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

BÁO CÁO THỰC HÀNH LAB 05

Học phần: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Mã học phần: IT3103
Mã lớp: 744530

Giảng viên hướng dẫn: GV Lê Thị Hoa
Sinh viên thực hiện: Tạ Hồng Phúc
MSSV: 20225906

Hà Nội, tháng 12 năm 2024

BÁO CÁO THỰC HÀNH LAB
5 LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Contents

1. Swing components

1.1 AWTAccumulator

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

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

public class AWTAccumulator extends Frame {

    private static final long serialVersionUID = 1L;

    private TextField tfInput;
    private TextField tfOutput;
    private int sum = 0;    // Sum initialize 0

    // Constructor to set up the GUI components and event handlers
    public AWTAccumulator() {
        setLayout(new GridLayout(2, 2));  // 2 rows, 2 columns layout

        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);  // Read-only
        add(tfOutput);

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

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

    private class TFInputListener implements ActionListener {
        @Override
        public void actionPerformed(ActionEvent evt) {
            int numberIn = Integer.parseInt(tfInput.getText());
            sum += numberIn;            // Add  number to sum
            tfInput.setText("");        // Clear input
            tfOutput.setText(sum + ""); // Display sum
```

Figure 1.1: Source code of AWTAccumulator

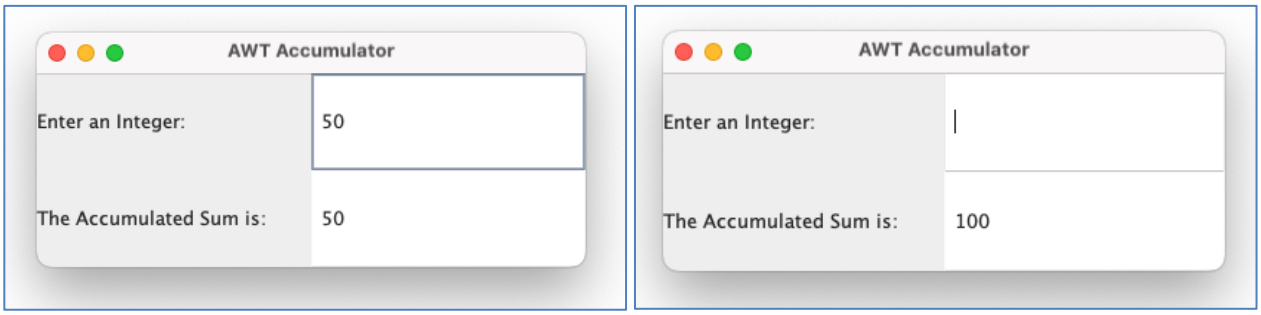


Figure 1.2: Demo of AWTAccumulator

1.2 SwingAccumulator

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

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class SwingAccumulator extends JFrame {
    private static final long serialVersionUID = 1L;
    private JTextField tfInput;    // Input TextField
    private JTextField tfOutput;   // Output TextField
    private int sum = 0;           // Accumulated sum, initialized to 0

    // Constructor to set up the GUI components and event handlers
    public SwingAccumulator() {
        Container cp = getContentPane();
        cp.setLayout(new GridLayout(2, 2));  // 2 rows, 2 columns layout

        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);  // Read-only
        cp.add(tfOutput);

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

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

    private class TFInputListener implements ActionListener {
        @Override
        public void actionPerformed(ActionEvent evt) {
            int numberIn = Integer.parseInt(tfInput.getText());
            sum += numberIn;            // Add number to sum
            tfInput.setText("");        // Clear input
            tfOutput.setText(sum + ""); // Display sum
        }
    }
}
```

Figure 1.3: Source code of SwingAccumulator

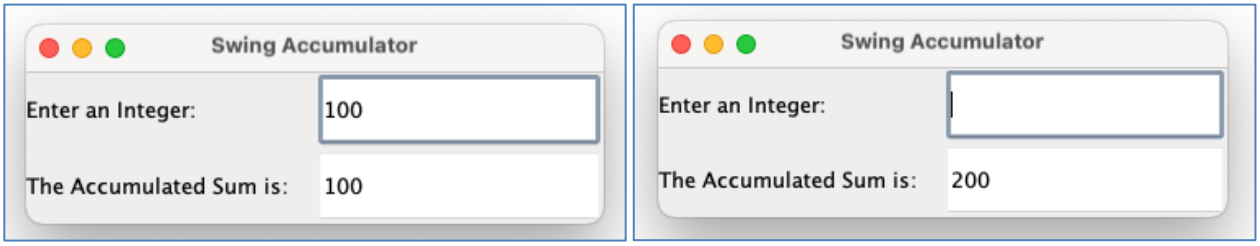


Figure 1.4: Demo of SwingAccumulator

2 Organizing Swing components with Layout Managers

2.1 Code

```
1  package hust.soict.vietnhat.swing;
2
3  import java.awt.*;
4  import java.awt.event.*;
5  import javax.swing.*;
6
7  public class NumberGrid extends JFrame {
8      private static final long serialVersionUID = 1L;
9      private JButton[] btnNumbers = new JButton[10];
10     private JButton btnDelete, btnReset;
11     private JTextField tfDisplay;
12
13     public NumberGrid() {
14         // Initialize display field
15         tfDisplay = new JTextField();
16         tfDisplay.setComponentOrientation(ComponentOrientation.RIGHT_TO_LEFT);
17
18         // Initialize panel for buttons with GridLayout
19         JPanel panelButtons = new JPanel(new GridLayout(4, 3));
20         addButtons(panelButtons);
21
22         // Set layout and add components
23         Container cp = getContentPane();
24         cp.setLayout(new BorderLayout());
25         cp.add(tfDisplay, BorderLayout.NORTH);
26         cp.add(panelButtons, BorderLayout.CENTER);
27
28         // Set frame properties
29         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
30         setTitle("Number Grid");
31         setSize(200, 200);
32         setVisible(true);
33     }
34
```

Figure 2.1: Source code of NumberGrid 1

```
35      // Add buttons to the panel
36      void addButtons(JPanel panelButtons) {
37          ButtonListener btnListener = new ButtonListener();
38
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      // Inner class for handling button actions
59      private class ButtonListener implements ActionListener {
60          @Override
61          public void actionPerformed(ActionEvent e) {
62              String button = e.getActionCommand();
63
64              if (button.charAt(0) >= '0' && button.charAt(0) <= '9') {
65                  // Append number to the display
66                  tfDisplay.setText(tfDisplay.getText() + button);
67              } else if (button.equals("DEL")) {
68                  // Handle delete (clear last character)
69                  String text = tfDisplay.getText();
70                  if (!text.isEmpty()) {
71                      tfDisplay.setText(text.substring(0, text.length() - 1));
72                  }
73              } else {
74                  // Handle clear ("C")
75                  tfDisplay.setText("");
76              }
77          }
78      }
79
80      public static void main(String[] args) {
81          new NumberGrid();
82      }
83  }
```

Figure 2.2: Source code of NumberGrid 2

2.2 Demo

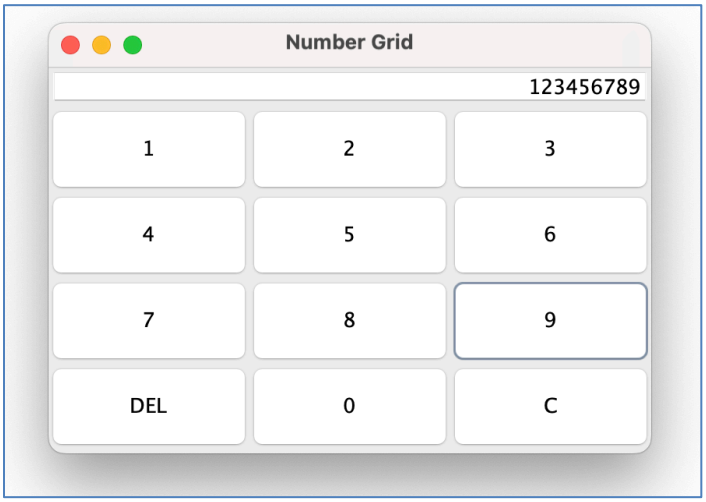


Figure 2.3: Demo number 0 – 9

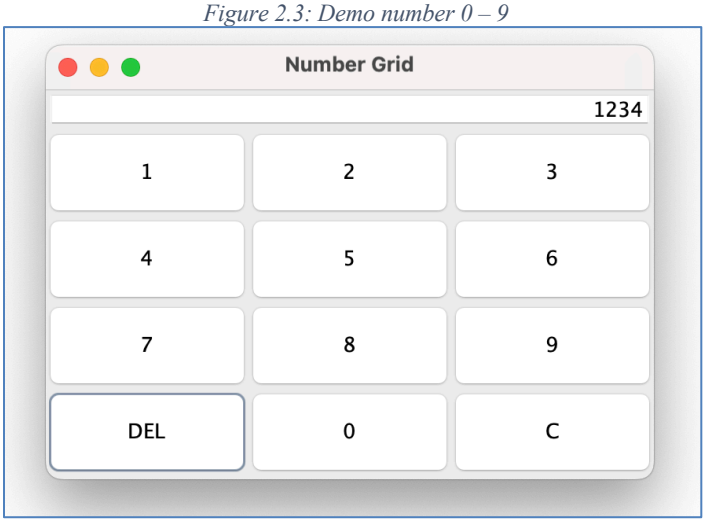


Figure 2.4: Demo DEL button

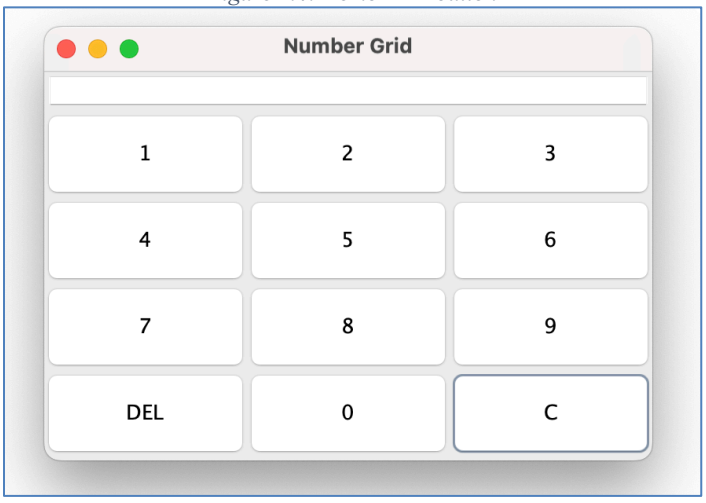


Figure 2.5: Demo C button

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 javax.swing.JFrame;
20
21 public class StoreScreen extends JFrame{
22     private static final long serialVersionUID = 1L;
23     private Store_phucth store;
24     public Cart_phucth cart;
25
26
27
28     // Constructor
29     public StoreScreen(Store_phucth store) {
30         this.store = store;
31         this.cart = new Cart_phucth(); // Initialize the cart
32         Container cp = getContentPane();
33         cp.setLayout(new BorderLayout());
34
35         cp.add(createNorth(), BorderLayout.NORTH);
36         cp.add(createCenter(), BorderLayout.CENTER);
37
38         setVisible(true);
39         setTitle("Store");
40         setSize(1024, 768);
41     }
42
43     // Method to create the NORTH component
44     JPanel createNorth() {
45         JPanel north = new JPanel();
46         north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
47         north.add(createMenuBar());
48         north.add(createHeader());
49         return north;
50     }
51
52     // Method to create the Menu Bar
53     JMenuBar createMenuBar() {
54         JMenu menu = new JMenu("Options");
55
56         JMenu smUpdateStore = new JMenu("Update Store");
57         smUpdateStore.add(new JMenuItem("Add Book"));
58         smUpdateStore.add(new JMenuItem("Add CD"));
59         smUpdateStore.add(new JMenuItem("Add DVD"));
60
61         menu.add(smUpdateStore);
62         menu.add(new JMenuItem("View store"));
63         menu.add(new JMenuItem("View cart"));
```

Figure 3.1: Class StoreScreen 1

```
61          menu.add(smUpdateStore);
62          menu.add(new JMenuItem("View store"));
63          menu.add(new JMenuItem("View cart"));
64
65          JMenuBar menuBar = new JMenuBar();
66          menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
67          menuBar.add(menu);
68
69          return menuBar;
70      }
71
72      // Method to create the Header
73      JPanel createHeader() {
74          JPanel header = new JPanel();
75          header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));
76
77          JLabel title = new JLabel("AIMS");
78          title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
79          title.setForeground(Color.CYAN);
80
81          JButton cart = new JButton("View cart");
82          cart.setPreferredSize(new Dimension(100, 50));
83          cart.setMaximumSize(new Dimension(100, 50));
84
85          header.add(Box.createRigidArea(new Dimension(10, 10)));
86          header.add(title);
87          header.add(Box.createHorizontalGlue());
88          header.add(cart);
89          header.add(Box.createRigidArea(new Dimension(10, 10)));
90
91          return header;
92      }
93
94      // Method to create the CENTER component
95      JPanel createCenter() {
96          JPanel center = new JPanel();
97          center.setLayout(new GridLayout(3, 3, 2, 2));
98
99          ArrayList<Media_phucth> mediaInStore = store.getItemsInStore_phucth();
100         for (int i = 0; i < mediaInStore.size(); i++) {
101             MediaStore cell = new MediaStore(mediaInStore.get(i));
102             center.add(cell);
103         }
104
105         return center;
106     }
```

Figure 3.2: Class StoreScreen 2

```
public static void main(String[] args) {
    // Example: Initialize Store and StoreScreen
    Store_phucth store1 = new Store_phucth();

    DigitalVideoDisc_phucth dvd1 =
            new DigitalVideoDisc_phucth("Doraemon", "Viễn tưởng", "Fujiko F. Fujio", 120, 20f);
    DigitalVideoDisc_phucth dvd2 =
            new DigitalVideoDisc_phucth("Conan", "Trinh thám", "Gosho Aoyama", 120, 50f);
    DigitalVideoDisc_phucth dvd3 =
            new DigitalVideoDisc_phucth("Shin - crayon boy", "Hài hước", "Yoshito Usui", 120, 10f);
    DigitalVideoDisc_phucth dvd4 =
            new DigitalVideoDisc_phucth("Miko", "Tình cảm", "Eriko Ono", 120, 40f);
    DigitalVideoDisc_phucth dvd5 =
            new DigitalVideoDisc_phucth("Dragonball", "Hành động", "Akira Toriyama", 120, 20f);

    CompactDisc_phucth cd1 =
            new CompactDisc_phucth("Thriller", "Pop", "Michael Jackson", 29.99f);
    CompactDisc_phucth cd2 =
            new CompactDisc_phucth("Back in Black", "Rock", "AC/DC", 24.99f);
    CompactDisc_phucth cd3 =
            new CompactDisc_phucth("The Dark Side of the Moon", "Progressive Rock", "Pink Floyd", 34.99f);

    Track_phucth track1 = new Track_phucth("Track 1", 180);
    Track_phucth track2 = new Track_phucth("Track 2", 240);
    Track_phucth track3 = new Track_phucth("Track 3", 300);
    Track_phucth track4 = new Track_phucth("Track 4", 360);
```

Figure 3.3: Class StoreScreen 3

```
    cd1.addTrack(track1);
    cd1.addTrack(track2);
    cd2.addTrack(track2);
    cd2.addTrack(track3);
    cd3.addTrack(track3);
    cd3.addTrack(track4);

    Book_phucth book1 = new Book_phucth("1984", "Dystopian", 15.99f);
    Book_phucth book2 = new Book_phucth("To Kill a Mockingbird", "Fiction", 12.49f);

    store1.addMedia_phucth(dvd1);
    store1.addMedia_phucth(dvd2);
    store1.addMedia_phucth(dvd3);
    store1.addMedia_phucth(dvd4);
    store1.addMedia_phucth(dvd5);
    store1.addMedia_phucth(cd1);
    store1.addMedia_phucth(cd2);
    store1.addMedia_phucth(cd3);
    store1.addMedia_phucth(book1);
    store1.addMedia_phucth(book2);

    new StoreScreen(store1);
```

Figure 3.4: Class StoreScreen 4

3.2 Create class MediaStore

```
// MediaStore Class
class MediaStore extends JPanel {
    private static final long serialVersionUID = 1L;
    private Media_phucth media;

    public MediaStore(Media_phucth media) {
        this.media = media;
        this.setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));

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

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

        // Button Container
        JPanel container = new JPanel();
        container.setLayout(new FlowLayout(FlowLayout.CENTER));

        // Add to Cart Button
        JButton addToCartButton = new JButton("Add to cart");
        addToCartButton.addActionListener(e -> {
            cart.addMedia_phucth(media);
            JOptionPane.showMessageDialog(null, media.getTitle() + " added to cart!");
        });
        container.add(addToCartButton);

        // Play Button for Playable items (like DVD or CD)
        if (media instanceof Playable_phucth) {
            JButton playButton = new JButton("Play");
            playButton.addActionListener(e -> showPlayDialog((Playable_phucth) media));
            container.add(playButton);
        }

        // Assemble the panel
        this.add(Box.createVerticalGlue());
        this.add(title);
        this.add(cost);
        this.add(Box.createVerticalGlue());
        this.add(container);
        this.setBorder(BorderFactory.createLineBorder(Color.BLACK));
    }
```

Figure 3.5: Class MediaStore 1

```
    // Method to display Play information in a JDialog
    private void showPlayDialog(Playable_phucth playableMedia) {
        // Format the message for Playable items
        String playMessage = "Now Playing:\n" +
                             "Title: " + ((Media_phucth) playableMedia).getTitle() + "\n" +
                             "Length: " + ((Disc_phucth) playableMedia).getLength() + " minutes";

        // Display the message in a JDialog
        JDialog playDialog = new JDialog((JFrame) SwingUtilities.getWindowAncestor(this), "Play Media", true);
        playDialog.setLayout(new BorderLayout());

        // Text area for the message
        JTextArea messageArea = new JTextArea(playMessage);
        messageArea.setFont(new Font("Arial", Font.PLAIN, 16));
        messageArea.setEditable(false);

        playDialog.add(new JScrollPane(messageArea), BorderLayout.CENTER);

        // Close Button
        JButton closeButton = new JButton("Close");
        closeButton.addActionListener(e -> playDialog.dispose());
        playDialog.add(closeButton, BorderLayout.SOUTH);

        playDialog.setSize(400, 200);
        playDialog.setLocationRelativeTo(this);
        playDialog.setVisible(true);
    }
}
```

Figure 3.6: Class MediaStore 2

3.3 Demo

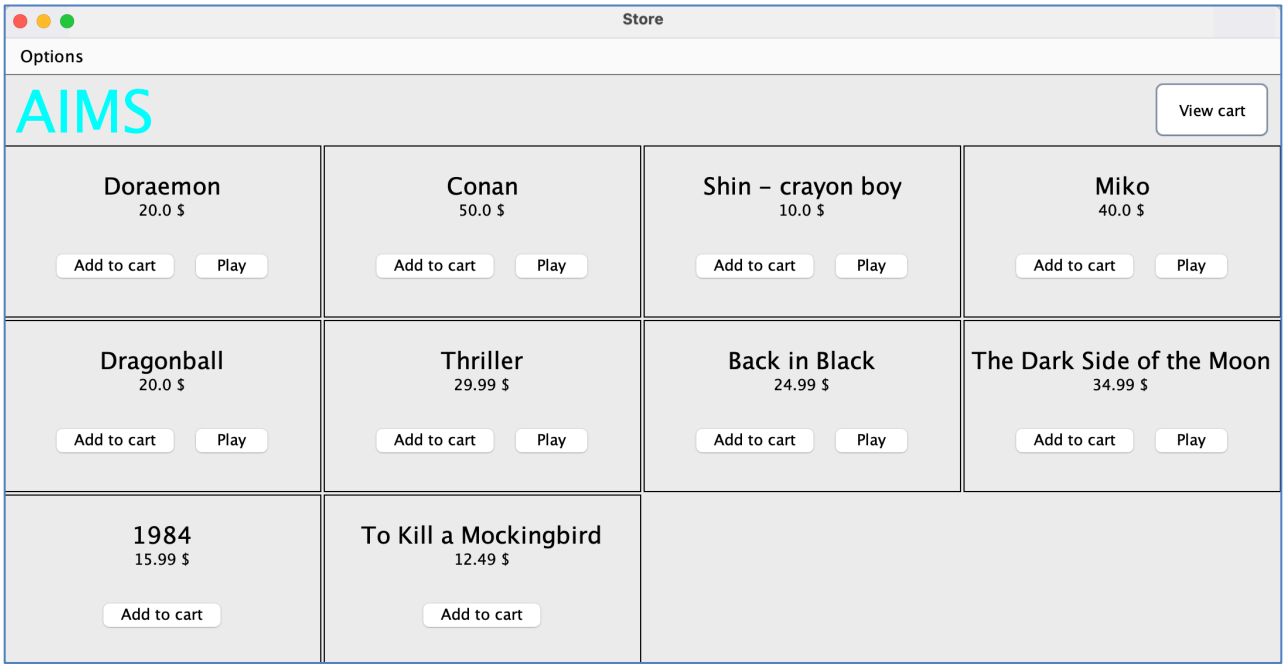


Figure 3.7 Demo StoreScreen

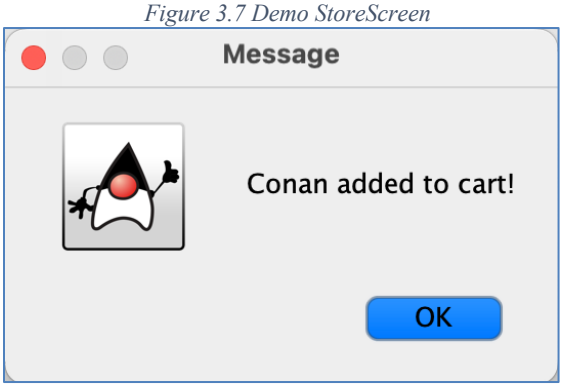


Figure 3.8 Demo Add to cart button

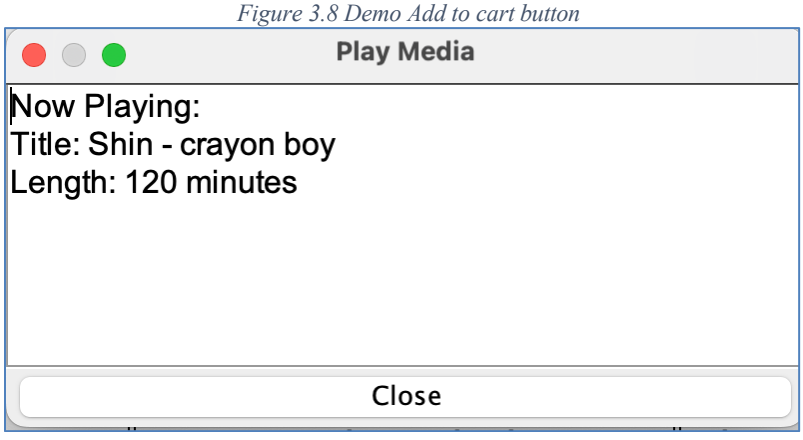


Figure 3.9 Demo Play button

4 JavaFX API

4.1 Create class Painter

```java
package hust.soict.hedspi.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/hedspi/javafx/Painter.fxml"));

        Scene scene = new Scene(root);
        stage.setTitle("Painter");
        stage.setScene(scene);
        stage.show();
    }
    public static void main(String[] args) {
        System.out.println("HI");
        launch(args);
    }
}
```

Figure 4.1: Class Painter

4.2 Create Painter.fxml

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

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

 Bind to grammar/schema...
<BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity"
            prefHeight="480.0" prefWidth="640.0" xmlns="http://javafx.com/javafx/19" xmlns:fx="http://javafx.com/fxml/1"
            fx:controller="hust.soict.hedspi.javafx.PainterController">
   <padding>
      <Insets bottom="8.0" left="8.0" right="8.0" top="8.0" />
   </padding>
   <left>
      <VBox maxHeight="1.7976931348623157E308" prefHeight="349.0" prefWidth="101.0" spacing="8.0" BorderPane.alignment="CENTER">
         <BorderPane.margin>
            <Insets right="8.0" />
         </BorderPane.margin>
         <children>
            <TitledPane animated="false" prefHeight="97.0" prefWidth="101.0" text="Tools">
              <content>
                <AnchorPane minHeight="0.0" minWidth="0.0" prefHeight="0.0" prefWidth="99.0">
                     <children>
                        <RadioButton fx:id="pen" layoutX="16.0" layoutY="14.0" mnemonicParsing="false" selected="true" text="Pen">
                           <toggleGroup>
                              <ToggleGroup fx:id="Tools" />
                           </toggleGroup>
                        </RadioButton>
                        <RadioButton fx:id="eraser" layoutX="16.0" layoutY="39.0" mnemonicParsing="false" text="Eraser" toggleGroup="$Tools" />
                     </children>
                  </AnchorPane>
              </content>
            </TitledPane>
            <Button maxWidth="1.7976931348623157E308" mnemonicParsing="false" onAction="#clearButtonPressed" prefWidth="101.0" text="Clear" />
         </children>
      </VBox>
   </left>
   <center>
      <Pane fx:id="drawingAreaPane" onMouseDragged="#drawingAreaMouseDragged" prefHeight="200.0" prefWidth="200.0"
            style="-fx-background-color: white;" BorderPane.alignment="CENTER" />
   </center>
</BorderPane>

```

Figure 4.2: Painter.fxml 1

4.3 Create class PainterController

```
1  package hust.soict.hedspi.javafx;
2
3  import java.net.URL;
13
14 public class PainterController {
15
16     @FXML
17     private RadioButton eraser;
18
19     @FXML
20     private RadioButton pen;
21
22     @FXML
23     private ResourceBundle resources;
24
25     @FXML
26     private URL location;
27
28     @FXML
29     private Pane drawingAreaPane;
30
31     @FXML
32     void clearButtonPressed(ActionEvent event) {
33           drawingAreaPane.getChildren().clear();
34     }
35
36     @FXML
37     void drawingAreaMouseDragged(MouseEvent event) {
38       Rectangle clipArea = new Rectangle(0, 0, drawingAreaPane.getWidth(), drawingAreaPane.getHeight());
39       drawingAreaPane.setClip(clipArea);
40       Color inkColor = Color.BLACK;
41       if (eraser.isSelected()) {
42         inkColor = Color.WHITE;
43       }
44       Circle newCircle = new Circle(event.getX(), event.getY(), 4, inkColor);
45       drawingAreaPane.getChildren().add(newCircle);
46     }
47
48     @FXML
49     void initialize() {
50         assert drawingAreaPane != null : "Check your FXML file 'Painter.fxml'.";
51     }
52
53 }
```

Figure 4.4: PainterController

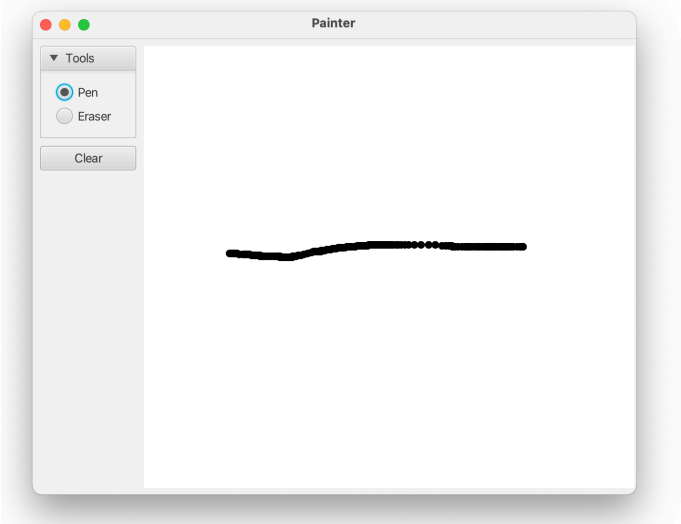


Figure 4.5: Use Pen

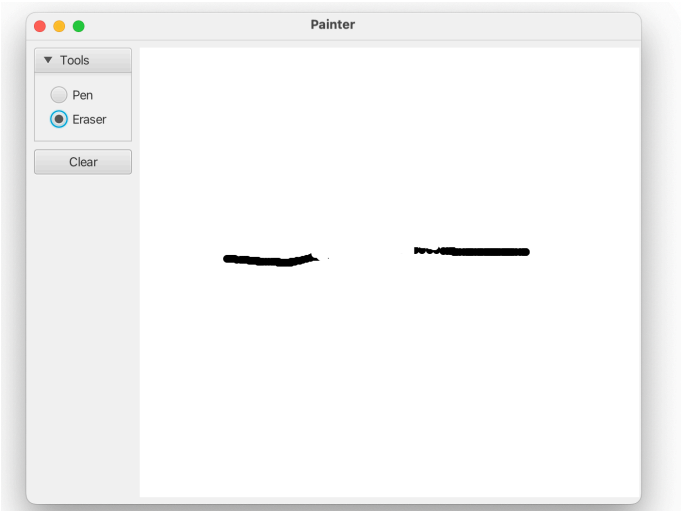


Figure 4.6: Use Eraser

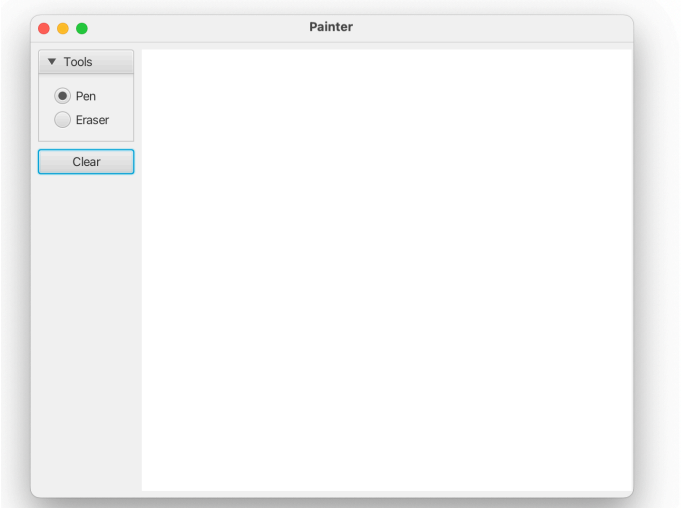


Figure 4.7: Clear button

5 View Cart Screen

5.1 Create cart.fxml

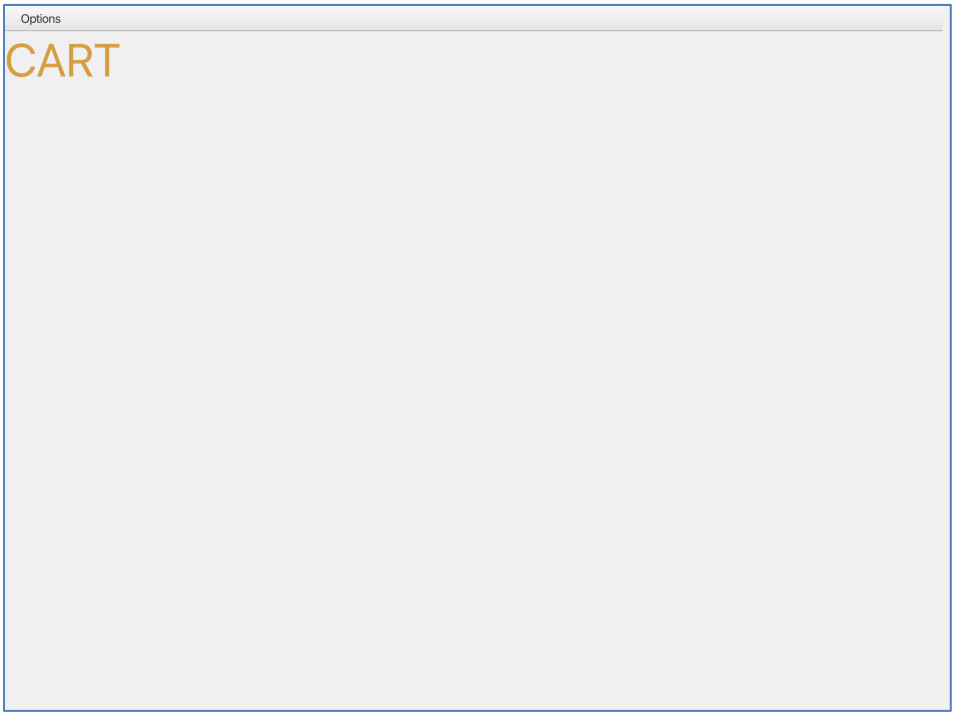


Figure 5.1: Cart.fxml 1

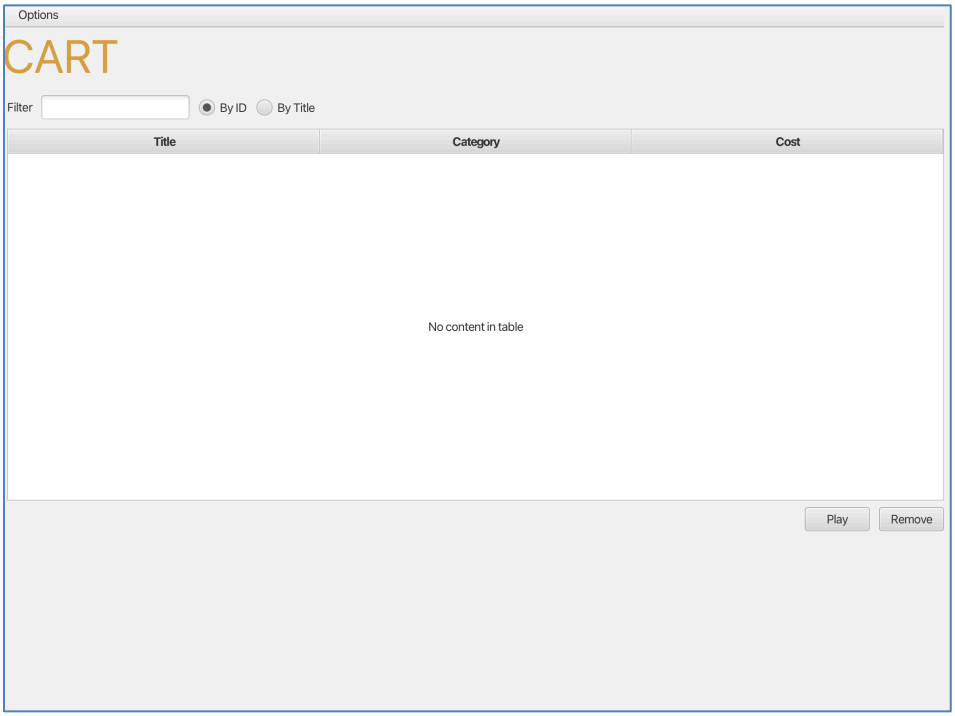


Figure 5.2: Cart.fxml 2

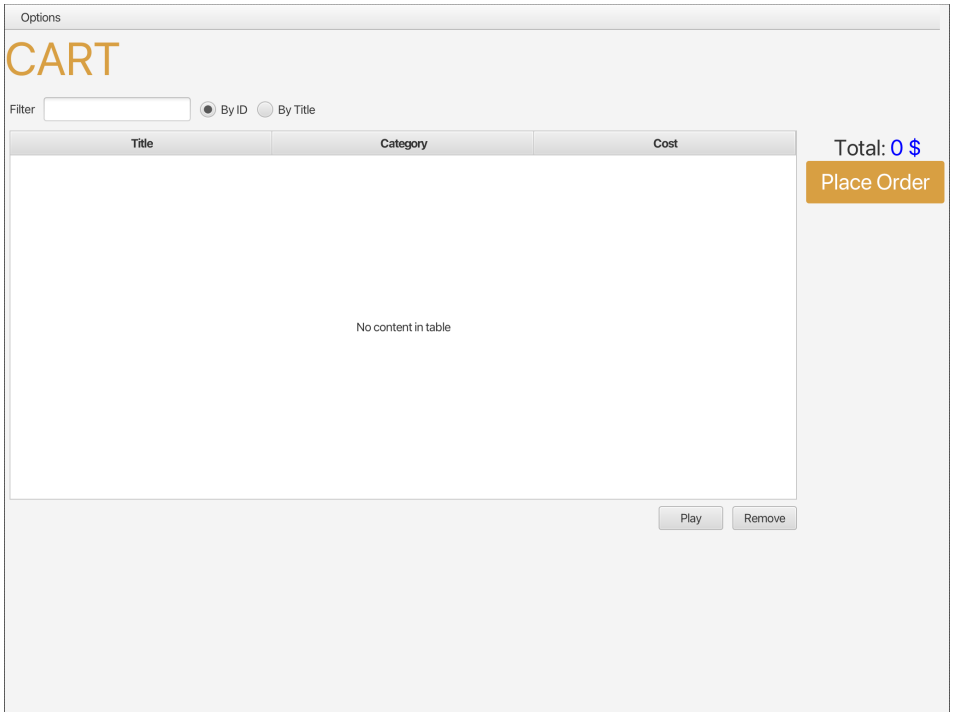


Figure 5.3: Cart.fxml 3

5.2 Create class CartScreen

```
1  package hust.soict.dsai.aims.cart;
2
3  import java.io.IOException;
17
18 public class CartScreen extends JFrame{
19     private static final long serialVersionUID = 1L;
20     private Cart_phucth cart;
21     public CartScreen(Cart_phucth cart) {
22         super();
23         this.cart = cart;
24
25         JFXPanel fxPanel = new JFXPanel();
26         this.add(fxPanel);
27         this.setTitle("Cart");
28         this.setVisible(true);
29         this.setSize(1024, 768);
30
31         Platform.runLater(new Runnable() {
32             @Override
33             public void run() {
34                 try {
35                     FXMLLoader loader = new FXMLLoader(getClass().getResource("/hust/soict/dsai/aims/cart/cart.fxml"));
36                     CartScreenController controller = new CartScreenController(cart);
37                     loader.setController(controller);
38                     Parent root = loader.load();
39                     fxPanel.setScene(new Scene(root));
40                 } catch (IOException e) {
41                     e.printStackTrace();
42                 }
43             }
44         });
45     }
```

Figure 5.4: CartScreen class

5.3 Create class CartScreenController

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

import lab04.Cart_phucth;

public class CartScreenController {
    private Cart_phucth cart;

    @FXML
    private TableColumn<Media_phucth, String> colMediaCategory;

    @FXML
    private TableColumn<Media_phucth, Float> colMediaCost;

    @FXML
    private TableColumn<Media_phucth, String> colMediaTitle;

    @FXML
    private ToggleGroup filterCategory;

    @FXML
    private TableView<Media_phucth> tblMedia;

    public CartScreenController (Cart_phucth cart) {
        super();
        this.cart = cart;
    }

```

Figure 5.5: CartScreenController 1

```java
@FXML
private void initialize() {
    colMediaTitle.setCellValueFactory(new PropertyValueFactory<Media_phucth, String>("Title"));
    colMediaCategory.setCellValueFactory(new PropertyValueFactory<Media_phucth, String>("Category"));
    colMediaCost.setCellValueFactory(new PropertyValueFactory<Media_phucth, Float>("Cost"));
    tblMedia.setItems(this.cart.getItemsOrdered());

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

Figure 5.6: CartScreenController 2

5.4 Demo

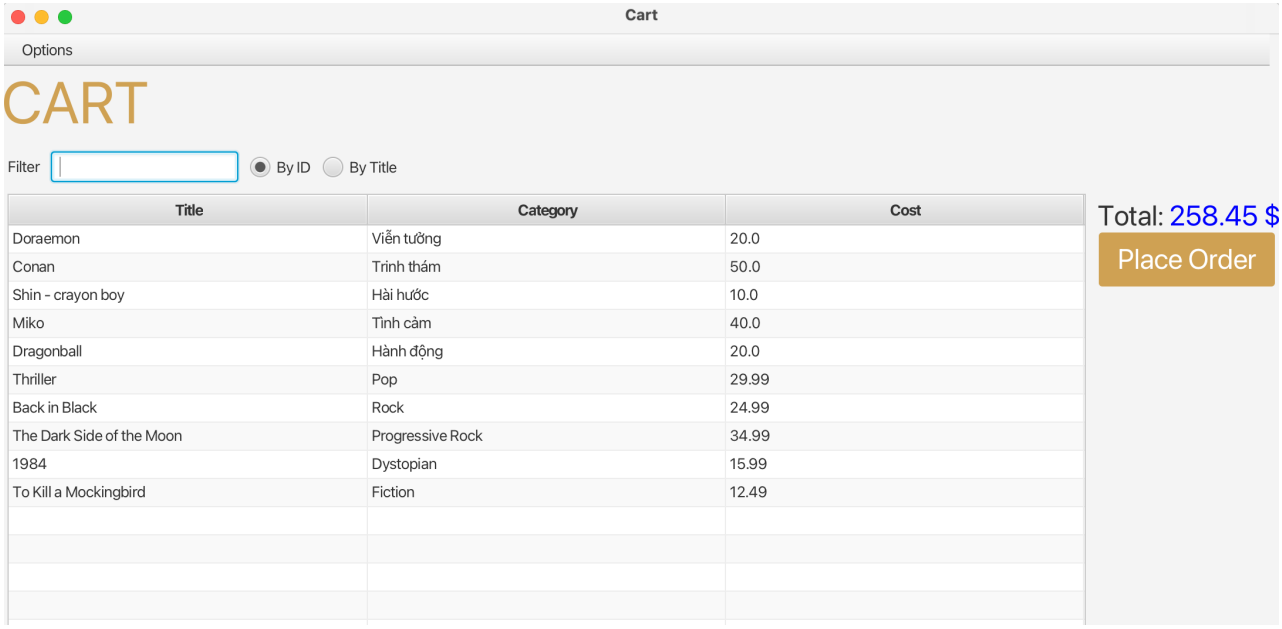


Figure 5.7: Demo CartScreen

6 Updating buttons based on selected item in TableView – ChangeListener

6.1 Edit class CartScreenController

```
// New method to calculate and update total price
private void updateTotalPrice() {
    float totalPrice = 0f;
    for (Media_phucth media : cart.getItemsOrdered()) {
        totalPrice += media.getCost();
    }

    // Update the total price label
    totalPriceLabel.setText(String.format("%.2f $", totalPrice));
}

void updateButtonBar(Media_phucth media) {
    btnRemove.setVisible(true);
    if(media instanceof Playable_phucth) btnPlay.setVisible(true);
    else btnPlay.setVisible(false);
}

void showFiterMedia(String s) {
    cart.searchByTitle(s);
}

 @FXML
void btnRemovePressed(ActionEvent event) {
     Media_phucth media = tblMedia.getSelectionModel().getSelectedItem();
    cart.removeMedia_phucth(media);
}
```

Figure 6.1: CartScreenController 1

```
@FXML
private void initialize() {
    colMediaTitle.setCellValueFactory(new PropertyValueFactory<Media_phucth, String>("Title"));
    colMediaCategory.setCellValueFactory(new PropertyValueFactory<Media_phucth, String>("Category"));
    colMediaCost.setCellValueFactory(new PropertyValueFactory<Media_phucth, Float>("Cost"));
    tblMedia.setItems(this.cart.getItemsOrdered());

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

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

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

 // Update total price initially and whenever the cart changes
    updateTotalPrice();

    // Listen for changes in the cart
    cart.getItemsOrdered().addListener(new ListChangeListener<Media_phucth>() {
        @Override
        public void onChanged(Change<? extends Media_phucth> c) {
            updateTotalPrice();
        }
    });
 // Add action for Place Order button
    btnPlaceOrder.setOnAction(this::handlePlaceOrder);
}
```

Figure 6.2: CartScreenController 2

6.2 Demo

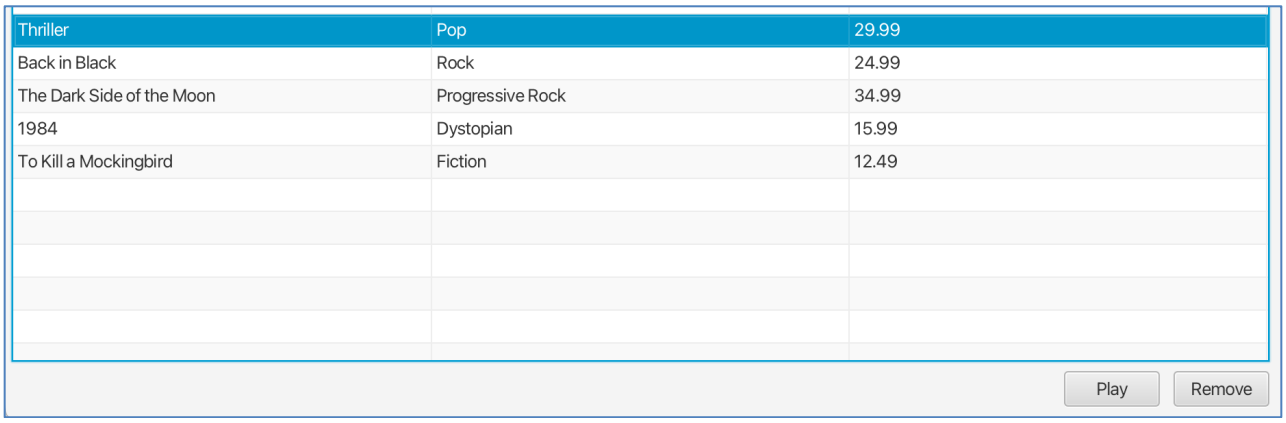


Figure 6.3: Demo media playable

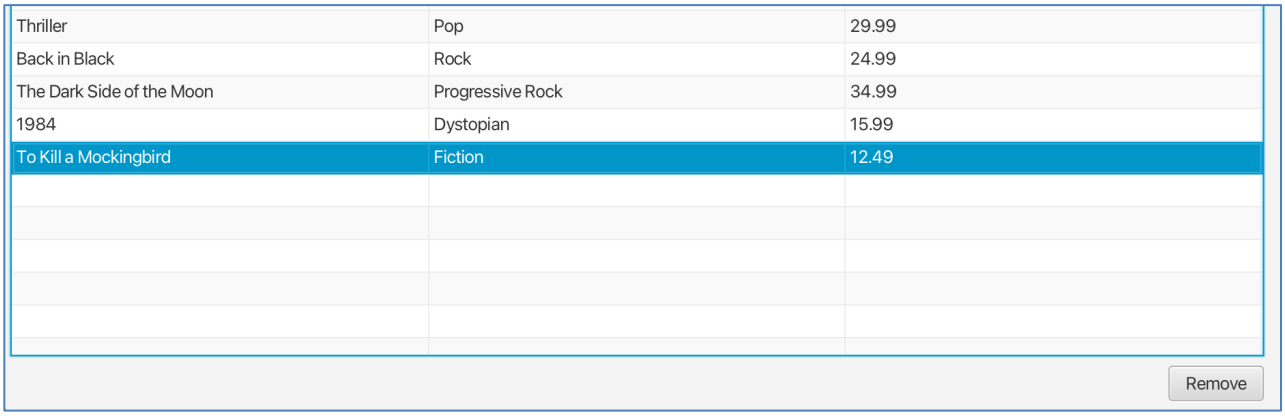


Figure 6.4: Demo media unplayable

7 Deleting a media

7.1 Code

```
 @FXML
void btnRemovePressed(ActionEvent event) {
     Media_phucth media = tblMedia.getSelectionModel().getSelectedItem();
    cart.removeMedia_phucth(media);
}
```

Figure 7.1: btnRemovePressed Method

7.2 Demo

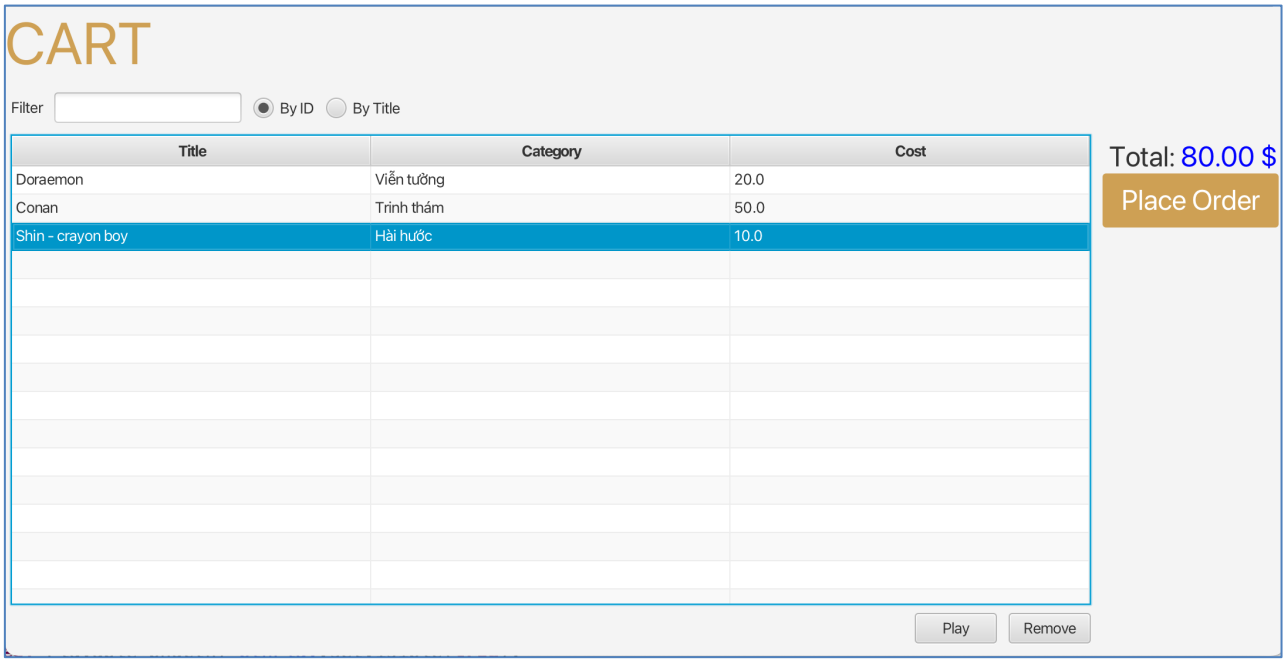


Figure 7.2: button Remove

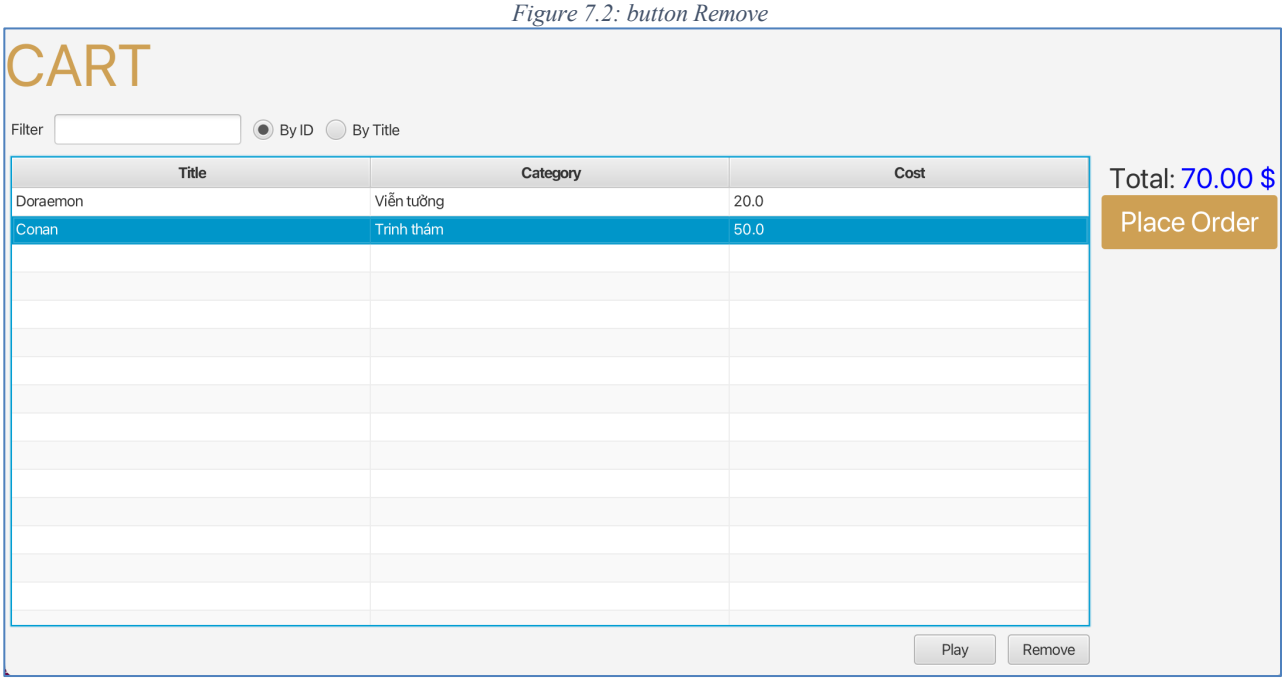


Figure 7.3: button Remove

8 Complete the Aims GUI application

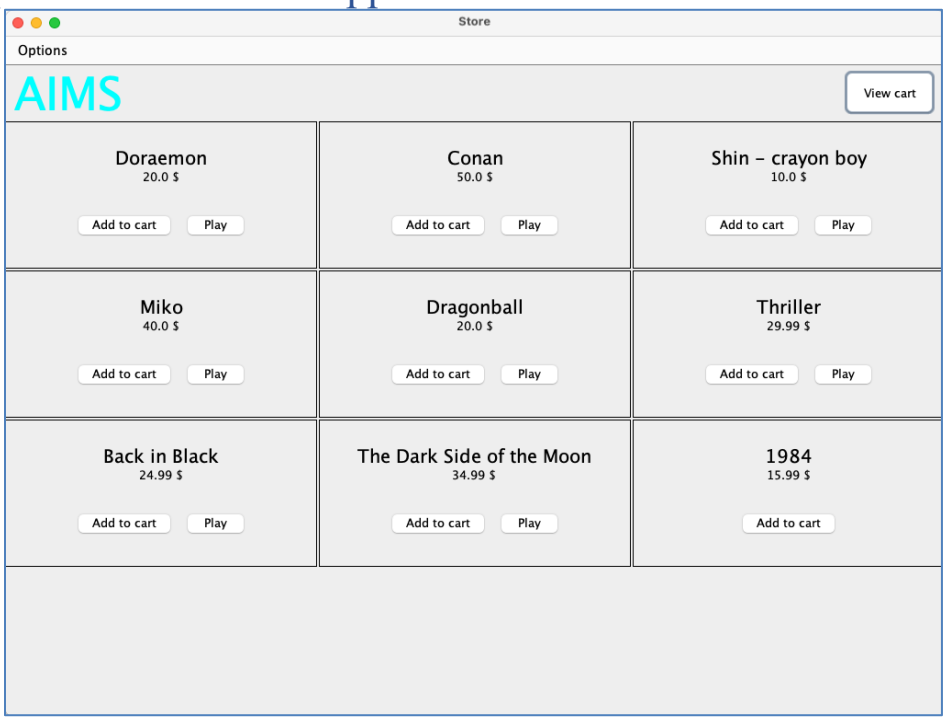


Figure 8.1: Store before add book

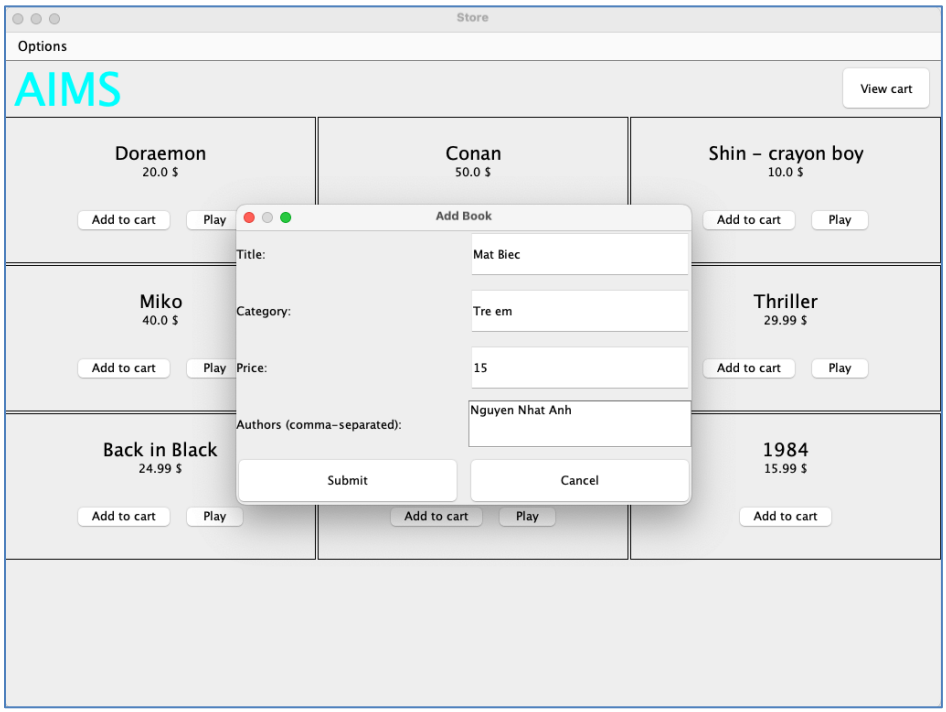


Figure 8.2: Add book

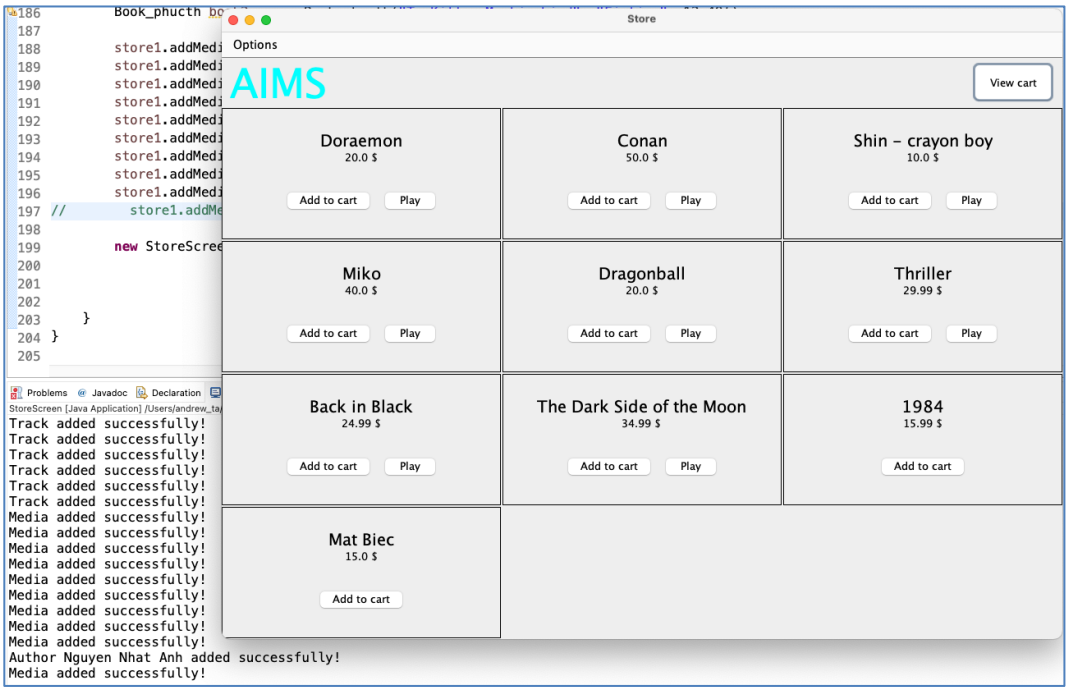


Figure 8.3: Store after add book

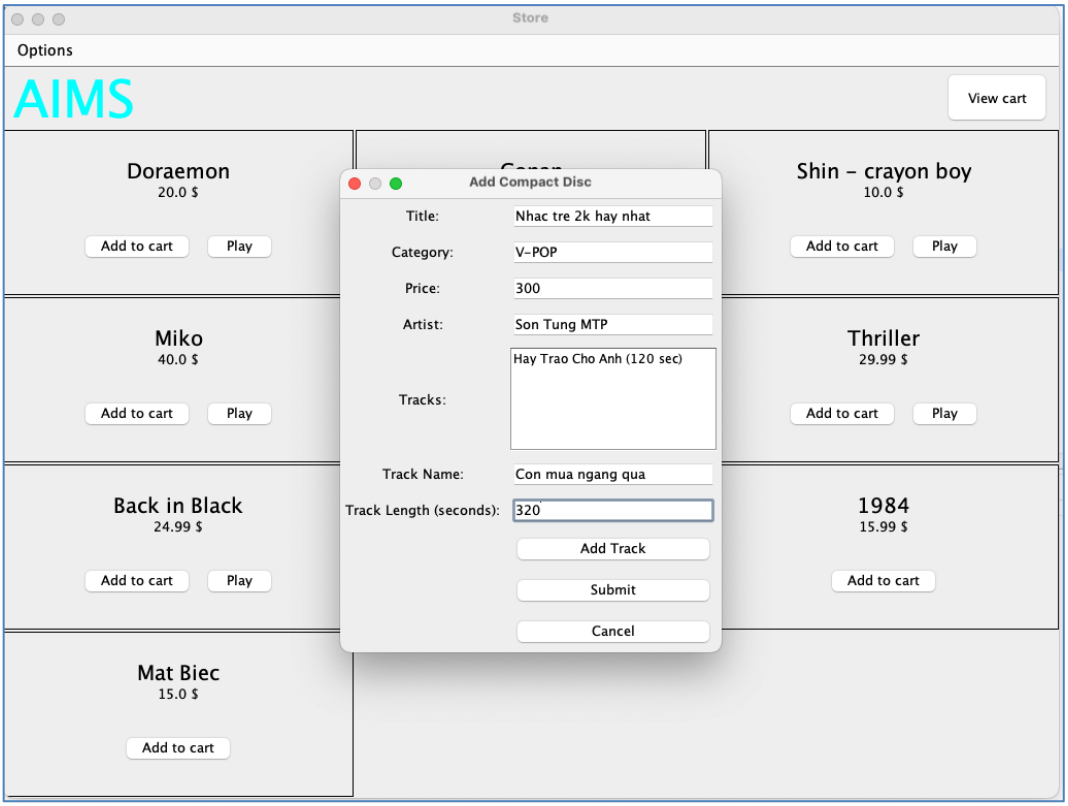


Figure 8.4: Add CD

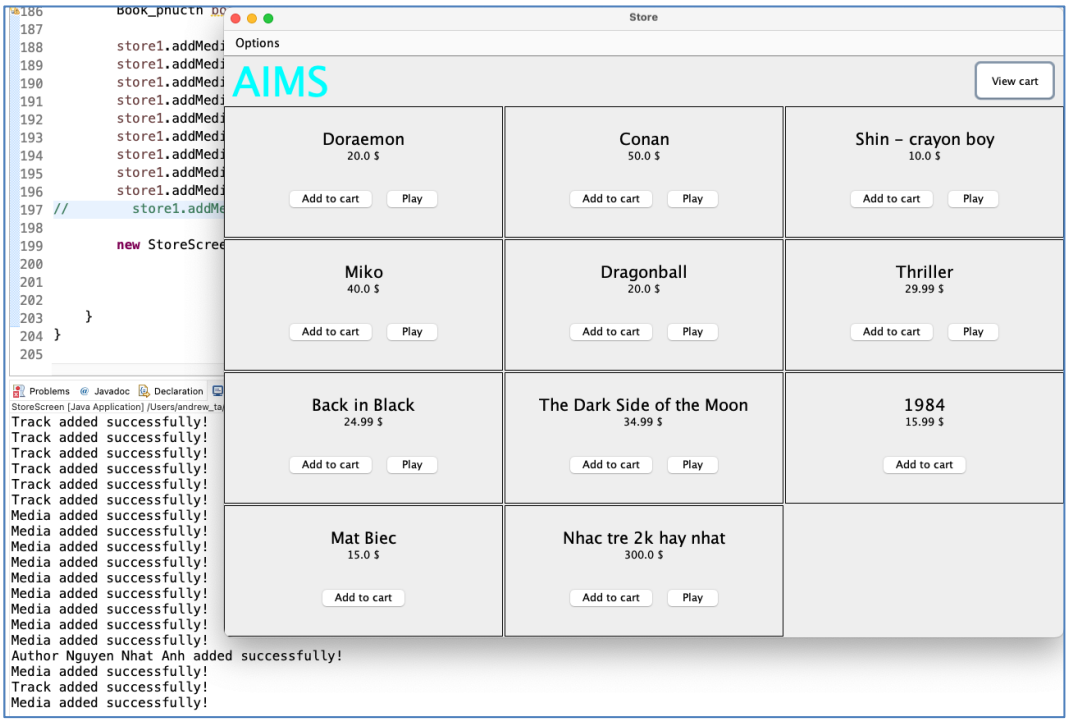


Figure 8.5: Store after add CD

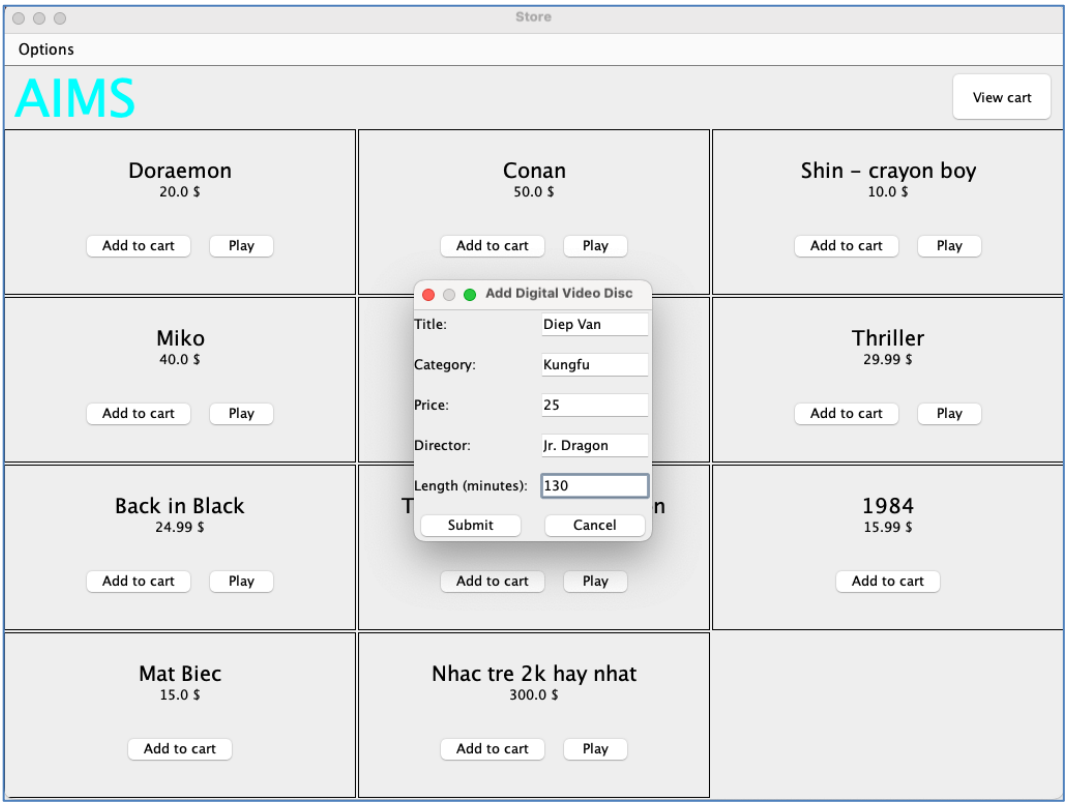


Figure 8.6 Add DVD

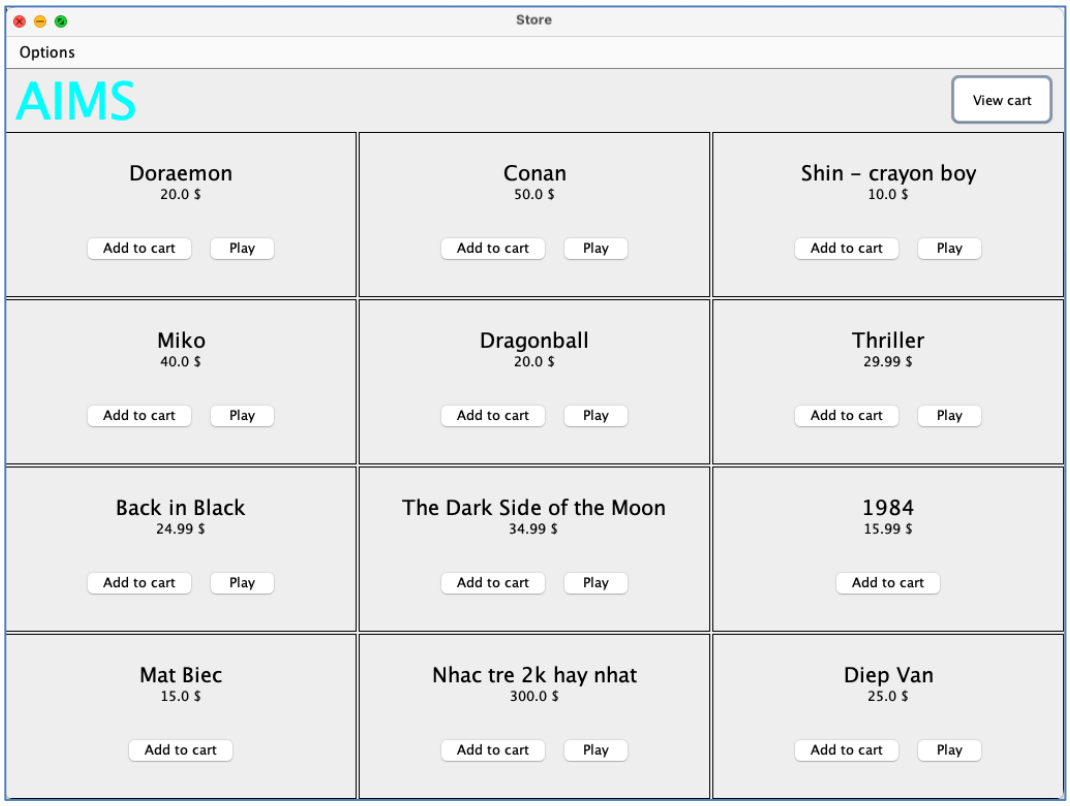


Figure 8.7: Store after add DVD

Figure 8.8: Cart

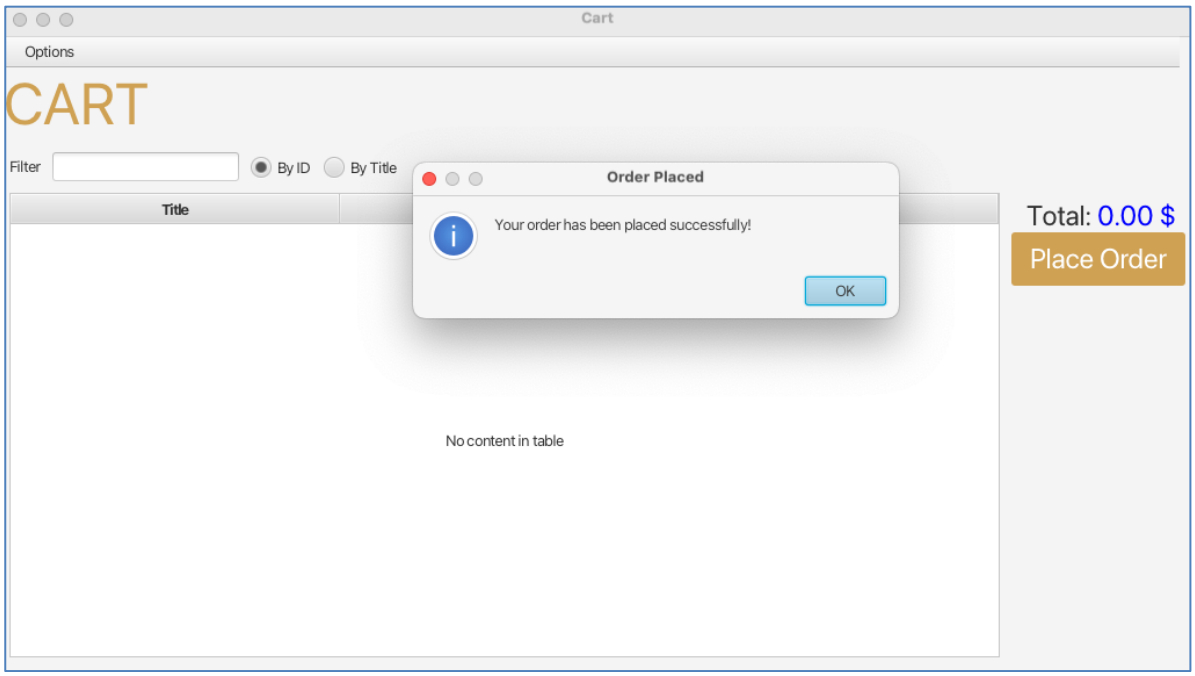


Figure 8.9: Place Order

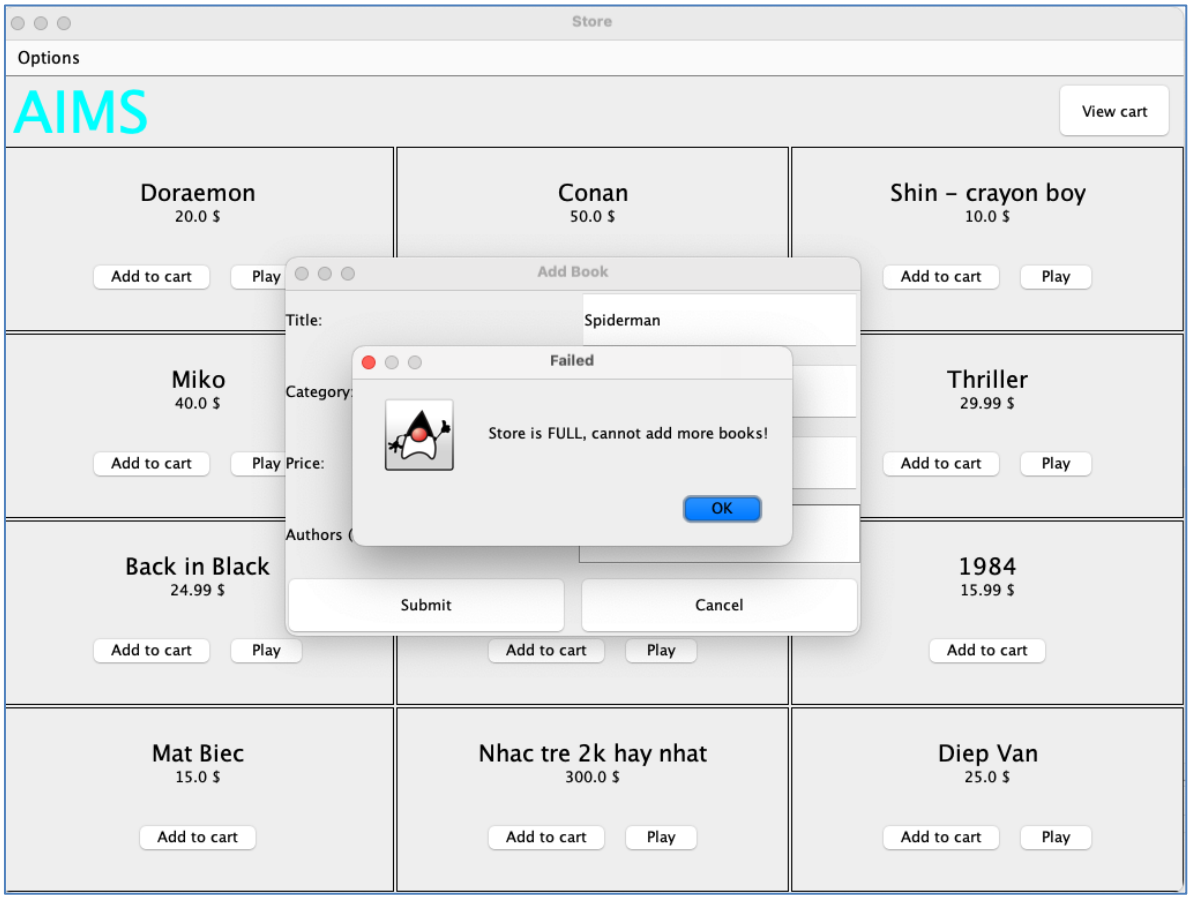
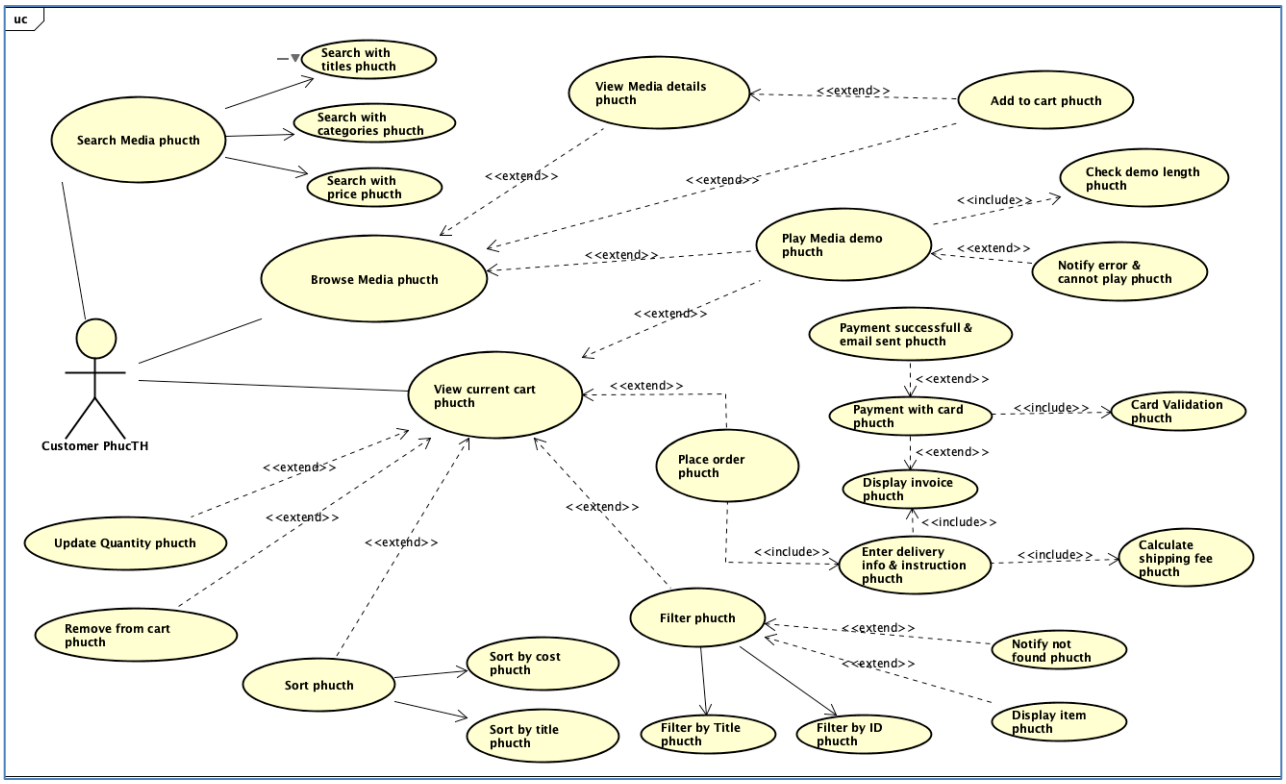
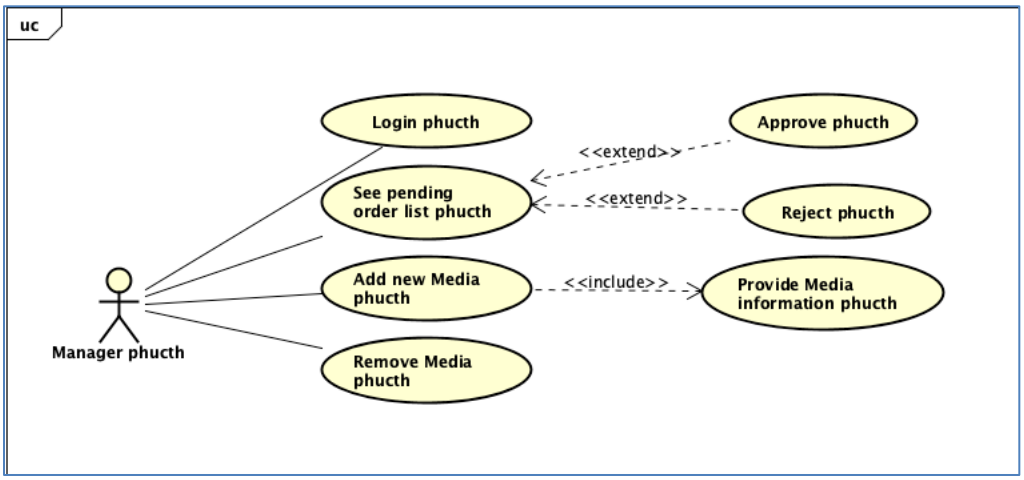


Figure 8.10: Exception

9 Use Case Diagram





10 Class Diagram