Report Lab 05

Name: Bùi Công Minh

Student ID: 20235601

1. UseCase Diagram

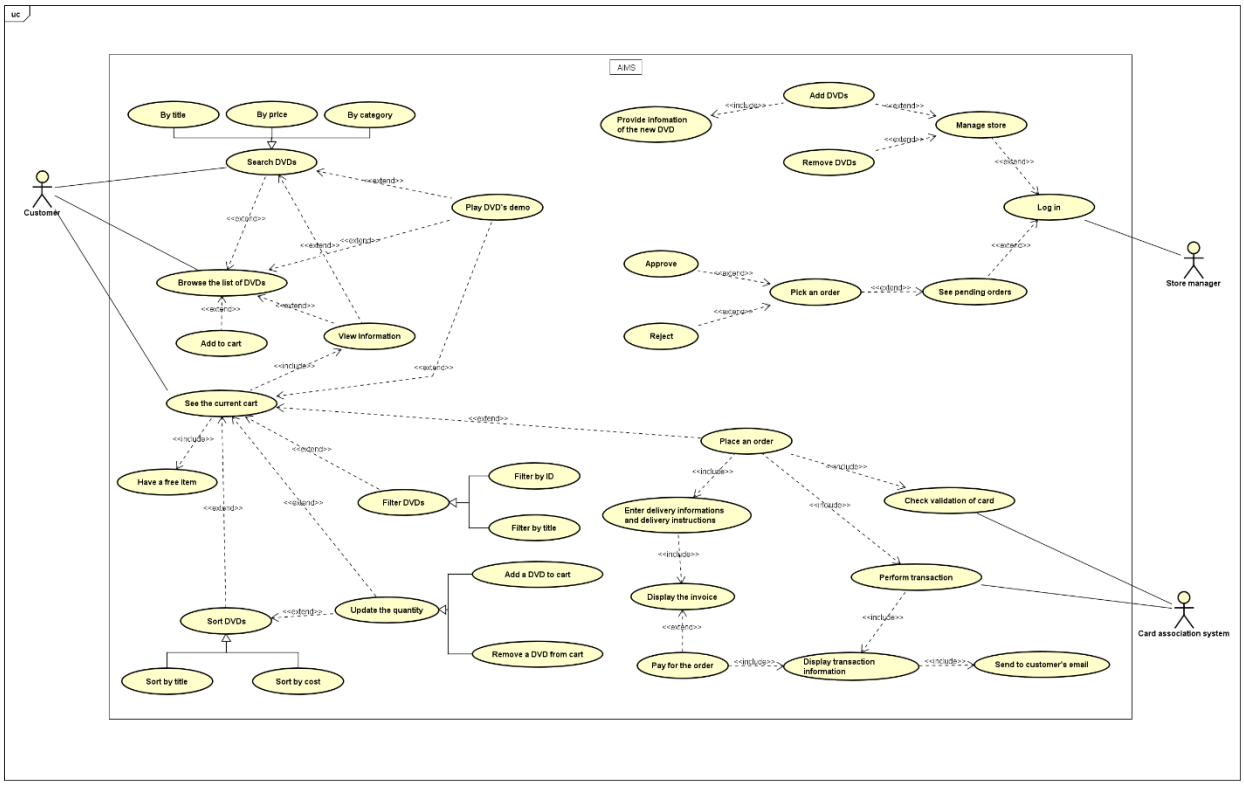

Figure 1: UseCase Diagram

2. Class Diagram

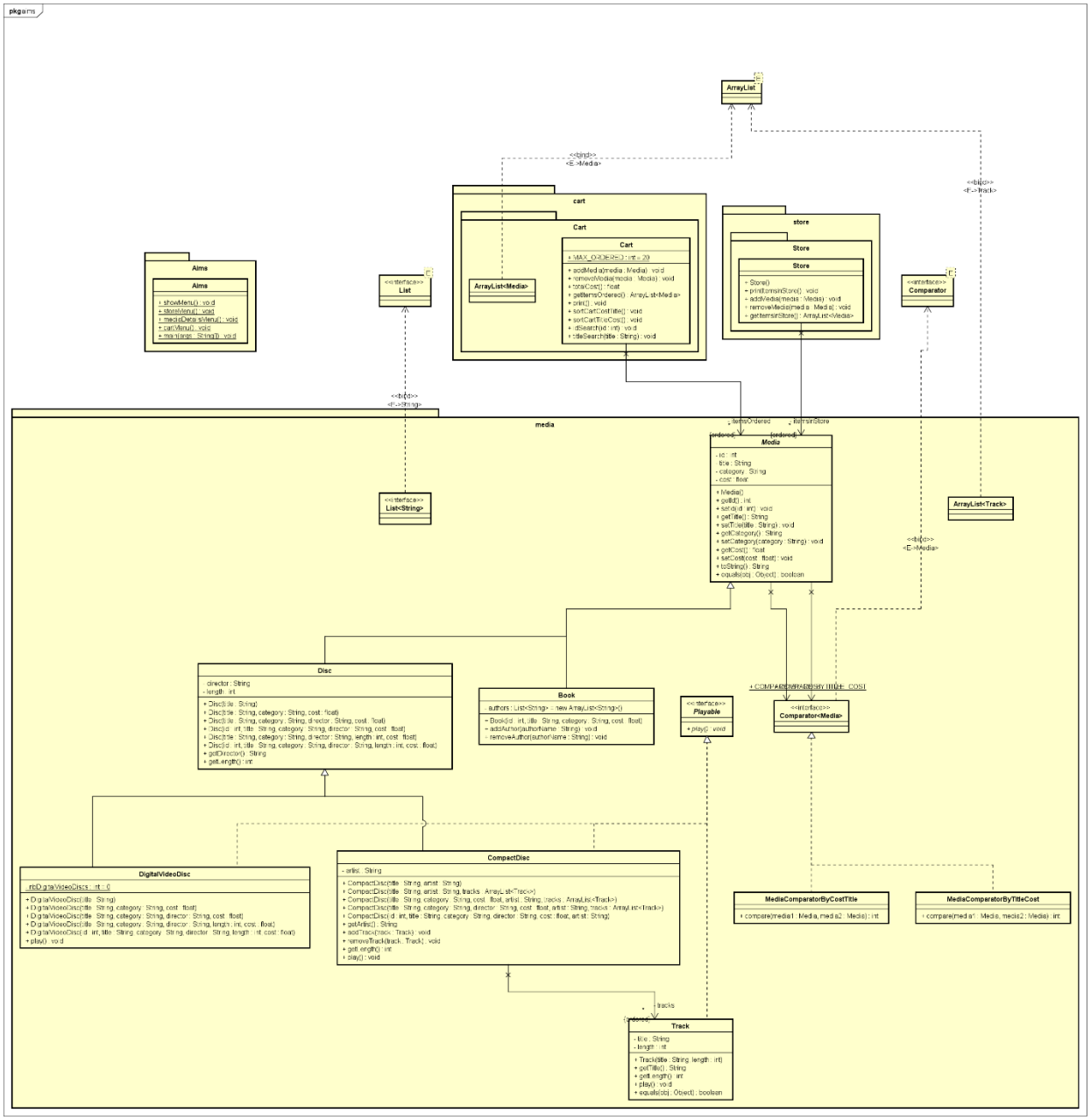

Figure 2: Class Diagram for AimsProject

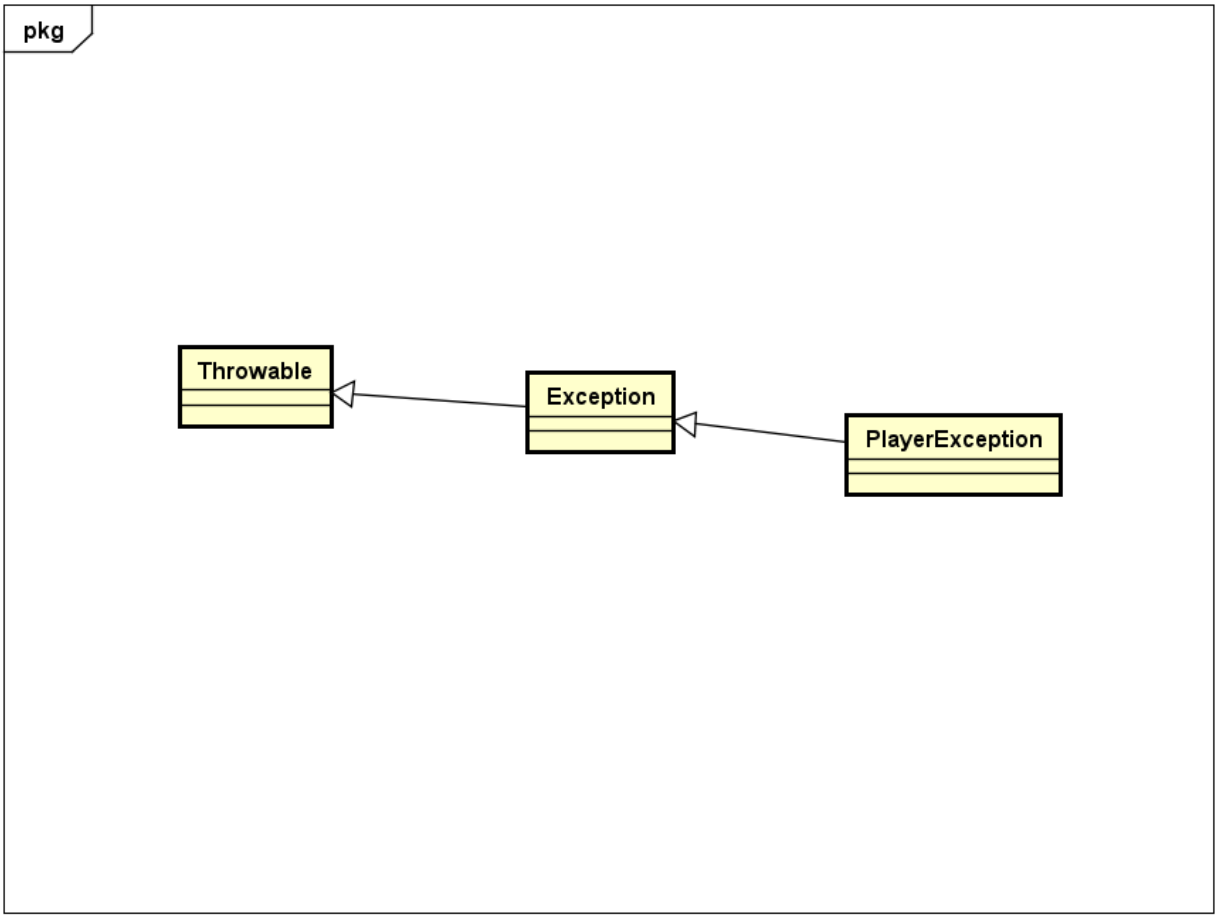

Figure 3: Exceptional Hierarchical Tree

3. Swing components

3.1. AWT Accumulator

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

import java.awt.*;

public class AWTAccumulator extends Frame{
    private TextField tfInput;
    private TextField tfOutput;
    private int sum = 0;

    public AWTAccumulator() {
        setLayout(new GridLayout(2, 2));

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

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

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

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

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

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

    private class TFInputListener implements ActionListener{
        @Override
        public void actionPerformed(ActionEvent e) {
            int numberIn = Integer.parseInt(tfInput.getText());
            sum += numberIn;
            tfInput.setText("");
            tfOutput.setText(sum + "");
        }
    }
}

```

Figure 4: Source code for AWT Accumulator

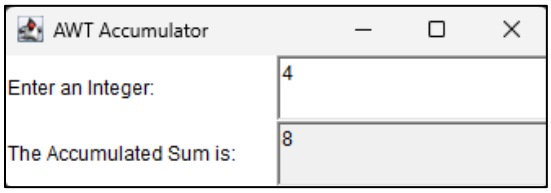

Figure 5: Before

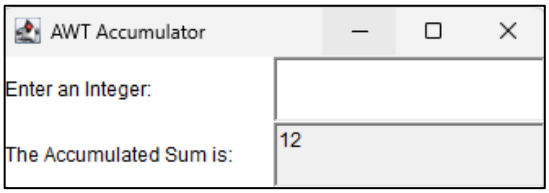

Figure 6: After

3.2. Swing Accumulator

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

import java.awt.Container;

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

    public SwingAccumulator() {
        Container cp = getContentPane();
        cp.setLayout(new GridLayout(2, 2));

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

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

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

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

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

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

    private class TFInputListener implements ActionListener{
        @Override
        public void actionPerformed(ActionEvent e) {
            int numberIn = Integer.parseInt(tfInput.getText());
            sum += numberIn;
            tfInput.setText("");
            tfOutput.setText(sum + "");
        }
    }
}
```

Figure 7: Source code for SwingAccumulator

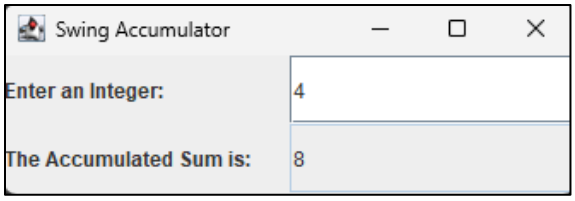

Figure 8: Before

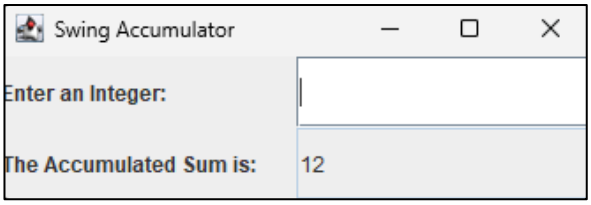

Figure 9: After

4. NumberGrid

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

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

import javax.swing.*;

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

    public NumberGrid() {

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

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

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

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

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

```
36
37      void addButtons(JPanel panelButtons) {
38          ButtonListener btnListener = new ButtonListener();
39          for (int i = 1; i <= 9; i++) {
40              btnNumbers[i] = new JButton("" + i);
41              panelButtons.add(btnNumbers[i]);
42              btnNumbers[i].addActionListener(btnListener);
43          }
44
45          btnDelete = new JButton("DEL");
46          panelButtons.add(btnDelete);
47          btnDelete.addActionListener(btnListener);
48
49          btnNumbers[0] = new JButton("0");
50          panelButtons.add(btnNumbers[0]);
51          btnNumbers[0].addActionListener(btnListener);
52
53          btnReset = new JButton("C");
54          panelButtons.add(btnReset);
55          btnReset.addActionListener(btnListener);
56      }
57
58      private class ButtonListener implements ActionListener{
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              }
65              else if (button.equals("DEL")) {
66                  String tmp = tfDisplay.getText();
67                  tfDisplay.setText(tmp.substring(0, tmp.length() - 1));
68              }
69              else {
70                  tfDisplay.setText("");
71              }
72          }
73      }
74  }
```

Figure 10: Source code for NumberGrid

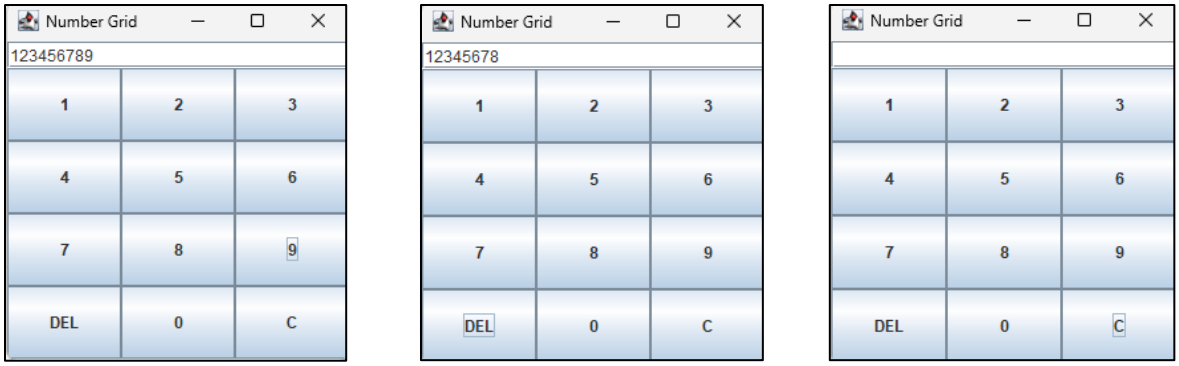

Figure 11: Pressing numbers, DEL and C

5. Painter (Java FX API)

5.1. Create a new FXML file

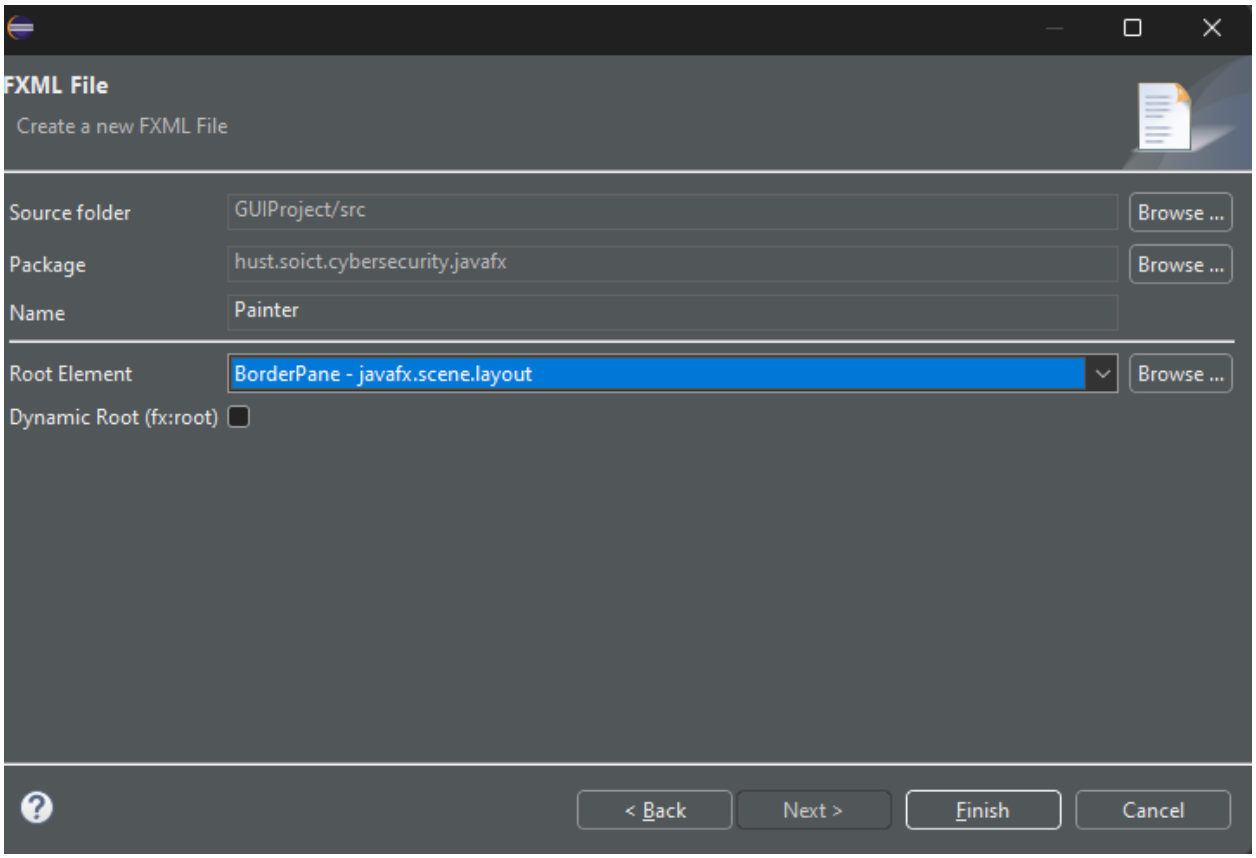

Figure 12: Create a new FXML File

5.2. Building the GUI

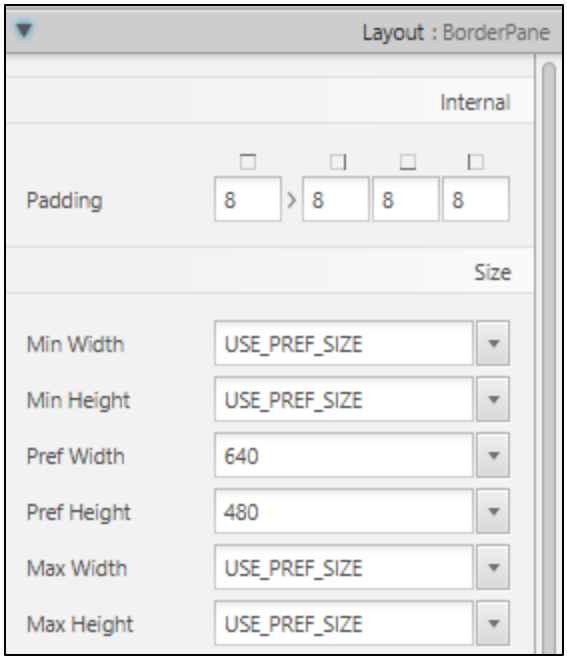

Figure 13: Configuring the BorderPane

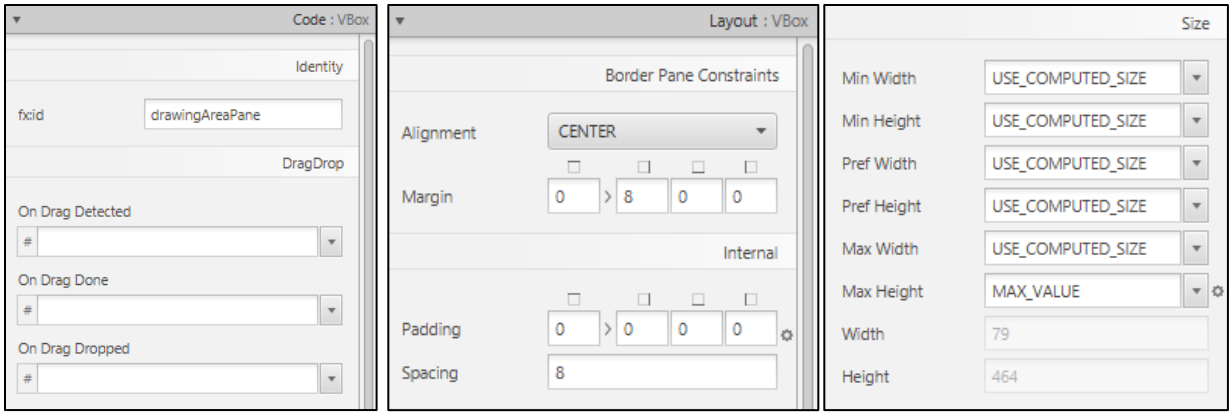

Figure 14: Configuring the Vbox

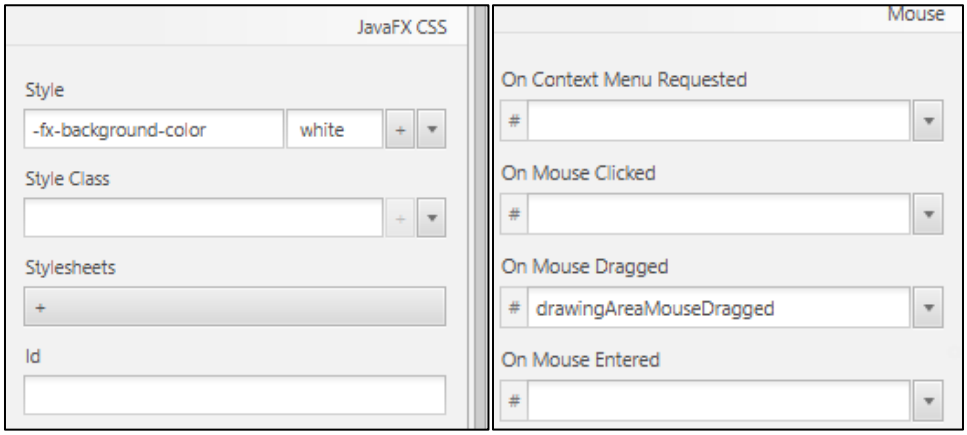

Figure 15: Configuring the Pane

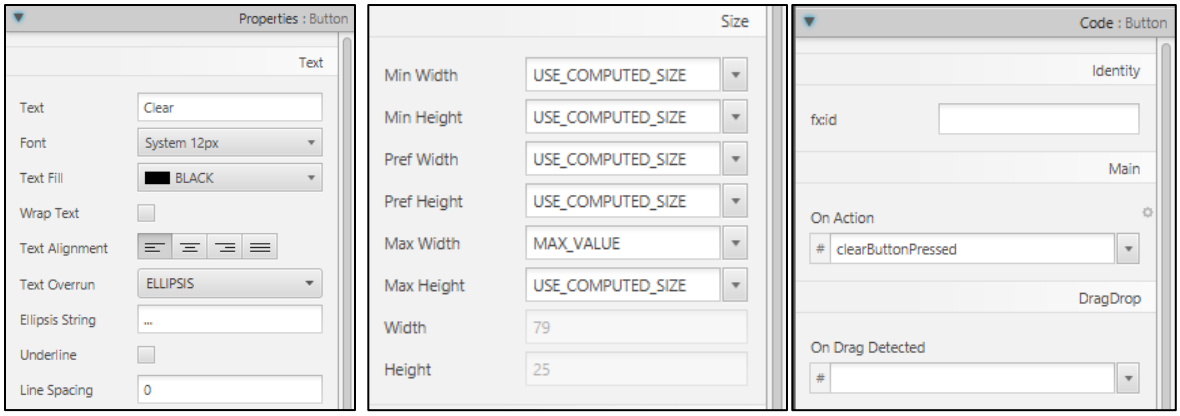

Figure 16: Configuring the Clear button

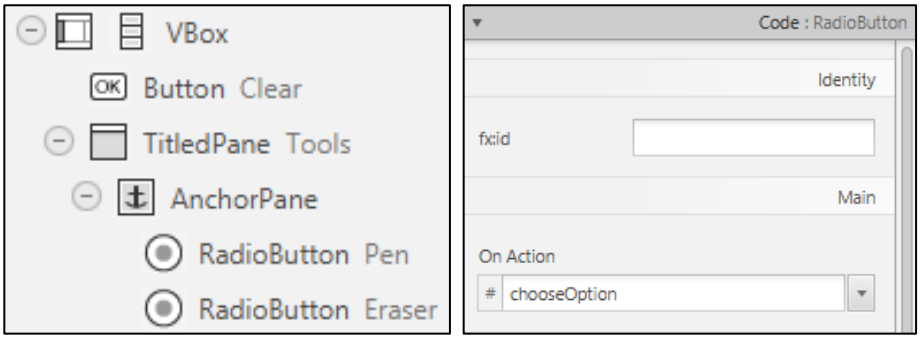

Figure 17: Adding and configuring TitledPane and RadioButtons

```java
package hust.soict.cybersecurity.javafx;

import javafx.event.ActionEvent;

public class PainterController {

    private int color;
    @FXML
    private Pane drawingAreaPane;

    @FXML
    private ToggleGroup identical;

    @FXML
    void chooseOption(ActionEvent event) {

        String button = ((RadioButton)event.getSource()).getText();
        if (button.equals("Pen")) {
            System.out.println("1");
            color = 1;
        }else {
            System.out.println("0");
            color = 0;
        }
    }

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

    @FXML
    void drawingAreaMouseDragged(MouseEvent event) {
        if (color == 1 && event.getX()>=0) {
            Circle newCircle = new Circle(event.getX(), event.getY(), 4.0, Color.BLACK);
            drawingAreaPane.getChildren().add(newCircle);
        }else if(color == 0 && event.getX()>=0){
            Circle newCircle = new Circle(event.getX(), event.getY(), 10.0, Color.WHITE);
            drawingAreaPane.getChildren().add(newCircle);
        }
    }

}
```

Figure 18: Source code for PainterController

```java
package hust.soict.cybersecurity.javafx;

import javafx.application.Application;

public class Painter extends Application{

    @Override
    public void start(Stage stage) throws Exception {
        Parent root = FXMLLoader.load(getClass().getResource("/hust/soict/cybersecurity/javafx/Painter.fxml"));

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

    }

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

```

Figure 19: Source code for Painter

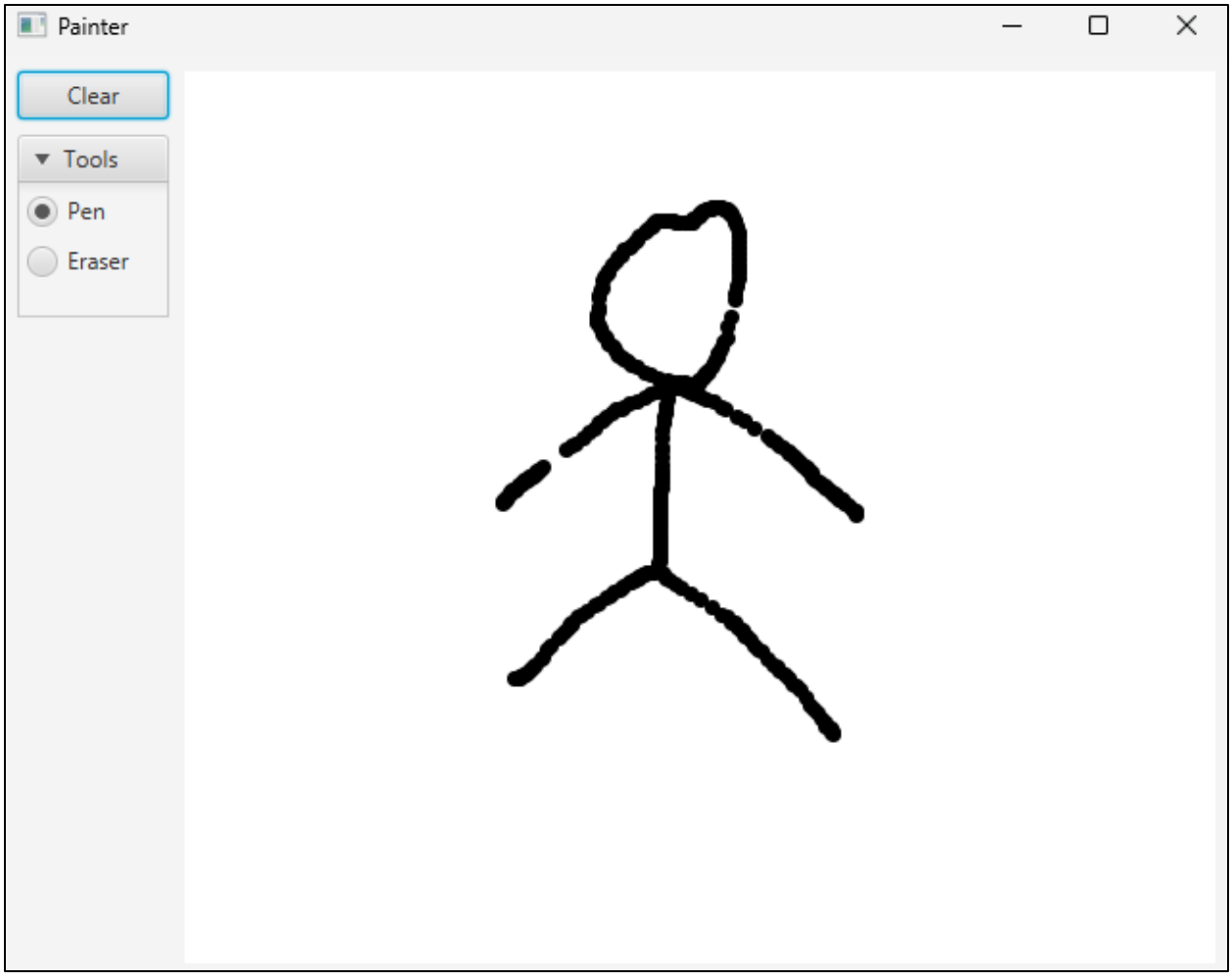

Figure 20: Painter results(Pen)

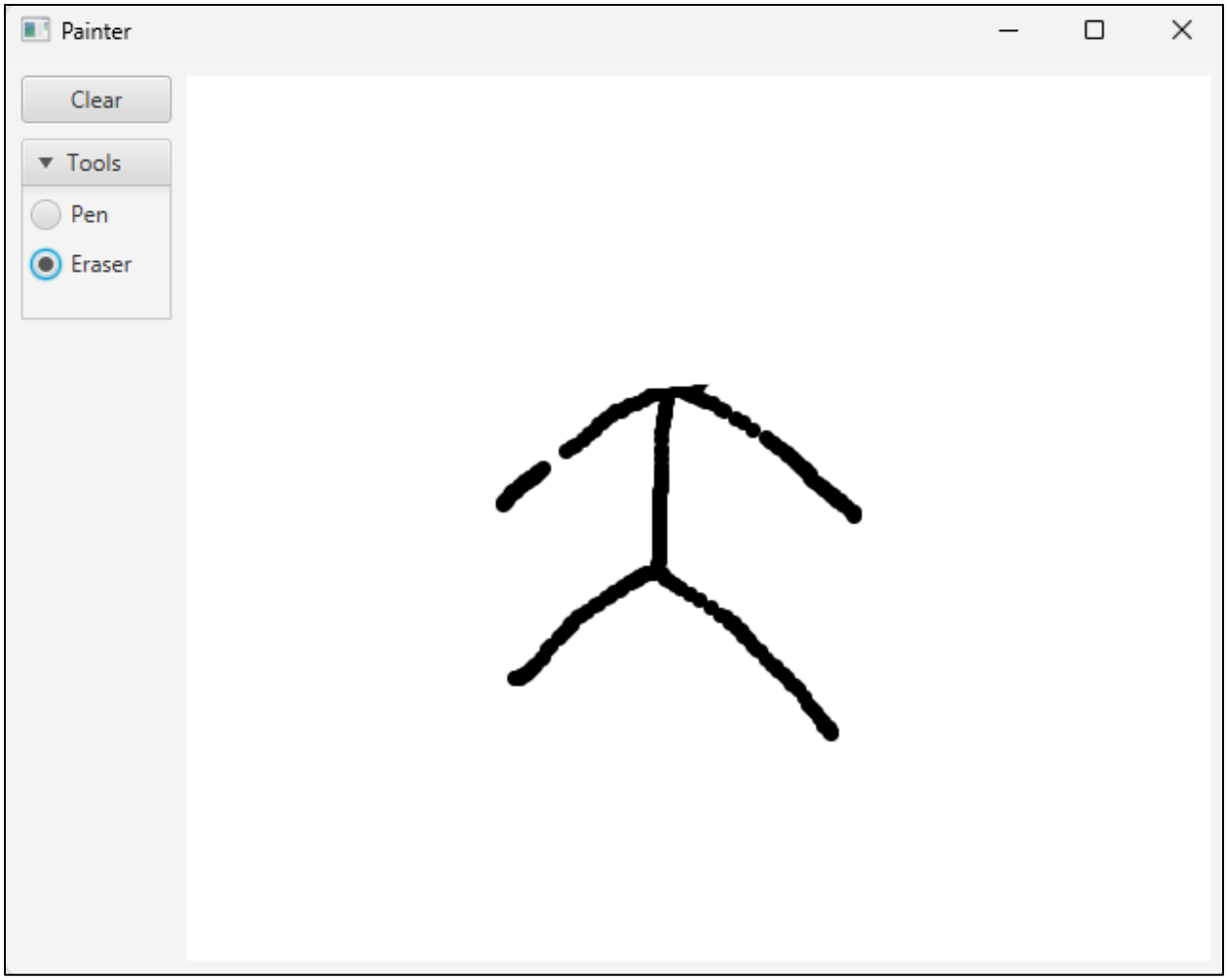

Figure 21: Painter results(Eraser)

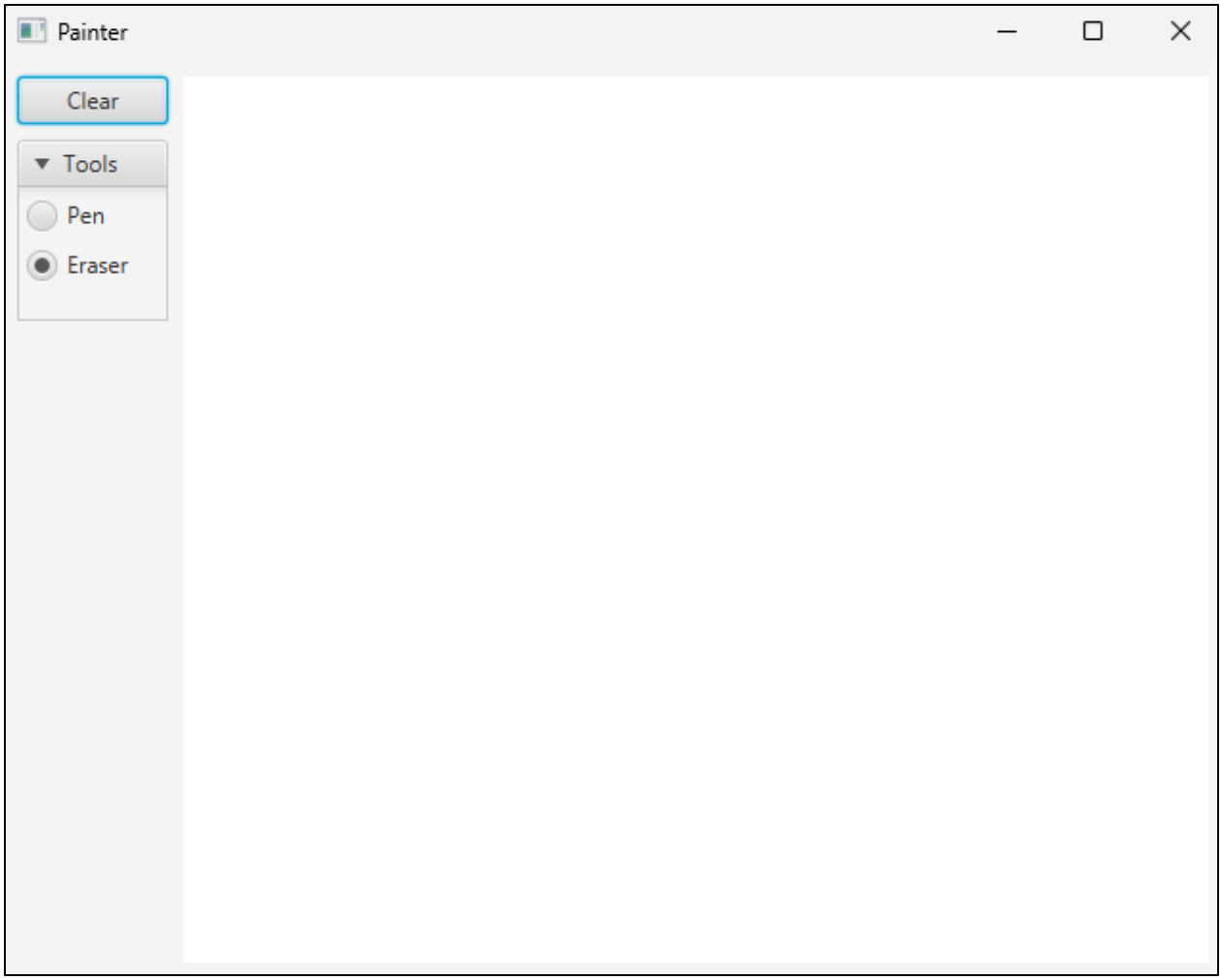

Figure 22: Painter results(Clear)

6. Aims GUI
   6.1.StoreScreen

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

import java.awt.BorderLayout;

public class StoreScreen extends JFrame{
    private Store store;
    private Cart cart;
    private ControllerScreen c;

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

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

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

    }

    JPanel createNorth() {
        JPanel north = new JPanel();
        north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
        north.add(createMenuBar());
        north.add(createHeader());
        return north;
    }

```

```
57      JMenuBar createMenuBar() {
58          JMenu menu = new JMenu("Option");
59          JMenu smUpdateStore = new JMenu("Update Store");
60          JMenuItem addBookMenu= new JMenuItem("Add Book");
61          addBookMenu.addActionListener(e->{
62              c.showAddBookScreen();
63          });
64          smUpdateStore.add(addBookMenu);
65          JMenuItem addCDMenu=new JMenuItem("Add CD");
66          addCDMenu.addActionListener(e->{
67              c.showAddCDCreen();
68          });
69          smUpdateStore.add(addCDMenu);
70          JMenuItem addDVDMenu= new JMenuItem("Add DVD");
71          addDVDMenu.addActionListener(e->{
72              c.showAddDVDScreen();
73          });
74          smUpdateStore.add(addDVDMenu);
75
76          menu.add(smUpdateStore);
77          JMenuItem viewStoreMenu= new JMenuItem("View store");
78          viewStoreMenu.addActionListener(e->{
79              c.showStoreScreen();
80          });
81          menu.add(viewStoreMenu);
82
83          JMenuItem viewCartMenu= new JMenuItem("View cart");
84          viewCartMenu.addActionListener(e->{
85              c.showCartScreen();
86          });
87          menu.add(viewCartMenu);
88
89          JMenuBar menuBar = new JMenuBar();
90          menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
91          menuBar.add(menu);
92
93          return menuBar;
94      }
95
```

```
96      JPanel createHeader() {
97
98          JPanel header = new JPanel();
99          header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));
100
101         JLabel title = new JLabel("AIMS");
102         title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
103         title.setForeground(Color.CYAN);
104
105         JButton cart = new JButton("View cart");
106         cart.setPreferredSize(new Dimension(100, 50));
107         cart.setMaximumSize(new Dimension(100, 50));
108
109         cart.addActionListener(new ActionListener() {
110             @Override
111             public void actionPerformed(ActionEvent e) {
112                 c.showCartScreen();
113             }
114     });
115
116         header.add(Box.createRigidArea(new Dimension(10, 10)));
117         header.add(title);
118         header.add(Box.createHorizontalGlue());
119         header.add(cart);
120         header.add(Box.createRigidArea(new Dimension(10, 10)));
121
122         return header;
123     }
124
125     JPanel createCenter() {
126
127         JPanel center = new JPanel();
128         center.setLayout(new GridLayout(3, 3, 2, 2));
129
130         ArrayList<Media> mediaInStore = store.getItemsInStore();
131         for (int i = 0; i < mediaInStore.size(); i++) {
132             MediaStore cell = new MediaStore(mediaInStore.get(i), cart);
133             center.add(cell);
134         }
135
136         return center;
137     }
```

Figure 23: Source code of StoreScreen

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

import java.awt.BorderLayout;

public class MediaStore extends JPanel{
    private Media media;

    public MediaStore(Media media, Cart cart) {
        this.media = media;
        this.setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));

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

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

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

        JButton addCartBtn = new JButton("Add to cart");
        addCartBtn.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                try {
                    cart.addMedia(media);
                } catch (LimitExceededException e1) {
                    e1.printStackTrace();
                }
            }
        });

        container.add(addCartBtn);
```

```
54
55          if (media instanceof Playable) {
56              JButton playBtn = new JButton("Play");
57              playBtn.addActionListener(new ActionListener() {
58                  @Override
59                  public void actionPerformed(ActionEvent e) {
60                      JDialog playDialog = new JDialog();
61                      JPanel mainGUI = new JPanel(new BorderLayout());
62                      mainGUI.setBorder(new EmptyBorder(100, 100, 100, 100));
63                      mainGUI.add(new JLabel("Playing....."), BorderLayout.CENTER);
64                      JPanel buttonPanel = new JPanel(new FlowLayout());
65                      mainGUI.add(buttonPanel, BorderLayout.SOUTH);
66                      JButton close = new JButton("Stop");
67                      close.addActionListener(ev->playDialog.setVisible(false));
68                      buttonPanel.add(close);
69                      playDialog.setContentPane(mainGUI);
70                      playDialog.setLocationRelativeTo(playBtn);
71                      playDialog.pack();
72                      playDialog.setVisible(true);
73                  }
74              });
75              container.add(playBtn);
76          }
77
78          this.add(Box.createVerticalGlue());
79          this.add(title);
80          this.add(cost);
81          this.add(Box.createVerticalGlue());
82          this.add(container);
83
84          this.setBorder(BorderFactory.createLineBorder(Color.BLACK));
85      }
86  }
87
```

Figure 24: Source code of MediaStore

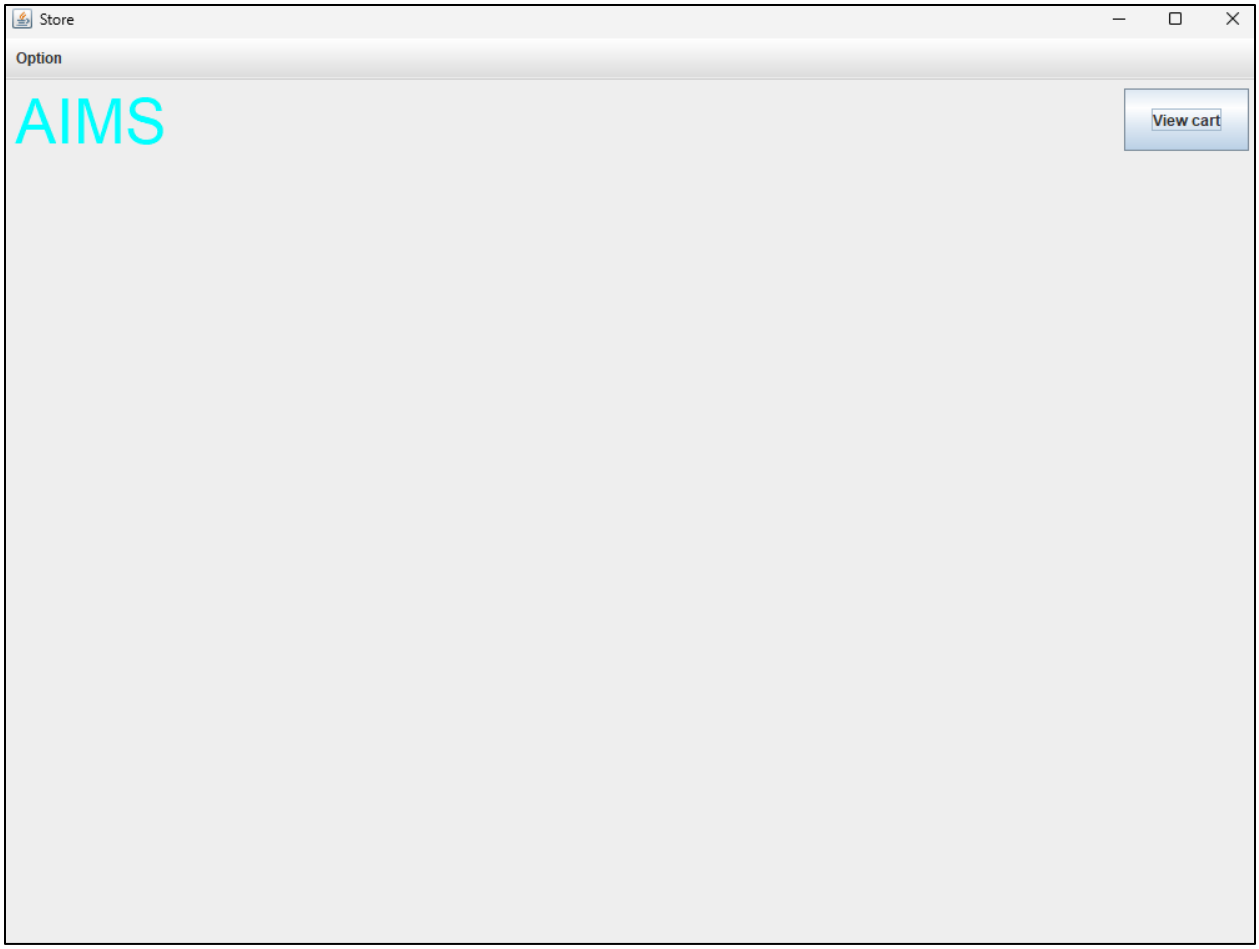

Figure 25: Result

6.2. Update Store

```java
public class AddItemToStoreScreen extends JFrame {
    protected Store store;
    protected Cart cart;
    protected JTextField title;
    protected JTextField category;
    protected JTextField cost;
    protected JButton addBtn;
    protected ControllerScreen c;
    protected JPanel center;
    protected int numberInput = 3;

    JMenuBar createMenuBar() {
        JMenu menu = new JMenu("Option");

        JMenu smUpdateStore = new JMenu("Update Store");
        JMenuItem addBookMenu = new JMenuItem("Add Book");
        addBookMenu.addActionListener(e -> {
            c.showAddBookScreen();
        });
        smUpdateStore.add(addBookMenu);
        JMenuItem addCDMenu = new JMenuItem("Add CD");
        addCDMenu.addActionListener(e -> {
            c.showAddCDCreen();
        });
        smUpdateStore.add(addCDMenu);
        JMenuItem addDVDMenu = new JMenuItem("Add DVD");
        addDVDMenu.addActionListener(e -> {
            c.showAddDVDScreen();
        });
        smUpdateStore.add(addDVDMenu);

        menu.add(smUpdateStore);
        JMenuItem viewStoreMenu = new JMenuItem("View store");
        viewStoreMenu.addActionListener(e -> {
            c.showStoreScreen();
        });
        menu.add(viewStoreMenu);
```

```
66
67          JMenuItem viewCartMenu = new JMenuItem("View cart");
68          viewCartMenu.addActionListener(e -> {
69              c.showCartScreen();
70          });
71          menu.add(viewCartMenu);
72
73          JMenuBar menuBar = new JMenuBar();
74          menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
75          menuBar.add(menu);
76
77          return menuBar;
78      }
79
80      JPanel createHeader() {
81          JPanel header = new JPanel();
82          header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));
83
84          JLabel title = new JLabel("AIMS");
85          title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
86          title.setForeground(Color.CYAN);
87
88          JButton btnCart = new JButton("View cart");
89
90          btnCart.setPreferredSize(new Dimension(100, 50));
91          btnCart.setMaximumSize(new Dimension(100, 50));
92          btnCart.addActionListener(new ActionListener() {
93              @Override
94              public void actionPerformed(ActionEvent e) {
95                  c.showCartScreen();
96                  System.out.println("test now");
97              }
98          });
99
100         header.add(Box.createRigidArea(new Dimension(10, 10)));
101         header.add(title);
102         header.add(Box.createHorizontalGlue());
103         header.add(btnCart);
104         header.add(Box.createRigidArea(new Dimension(10, 10)));
105
106         return header;
107     }
108
```

```
109     JPanel createNorth() {
110         JPanel north = new JPanel();
111         north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
112         north.add(createMenuBar());
113         north.add(createHeader());
114         return north;
115     }
116 
117     void updateAdd(JPanel panel) {
118         JButton tes = new JButton("Add");
119         center.add(tes, BorderLayout.CENTER);
120     }
121 
122     JPanel createCenter() {
123         center = new JPanel();
124         center.setLayout(new BorderLayout());
125         SpringLayout layout = new SpringLayout();
126         JPanel p = new JPanel(layout);
127         JLabel titleLabel = new JLabel("Title", JLabel.TRAILING);
128         p.add(titleLabel);
129         title = new JTextField(10);
130         titleLabel.setLabelFor(title);
131         title.addActionListener(new ActionListener() {
132 
133             public void actionPerformed(ActionEvent e) {
134 
135                 System.out.print("Hello");
136                 JButton tes = new JButton("Add");
137                 center.add(tes, BorderLayout.CENTER);
138                 c.updateAddItem();
139 
140             }
141         });
142 
```

```
143         p.add(title);
144         JLabel categoryLabel = new JLabel("Category", JLabel.TRAILING);
145         p.add(categoryLabel);
146         category = new JTextField(10);
147         p.add(category);
148         categoryLabel.setLabelFor(category);
149         JLabel costLabel = new JLabel("Cost", JLabel.TRAILING);
150         p.add(costLabel);
151         cost = new JTextField(10);
152         costLabel.setLabelFor(cost);
153         p.add(cost);
154 
155         updateAdd(p);
156         SpringUtilities.makeCompactGrid(p,
157                 this.numberInput, 2,
158                 6, 6,
159                 6, 6);
160 
161         p.setMaximumSize(new Dimension(
162                 100,
163                 300));
164 
165         center.add(p, BorderLayout.NORTH);
166         return center;
167     }
168 
169     public AddItemToStoreScreen(Store store, Cart cart, ControllerScreen c) {
170         this.store = store;
171         this.cart = cart;
172         this.c = c;
173         Container cp = getContentPane();
174         cp.setLayout(new BorderLayout());
175 
176         cp.add(createNorth(), BorderLayout.NORTH);
177         cp.add(createCenter(), BorderLayout.CENTER);
178 
179         setTitle("Store");
180         setSize(1024, 768);
181 
182     }
```

Figure 26: AddItemToStoreScreen

```java
public class AddDigitalVideoDiscToStoreScreen extends AddItemToStoreScreen {
    private JTextField director;
    private JTextField length;

    public AddDigitalVideoDiscToStoreScreen(Store store, Cart cart, ControllerScreen c) {
        super(store, cart, c);
    }
    @Override
    void updateAdd(JPanel panel) {
        this.numberInput = 6;
        JLabel directorLabel = new JLabel("Director", JLabel.TRAILING);
        panel.add(directorLabel);
        director = new JTextField(2);
        director.setPreferredSize(new Dimension(150, 20));
        directorLabel.setLabelFor(director);
        panel.add(director);
        JLabel lengthLabel = new JLabel("Length", JLabel.TRAILING);
        panel.add(lengthLabel);
        length = new JTextField(2);
        lengthLabel.setLabelFor(length);
        panel.add(length);
        JButton tes = new JButton("Ddd");
        tes.setVisible(false);
        panel.add(tes);
        panel.setPreferredSize(new Dimension(100, 300));
        addBtn = new JButton("Ddd");
        addBtn.addActionListener(e -> {
            addMedia();
        });
        panel.add(addBtn);
    }

    public void addMedia() {
        String title = this.title.getText();
        String director = this.director.getText();
        String category = this.category.getText();
        float cost = Float.parseFloat(this.cost.getText());
        int length = Integer.parseInt(this.length.getText());
        DigitalVideoDisc dvd = new DigitalVideoDisc(title, category, director, length, cost);
        this.store.addMedia(dvd);
        JOptionPane.showMessageDialog(null, "DVD added successfully!");
        clearTextField();

    }
    public void clearTextField(){
        this.title.setText("");
        this.director.setText("");
        this.cost.setText("");
        this.length.setText("");
        this.category.setText("");
    }


}
```

Figure 27: AddDigitalVideoDiscToStoreScreen

```java
public class AddCompactDiscToStoreScreen extends AddItemToStoreScreen {
    private JTextField artist;
    private JTextField listTrack;

    public AddCompactDiscToStoreScreen(Store store, Cart cart, ControllerScreen c) {
        super(store, cart, c);
    }

    @Override
    void updateAdd(JPanel panel) {
        this.numberInput = 6;

        JLabel artistLabel = new JLabel("Artist", JLabel.TRAILING);
        panel.add(artistLabel);
        artist = new JTextField(2);
        artist.setPreferredSize(new Dimension(150, 20));
        artistLabel.setLabelFor(artist);
        panel.add(artist);
        JLabel listTrackLabel = new JLabel("List track (each track separated by a comma Ex: track-length)",
                JLabel.TRAILING);

        panel.add(listTrackLabel);
        listTrack = new JTextField(2);
        listTrackLabel.setLabelFor(listTrack);
        panel.add(listTrack);
        JButton tes = new JButton("Add");
        tes.setVisible(false);
        panel.add(tes);
        panel.setPreferredSize(new Dimension(100, 300));
        addBtn = new JButton("Add");
        addBtn.addActionListener(e -> {
            addMedia();
        });
        panel.add(addBtn);
    }

```

```
52      public void addMedia() {
53          String title = this.title.getText();
54          String listTrack = this.listTrack.getText();
55          String category = this.category.getText();
56          float cost = Float.parseFloat(this.cost.getText());
57          String artist = (this.artist.getText());
58          String[] arrayTrack = listTrack.trim().split(",");
59
60          CompactDisc cd = new CompactDisc(title, category, artist, cost);
61          ;
62          for (String track : arrayTrack) {
63              String titleTrack = track.split("-")[0].trim();
64              int lengthTrack = Integer.parseInt(track.split("-")[1].trim());
65              Track newTrack = new Track(titleTrack, lengthTrack);
66              cd.addTrack(newTrack);
67
68          }
69          this.store.addMedia(cd);
70          JOptionPane.showMessageDialog(null, "CD added successfully!");
71          clearTextField();
72
73      }
74
75      public void clearTextField() {
76          this.title.setText("");
77          this.listTrack.setText("");
78          this.cost.setText("");
79          this.artist.setText("");
80          this.category.setText("");
81      }
82
83  }
```

Figure 28: AddCompactDiscToStoreScreen

```java
public class AddBookToStoreScreen extends AddItemToStoreScreen {
    private JTextField listAuthor;

    public AddBookToStoreScreen(Store store, Cart cart, ControllerScreen c) {
        super(store, cart, c);
    }
    @Override
    void updateAdd(JPanel panel) {
        this.numberInput = 5;

        JLabel listAuthorLabel = new JLabel("Authors(Names are separated by a comma)", JLabel.TRAILING);
        panel.add(listAuthorLabel);
        listAuthor = new JTextField(2);
        listAuthor.setPreferredSize(new Dimension(150, 20));
        listAuthorLabel.setLabelFor(listAuthor);
        panel.add(listAuthor);

        JButton tes = new JButton("Add");
        tes.setVisible(false);
        panel.add(tes);
        panel.setPreferredSize(new Dimension(100, 300));
        addBtn = new JButton("Add");
        addBtn.addActionListener(e -> {
            addMediaToStore();
        });
        panel.add(addBtn);
    }
    public void addMediaToStore() {
        String title = this.title.getText();
        String listAuthor = this.listAuthor.getText();
        String[] arrayAuthor=listAuthor.split(",");
        String category = this.category.getText();
        float cost = Float.parseFloat(this.cost.getText());
        Book book = new Book(title, category, cost);
        for(String author:arrayAuthor) {
        book.addAuthor(author);
        }
        this.store.addMedia(book);
        JOptionPane.showMessageDialog(null, "Book added successfully!");
        clearTextField();

    }
    public void clearTextField(){
        this.title.setText("");
        this.listAuthor.setText("");
        this.cost.setText("");
        this.category.setText("");
    }


}
```

Figure 29: AddBookToStoreScreen

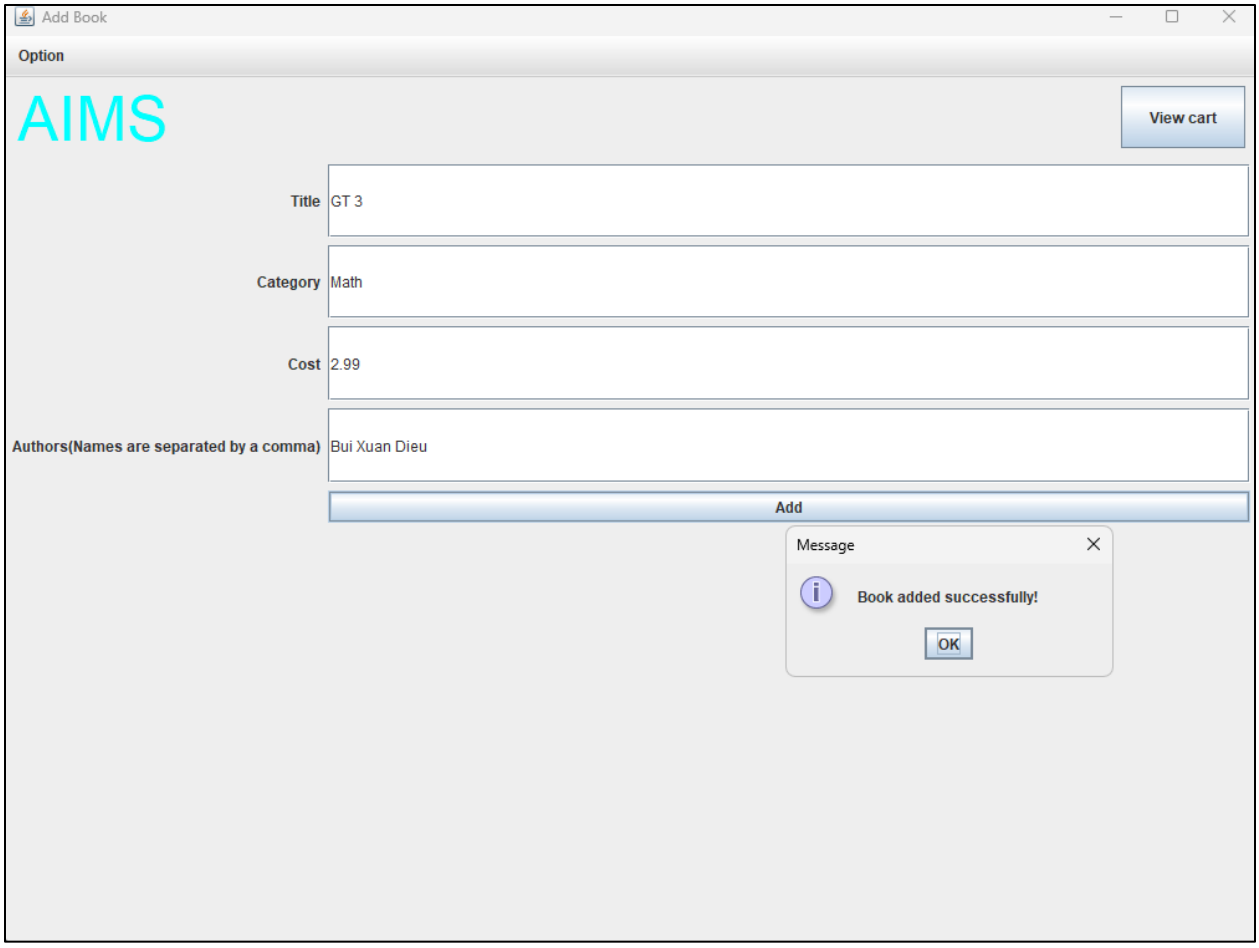

Figure 30: Adding a book to store in AIMS

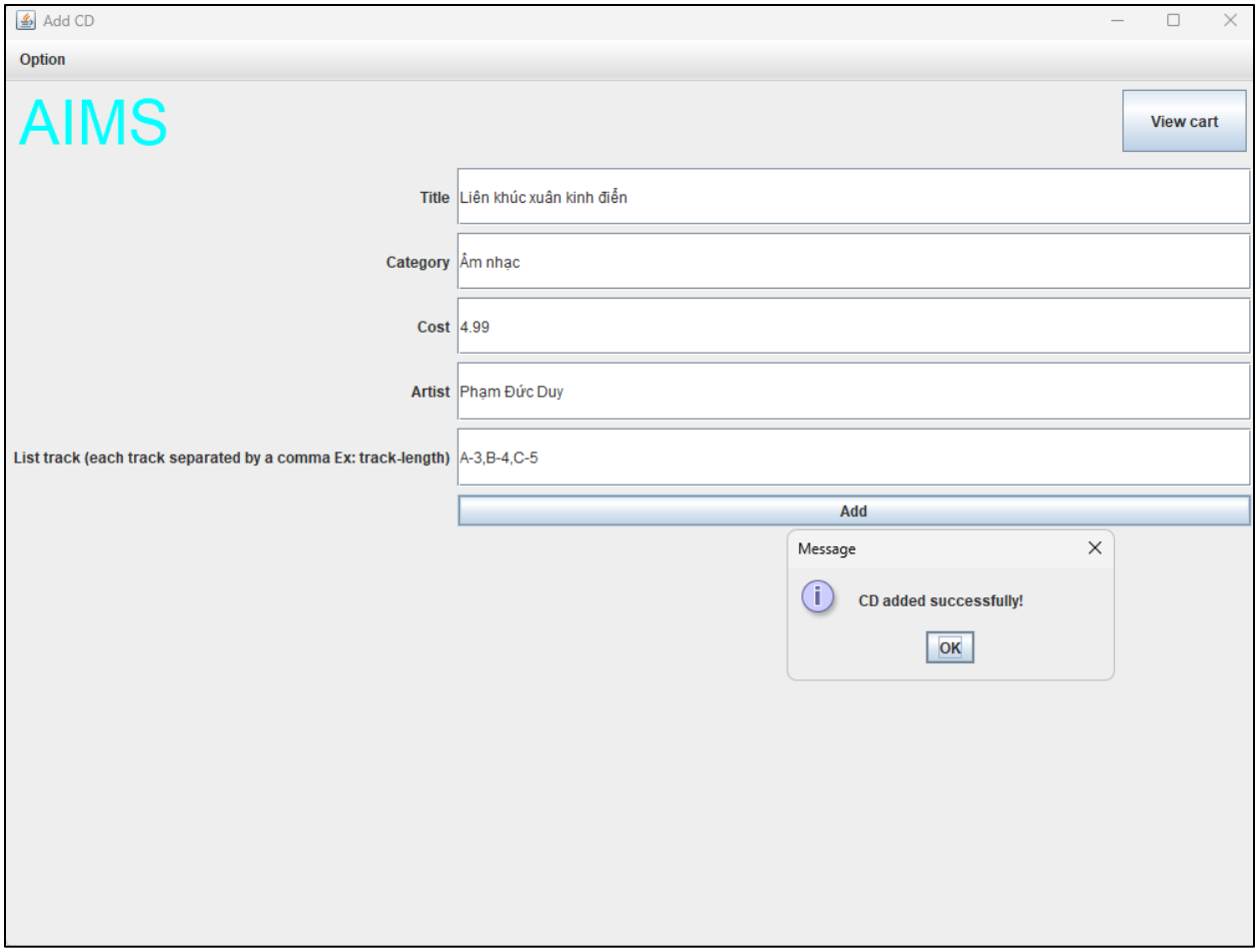

Figure 31: Adding a CD to store in AIMS

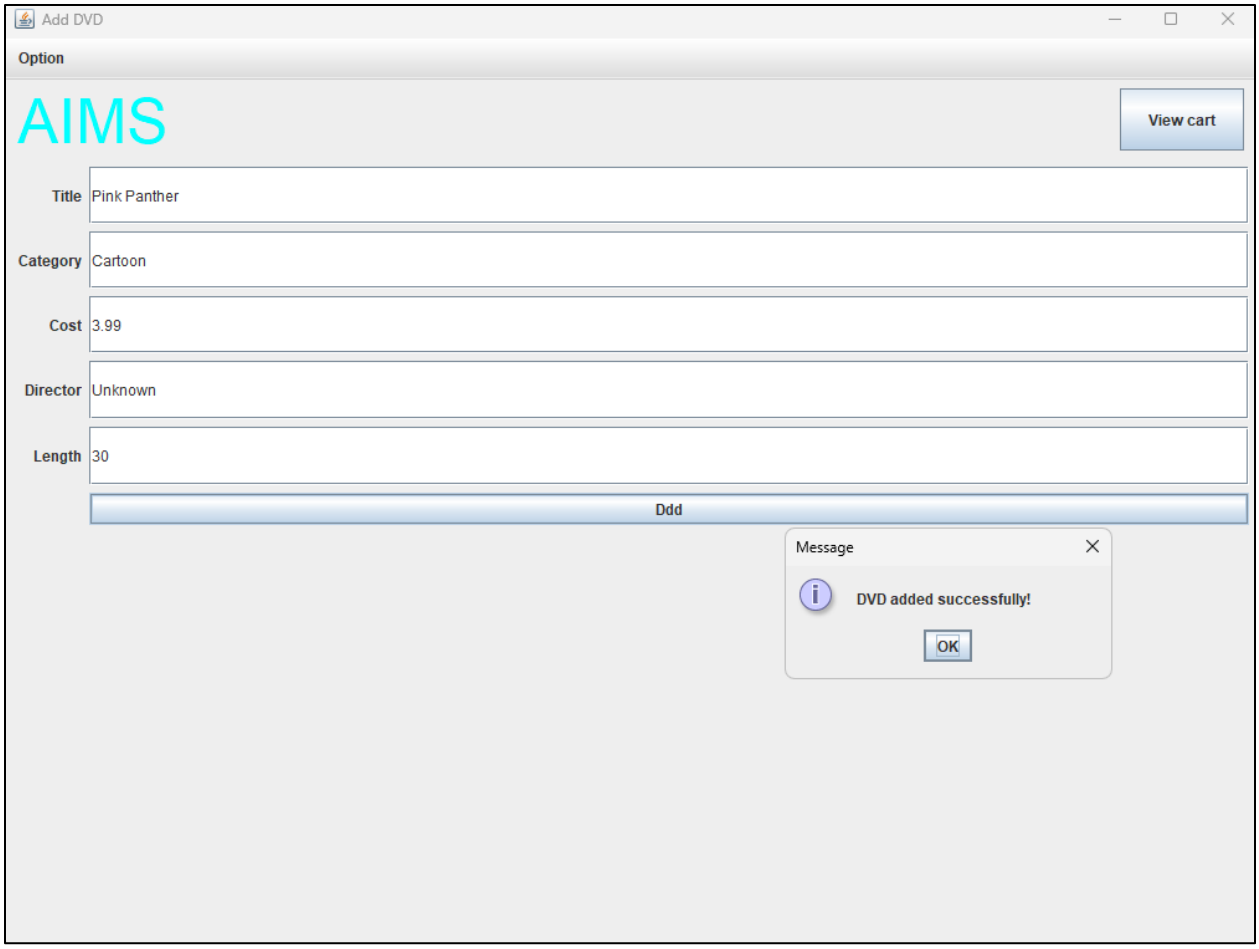

Figure 32: Adding a DVD to store in AIMS

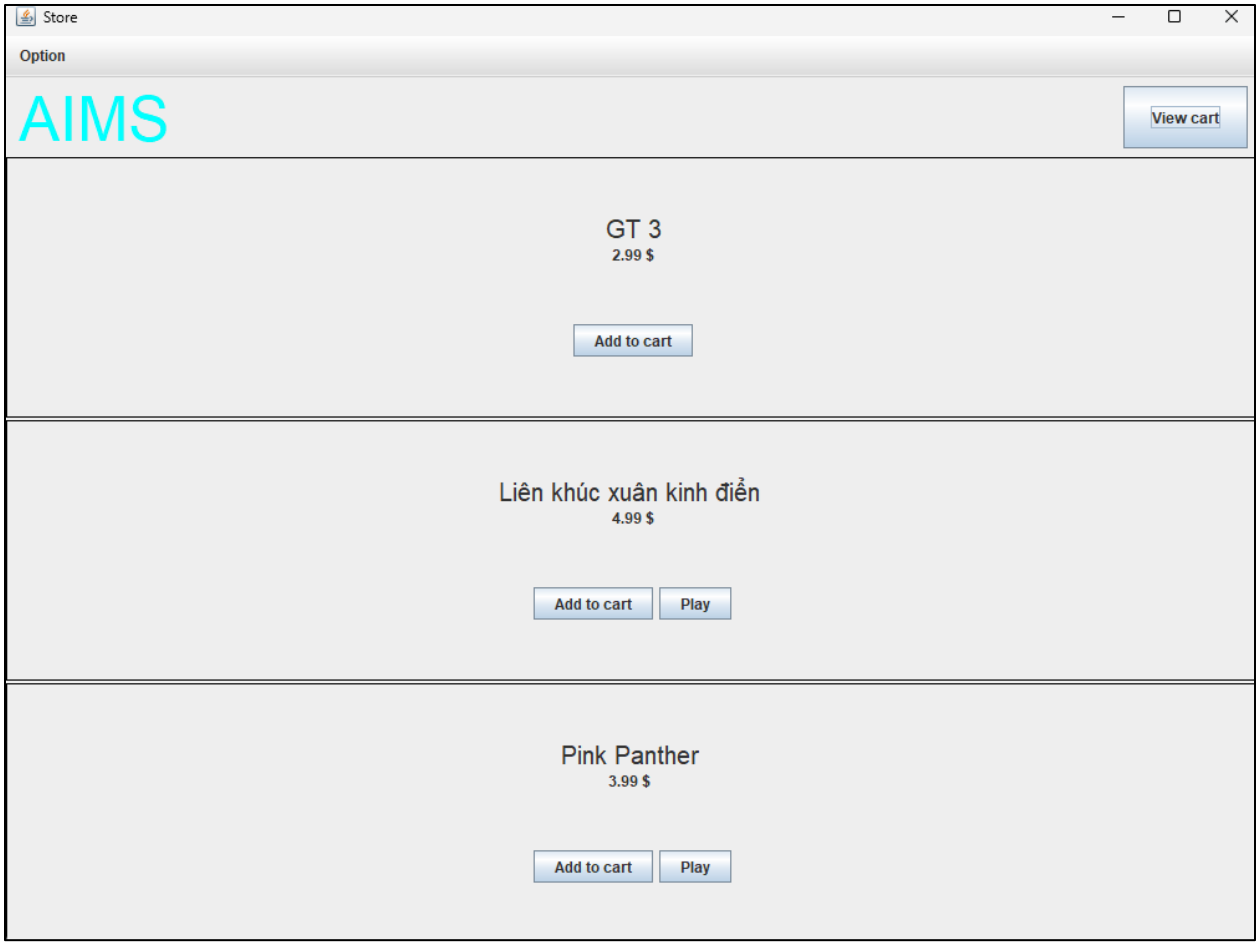

Figure 33: After updating the store

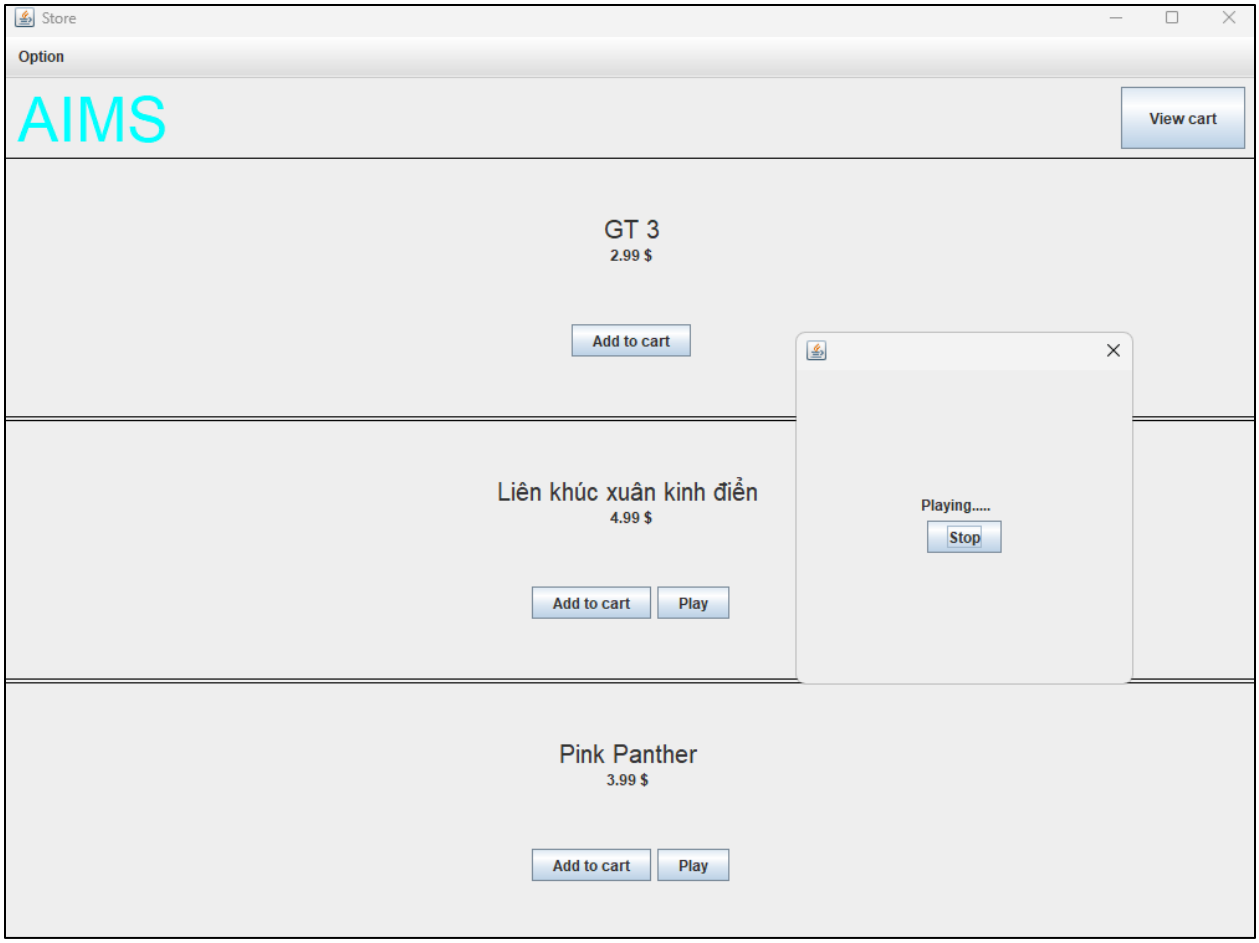

Figure 34: Playing media

6.3.CartScreen

```java
public class CartScreen extends JFrame {
    private Cart cart;
    private ControllerScreen controllerscreen;

    public CartScreen(Cart cart, ControllerScreen c) {
        super();
        this.cart = cart;

        JFXPanel fxPanel = new JFXPanel();
        this.add(fxPanel);
        this.setPreferredSize(new Dimension(1024, 768));
        this.setTitle("Cart");
        this.setVisible(true);
        pack();

        Platform.runLater(new Runnable(){
            @Override
            public void run() {
                try {
                    FXMLLoader loader = new FXMLLoader(getClass().
                            getResource("/hust/soict/cybersecurity/aims/screen/cart.fxml"));
                    CartScreenController controller = new CartScreenController(cart, c);
                    loader.setController(controller);
                    Parent root = loader.load();
                    fxPanel.setScene(new Scene(root));
                }
                catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });
    }
```

Figure 35: CartScreen

```java
public class CartScreenController {

    private Cart cart;
    private ControllerScreen controllerScreen;

    @FXML
    private TableView<Media> tblMedia;

    @FXML
    private TableColumn<Media, String> colMediaTitle;

    @FXML
    private TableColumn<Media, String> colMediacategory;

    @FXML
    private TableColumn<Media, Float> colMediaCost;

    @FXML
    private Button btnPlay;

    @FXML
    private Label totalCost;

    @FXML
    private Button btnRemove;

    @FXML
    private ToggleGroup filterCategory;

    @FXML
    private TextField tfFilter;

    @FXML
    private RadioButton radioBtnFilterId;

    @FXML
    private RadioButton radioBtnFilterTitle;

    @FXML
    private Button btnOrder;

    @FXML
    private Button btnStop;
```

```
66      @FXML
67      private Label playingMedia;
68
69      public CartScreenController(Cart cart , ControllerScreen controllerScreen) {
70          super();
71          this.cart = cart;
72          this.controllerScreen = controllerScreen;
73      }
74
75      @FXML
76      void btnRemovePressed(ActionEvent event) {
77          Media media = tblMedia.getSelectionModel().getSelectedItem();
78          cart.removeMedia(media);
79          totalCost.setText(cart.totalCost()+"$");
80      }
81
82      @FXML
83      void btnOrderPressed(ActionEvent event) {
84          System.out.println("Order");
85          btnOrder.setText("Success!!!");
86          btnOrder.setDisable(true);
87          cart.getItemsOrdered().removeAll(cart.getItemsOrdered());
88          totalCost.setText("0.0$");
89          PauseTransition pt = new PauseTransition(Duration.seconds(1));
90          pt.setOnFinished(e ->{
91              btnOrder.setDisable(false);
92              playingMedia.setVisible(false);
93              btnPlay.setVisible(false);
94              btnStop.setVisible(false);
95              btnOrder.setText("Order");
96          } );
97          pt.playFromStart();
98      }
99
100     @FXML
101     void btnPlayPressed(ActionEvent event) {
102         Media media = tblMedia.getSelectionModel().getSelectedItem();
103         playingMedia.setText("Playing "+media.getTitle()+"....");
104         playingMedia.setVisible(true);
105         btnStop.setVisible(true);
106     }
107
```

```
    @FXML
    void btnStopPressed(ActionEvent event) {
        playingMedia.setVisible(false);
        btnStop.setVisible(false);
    }

    @FXML
    private void initialize() {
        colMediaTitle.setCellValueFactory(
                new PropertyValueFactory<Media, String>("title"));
        colMediacategory.setCellValueFactory(
                new PropertyValueFactory<Media, String>("category"));
        colMediaCost.setCellValueFactory(
                new PropertyValueFactory<Media, Float>("cost"));
        tblMedia.setItems(this.cart.getItemsOrdered());

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

        tblMedia.getSelectionModel().selectedItemProperty().addListener(
            new ChangeListener<Media>() {

                @Override
                public void changed(ObservableValue<? extends Media> observable, Media oldValue, Media newValue) {
                    if (newValue != null) {
                        updateButtonBar(newValue);
                    }

                    totalCost.setText(cart.totalCost()+"$");
                }
            });

        tfFilter.textProperty().addListener(new ChangeListener<String>() {

            @Override
            public void changed(ObservableValue<? extends String> observable, String oldValue, String newValue) {

                showFilterMedia(newValue);
            }
        });
    }

```

```
151     @FXML
152     void changeToStoreScreen(ActionEvent event) {
153         this.controllerScreen.showStoreScreen();
154     }
155     @FXML
156     void changeToAddBookScreen(ActionEvent event) {
157         this.controllerScreen.showAddBookScreen();
158     }
159
160     @FXML
161     void changeToAddCDScreen(ActionEvent event) {
162         this.controllerScreen.showAddCDCreen();
163     }
164
165     @FXML
166     void changeToAddDVDScreen(ActionEvent event) {
167         this.controllerScreen.showAddDVDScreen();
168     }
169
170     @FXML
171     void changeToCartScreen(ActionEvent event) {
172         this.controllerScreen.showCartScreen();
173     }
174
175     void showFilterMedia(String searchString) {
176         if(searchString.isEmpty()) {
177             tblMedia.setItems(this.cart.getItemsOrdered());
178         } else {
179             if(radioBtnFilterId.isSelected()) {
180                 tblMedia.setItems(new FilteredList<Media>(this.cart.getItemsOrdered(),
181                         item-> item.getId()==Integer.parseInt(searchString)));
182             }else
183             tblMedia.setItems(new FilteredList<Media>(this.cart.getItemsOrdered(),
184                     item-> item.getTitle().contains(searchString)));
185         }
186     }
187
188     void updateButtonBar(Media media) {
189         btnRemove.setVisible(true);
190         if (media instanceof Playable) {
191             btnPlay.setVisible(true);
192         }
193         else {
194             btnPlay.setVisible(false);
```

Figure 36: CartScreenController

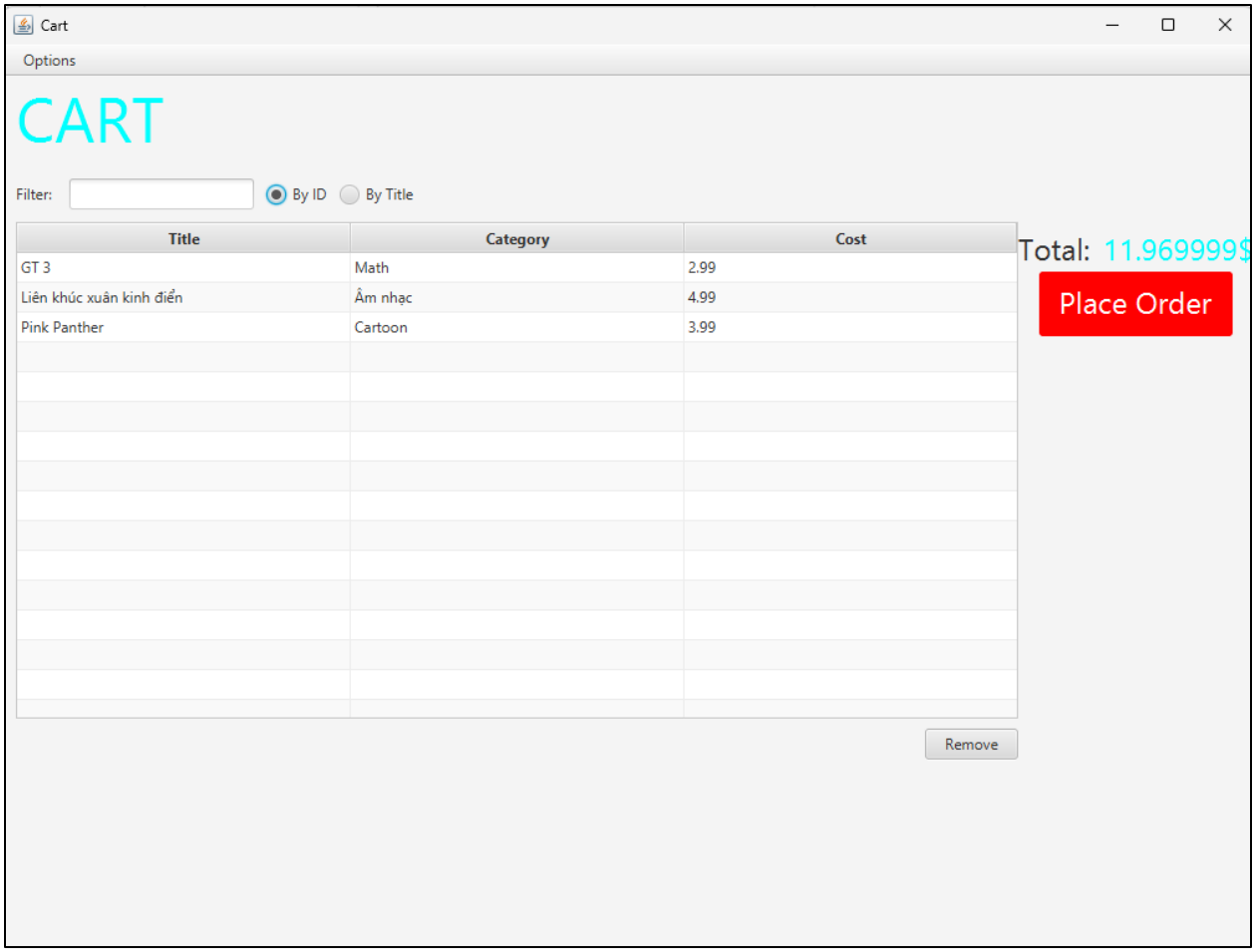

Figure 37: Cart screen in AIMS

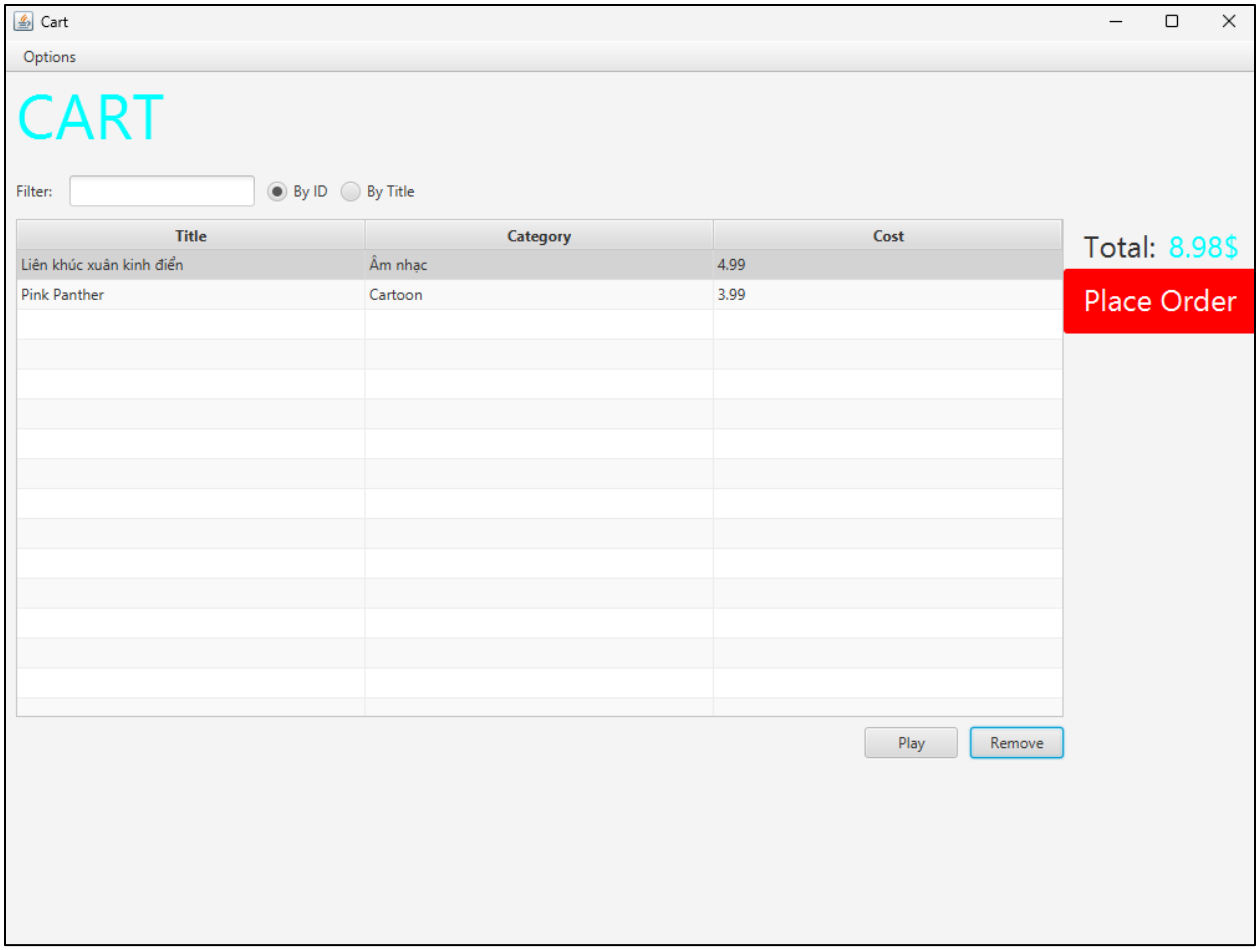

Figure 38: Remove media using Remove button

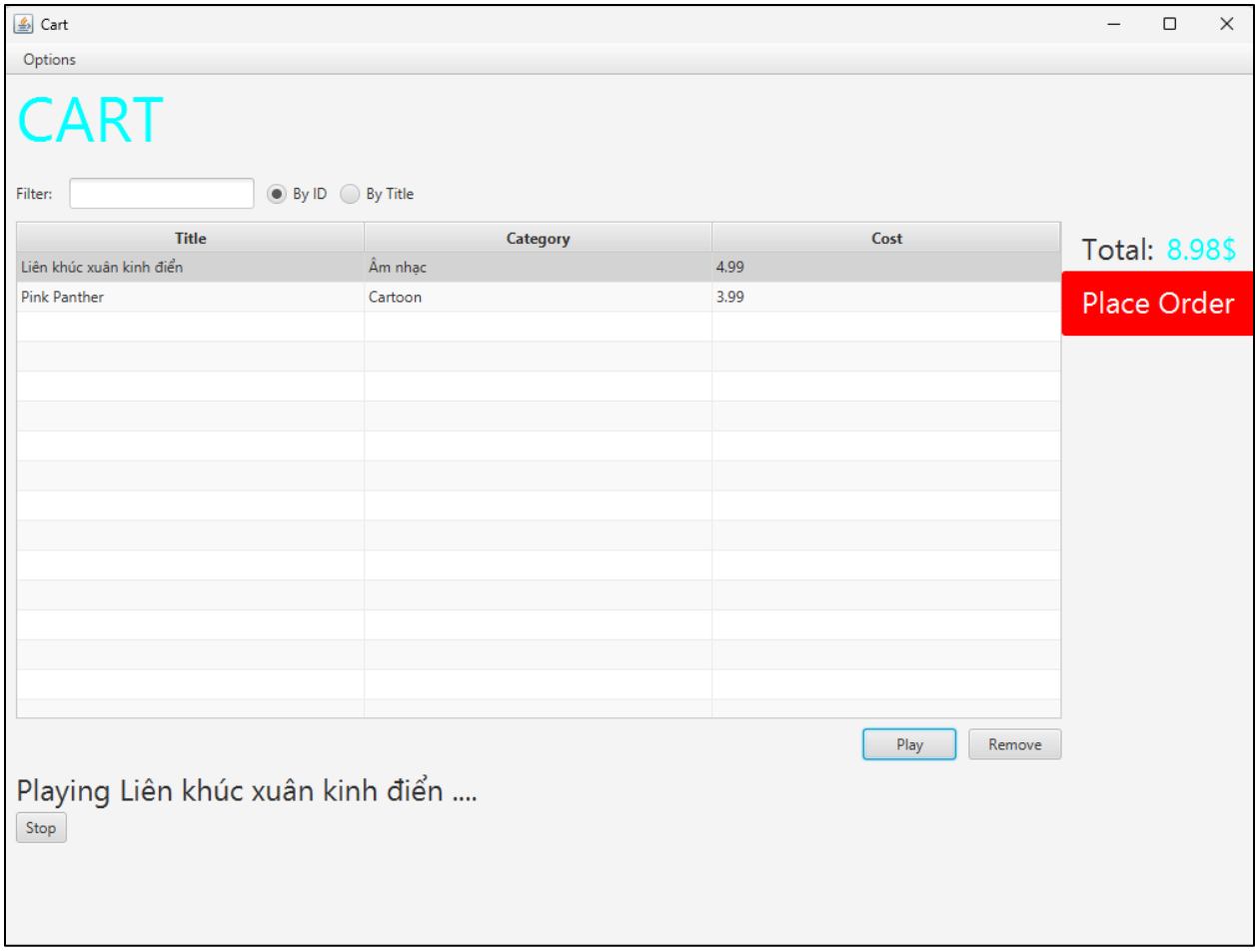

Figure 39: Playing media

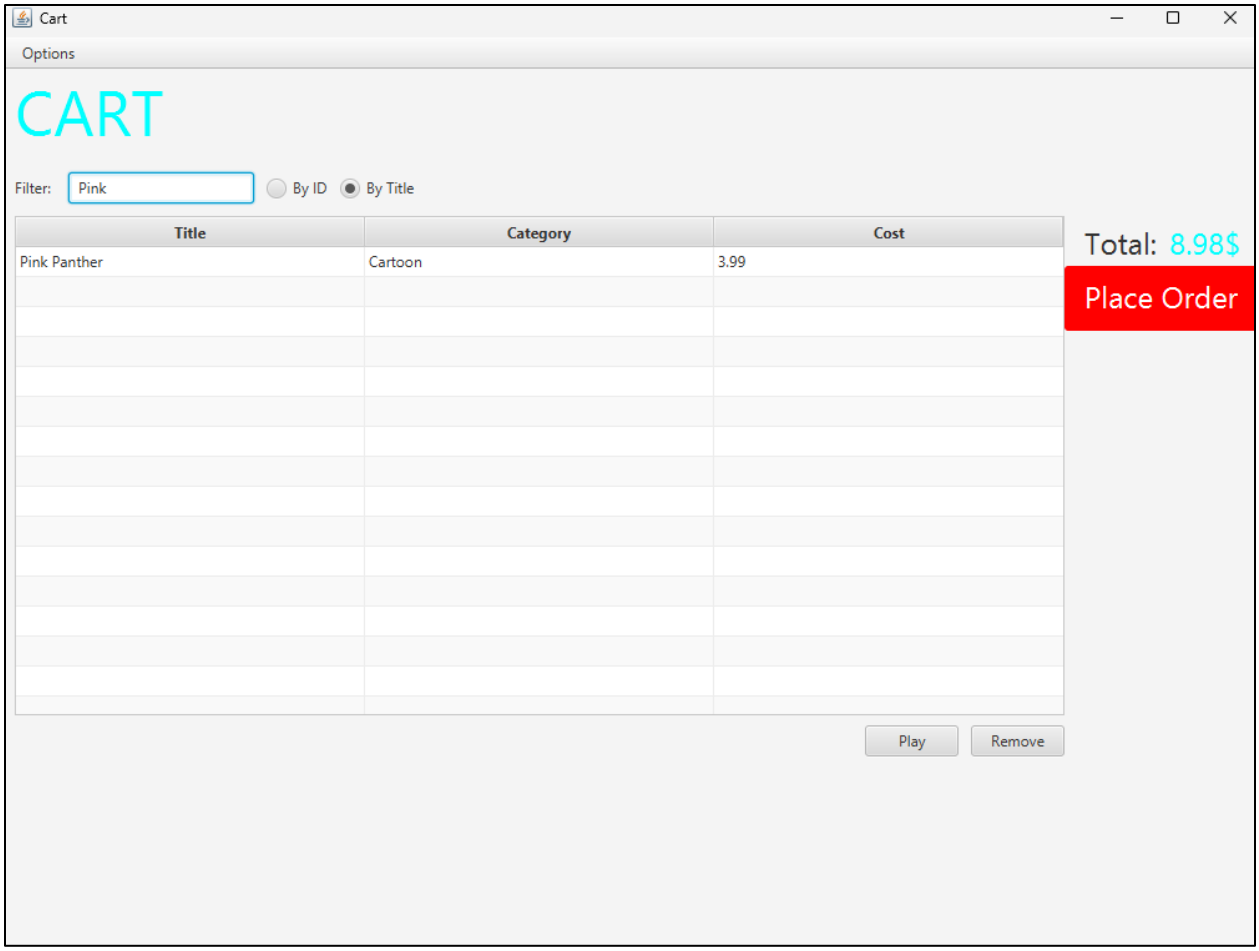

Figure 40: Filter by title

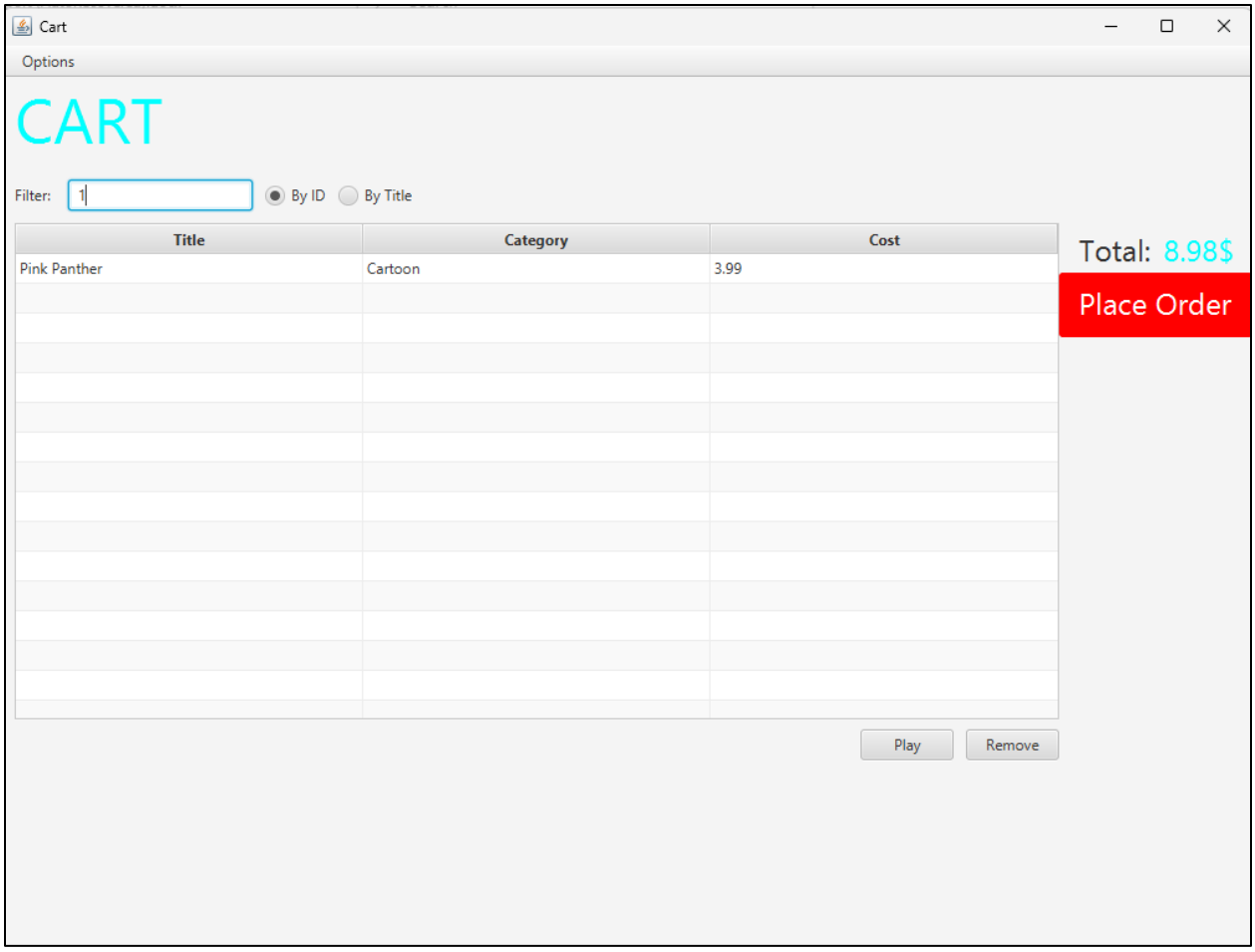

Figure 41: Filter by ID

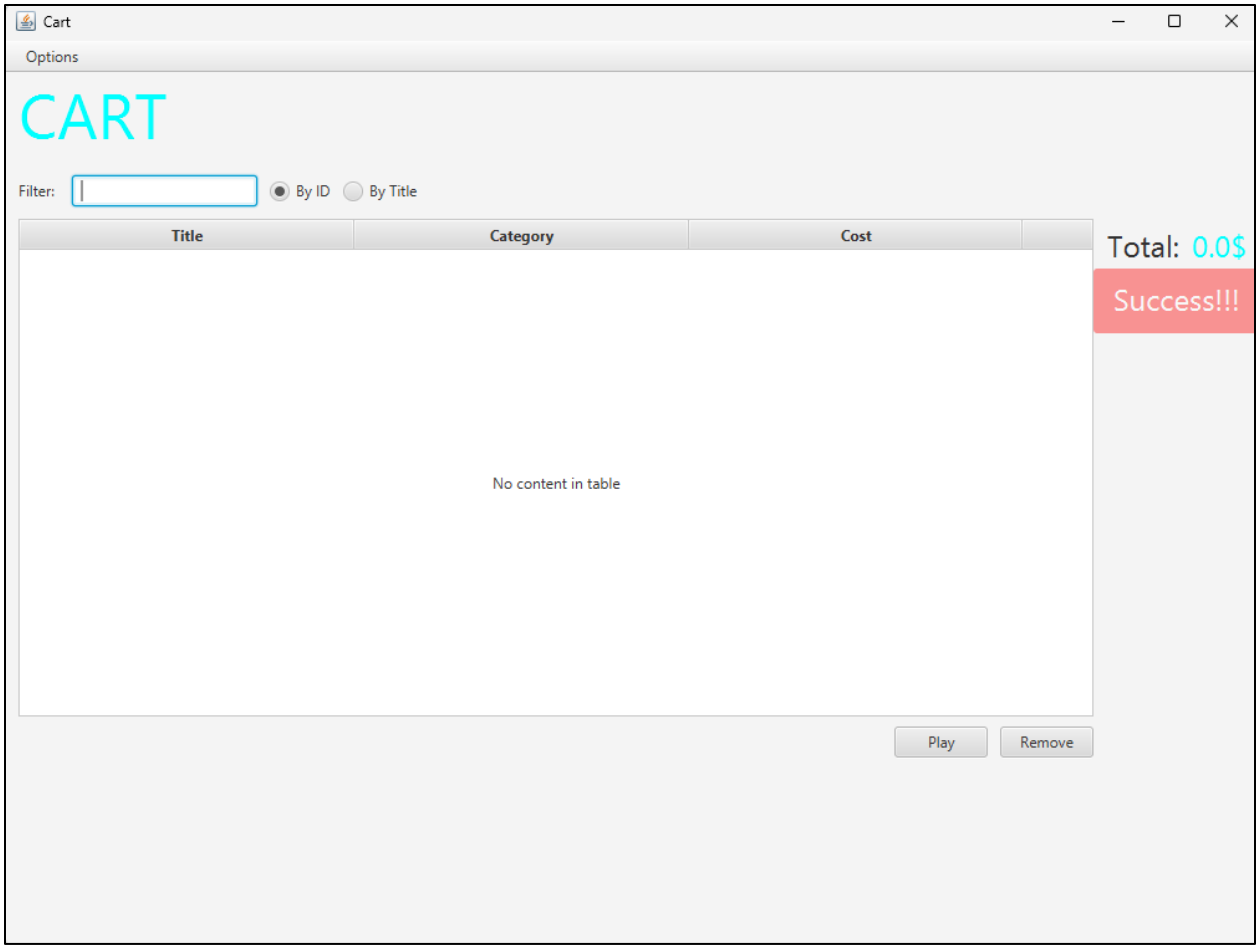

Figure 42: Clicking Order button

7. Exception handling

```
16    public void addMedia(Media media) throws LimitExceededException {
17        if ((itemsOrdered.size()) >= MAX_NUMBERS_ORDERED) {
18            throw new LimitExceededException("The cart is full");
19        }
20        else if (itemsOrdered.contains(media)) {
21            System.out.println("This is already in your order!");
22        }
23        else {
24            itemsOrdered.add(media);
25            System.out.println("Media added!");
26            }
27    }
```

Figure 43: Exception handling

8. Create a class which inherits from Exception

```java
public class PlayerException extends RuntimeException {

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

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

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

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

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

}

```

Figure 44: PlayerException

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

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

Figure 45: PlayerException in DigitalVideoDisc and Track

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

Figure 46: PlayerException in Playable

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

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

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

Figure 47: PlayerException in CompactDisc