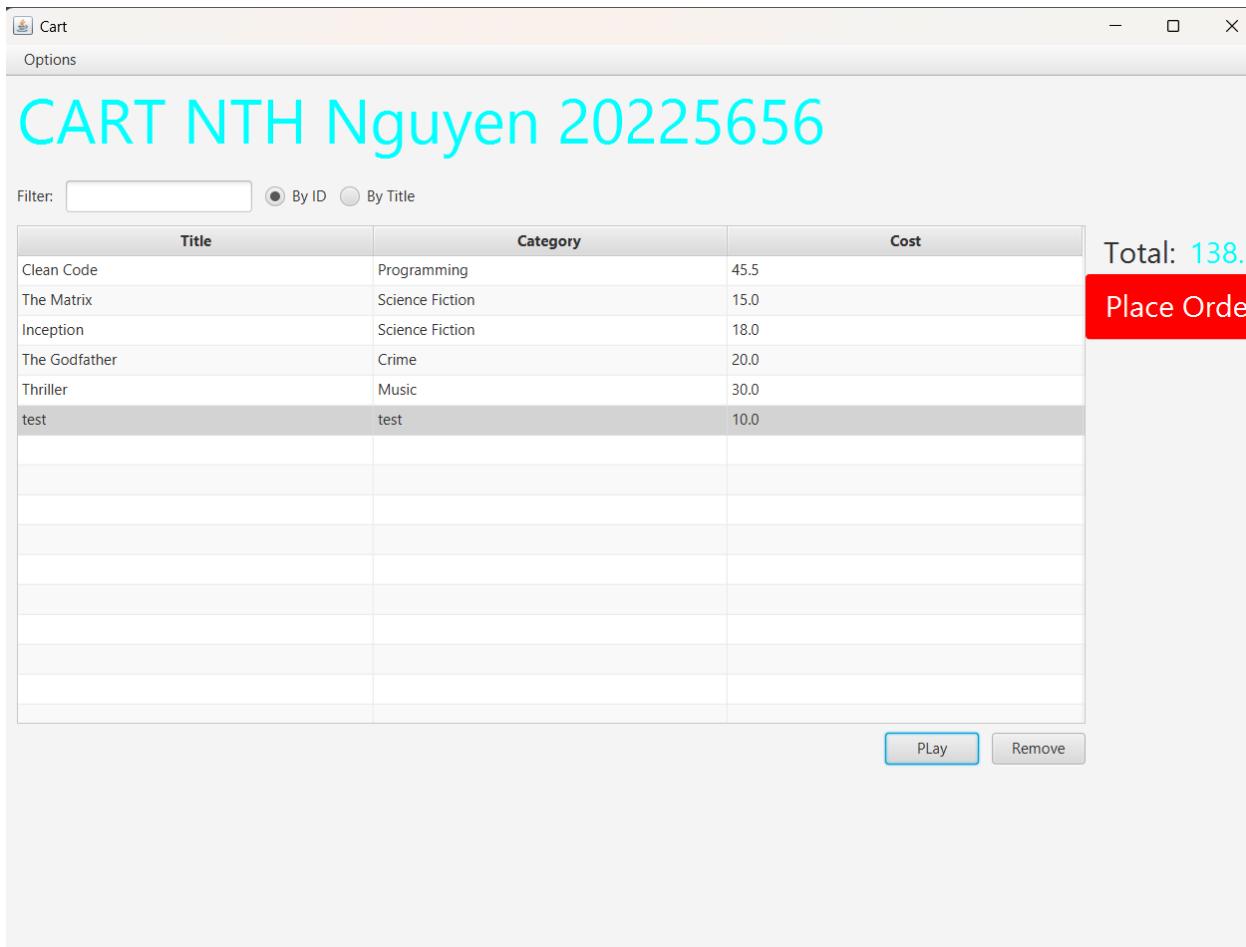


Figure 61.play test

```
... 47 more
Caused by: hust.soict.dsai.aims.exception.PlayerException Create breakpoint : Error while trying to play the media: DVD Length is zero
    at AimsProject/hust.soict.dsai.aims.screen.CartScreenController.btnPlayPressed(CartScreenController.java:193)
    at java.base/jdk.internal.reflect.DirectMethodHandleAccessor.invoke(DirectMethodHandleAccessor.java:103)
    ... 56 more
Caused by: hust.soict.dsai.aims.exception.PlayerException Create breakpoint : DVD Length is zero
    at AimsProject/hust.soict.dsai.aims.media.DigitalVideoDisc.play(DigitalVideoDisc.java:52)
    at AimsProject/hust.soict.dsai.aims.screen.CartScreenController.btnPlayPressed(CartScreenController.java:190)
    ... 57 more
```

Figure 62.Exception

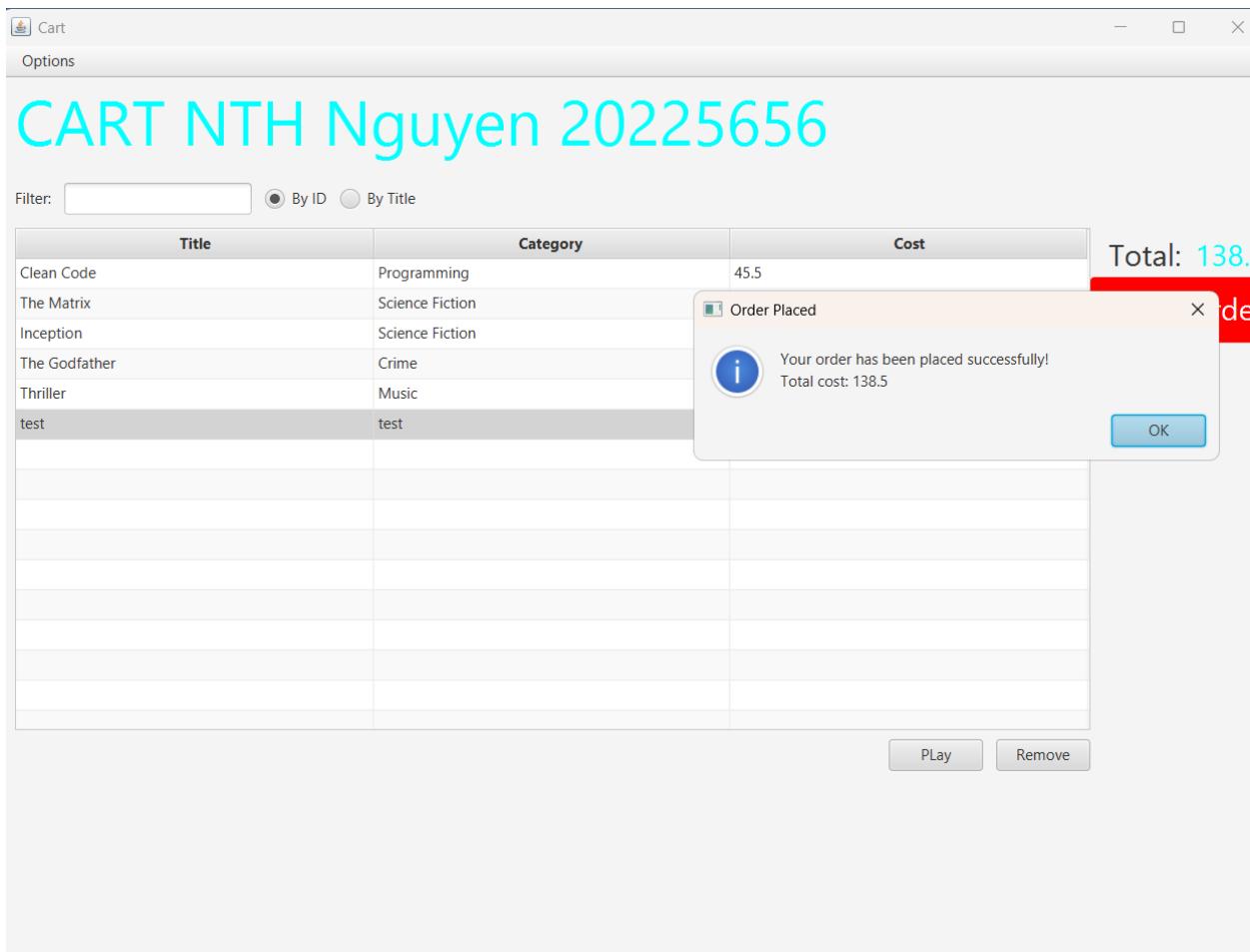


Figure 63. Place Order

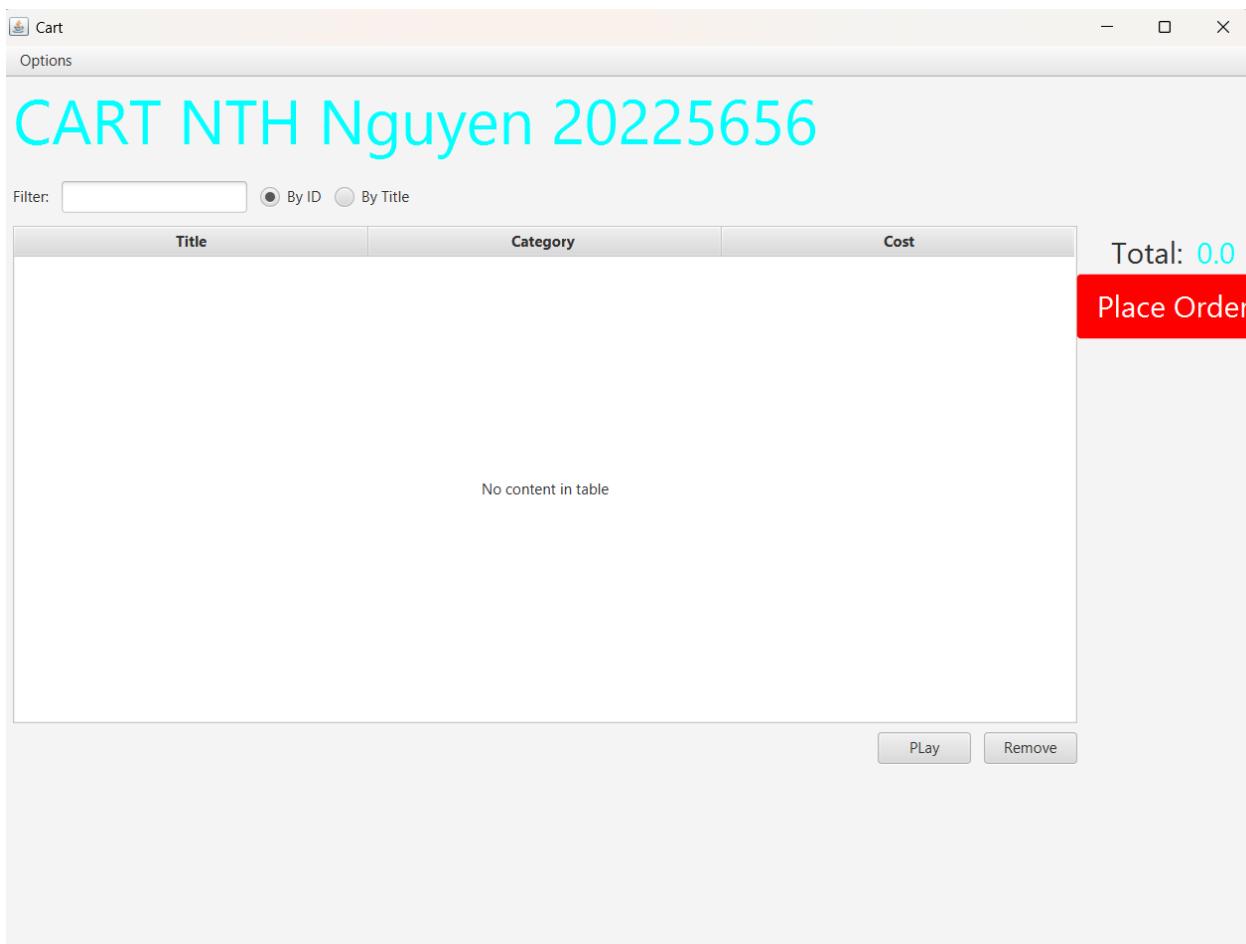


Figure 64.After Place Order

9. equals() method in Media class

```
@Override ▲ Hoàng Nguyễn
public boolean equals(Object object){
    if (object instanceof Media) {
        try {
            Media that = (Media) object;
            return this.title.equalsIgnoreCase(that.getTitle());
        } catch (NullPointerException | ClassCastException e1) {
            return false;
        }
    } else {
        return false;
    }
}
```

Figure 65.equals() method

10. Use case Diagram

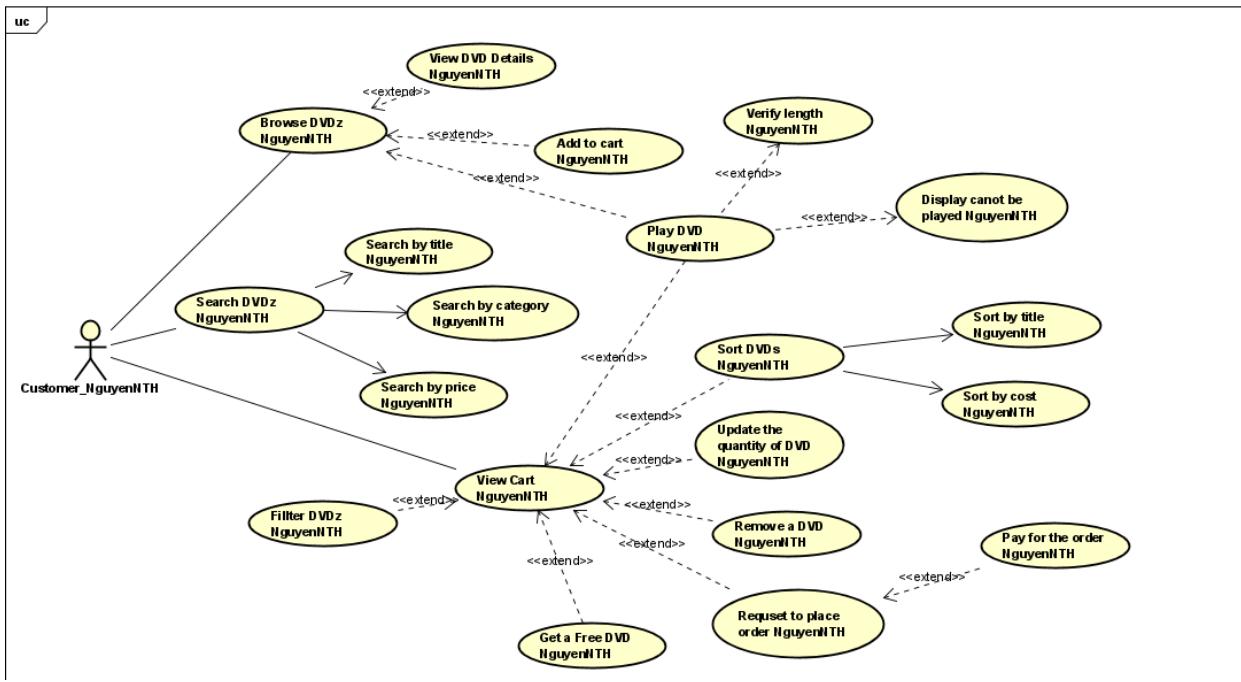


Figure 66. UC Diagram 1

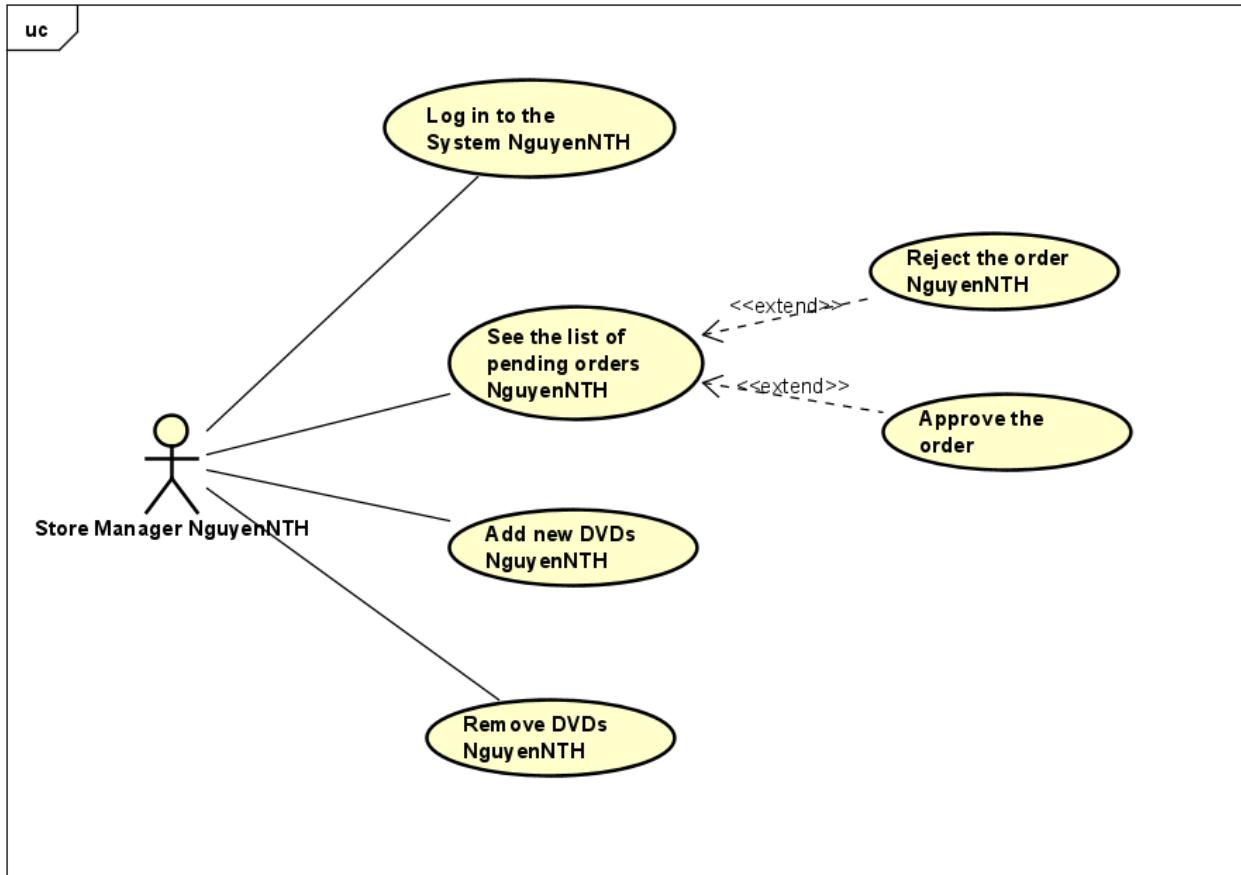


Figure 67..UC Diagram 2

11. Class Diagram

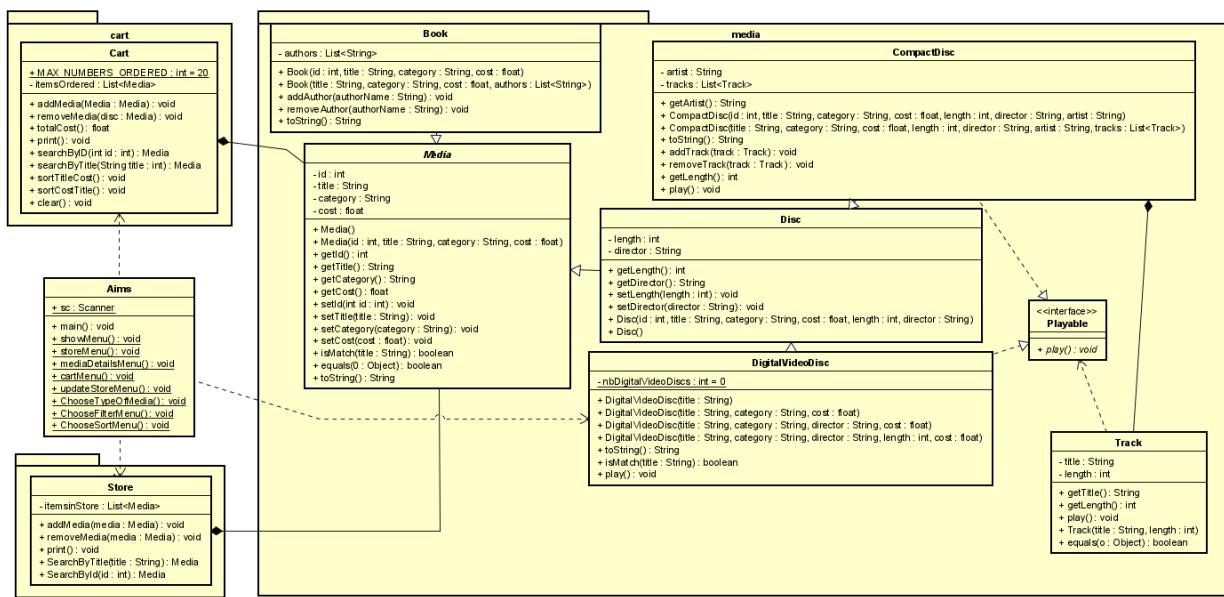


Figure 68. Class Diagram