

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 05

Họ và tên sv: **Hồ Tuấn Anh**

Lớp: **K67-ITEP01**

GVHD: **Lê Thị Hoa**

TA: **Đặng Mạnh Cường**

Hà Nội 12/2024

BÁO CÁO THỰC HÀNH LAB 5

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

Contents

1.	Swing components.....	4
1.1	AWTAccumulator	4
1.2	SwingAccumulator	5
2	Organizing Swing components with Layout Managers	6
2.1	Code.....	6
2.2	Demo	8
3	Create a graphical user interface for AIMS with Swing	9
3.1	Create class StoreScreen.....	9
3.2	Create class MediaStore	13
3.3	Demo	14
4	JavaFX API.....	16
4.1	Create class Painter	16
4.2	Create Painter.fxml.....	16
4.3	Create class PainterController	17
5	View Cart Screen.....	19
5.1	Create cart.fxml.....	19
5.2	Create class CartScreen	20
5.3	Create class CartScreenController	21
5.4	Demo	22
6	Updating buttons based on selected item in TableView – ChangeListener.....	22
6.1	Edit class CartScreenController	22
6.2	Demo	23
7	Deleting a media	24
7.1	Code.....	24
7.2	Demo	25
8	Complete the Aims GUI application	26
9	Use case Diagram.....	30
10	Class Diagram.....	31

Figure 1.1: Source code of AWTAccumulator	4
Figure 1.2: Demo of AWTAccumulator	5
Figure 1.3: Source code of SwingAccumulator	5
Figure 1.4: Demo of SwingAccumulator	6
Figure 2.1: Source code of NumberGrid 1	6
Figure 2.2: Source code of NumberGrid 2	7
Figure 2.3: Demo buttons 0-9	8
Figure 2.4: Demo DEL button	8
Figure 2.5: Demo C button	8
Figure 3.1: Class StoreScreen 1	9
Figure 3.2: Class StoreScreen 2	10
Figure 3.3: Class StoreScreen 3	10
Figure 3.4: Class StoreScreen 4	11
Figure 3.5: Class StoreScreen 5	11
Figure 3.6: Class StoreScreen 6	12
Figure 3.7: Class MediaStore 1	13
Figure 3.8: Class MediaStore 2	13
Figure 3.9: Class MediaStore 3	14
Figure 3.10: StoreScreen	14
Figure 3.11 Demo Add to cart button	15
Figure 3.12 Demo Play button	15
Figure 3.13 Demo View cart button	15
Figure 4.1: Class Painter	16
Figure 4.2: Painter.fxml 1	16
Figure 4.3: Painter.fxml 2	17
Figure 4.4: PainterController	17
Figure 4.5: Use Pen	18
Figure 4.6: Use Eraser	18
Figure 4.7: Clear button	18
Figure 5.1: Cart.fxml 1	19
Figure 5.2: Cart.fxml 2	19
Figure 5.3: Cart.fxml 3	20
Figure 5.4: CartScreen class	20
Figure 5.5: CartScreenController 1	21
Figure 5.6: CartScreenController 2	21
Figure 5.7: Demo CartScreen	22
Figure 6.1: CartScreenController 1	22
Figure 6.2: CartScreenController 2	23
Figure 6.3: Demo media playable	23
Figure 6.4: Demo media unplayable	24
Figure 7.1: btnRemovePressed Method	24
Figure 7.2: button Remove	25
Figure 7.3: button Remove	25
Figure 8.1: Store before add book	26

Figure 8.2: Add book	26
Figure 8.3: Store after add book	27
Figure 8.4: Add CD	27
Figure 8.5: Store after add CD	28
Figure 8.6 Add DVD	28
Figure 8.7: Store after add DVD	29
Figure 8.8: Cart	29
Figure 8.9: Exception	30

1. Swing components

1.1 AWTAccumulator

```
package hust.soict.itcp.swing;

import java.awt.Label;
import java.awt.GridLayout;
import java.awt.TextField;
import java.awt.Frame;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

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("HoTuanAnh_20226100 Enter an integer: "));

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

        add(new Label("HoTuanAnh_20226100 The Accumulated Sum is: "));

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

        setTitle("HoTuanAnh_20226100 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;
            tfInput.setText("");
            tfOutput.setText(sum + "");
        }
    }
}
```

Figure 1.1: Source code of AWTAccumulator

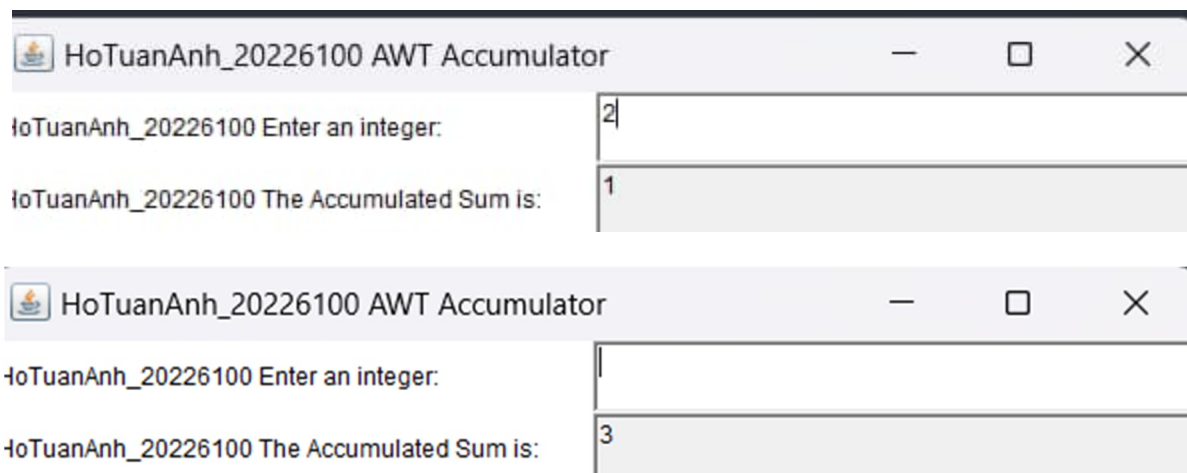


Figure 1.2: Demo of AWTAccumulator

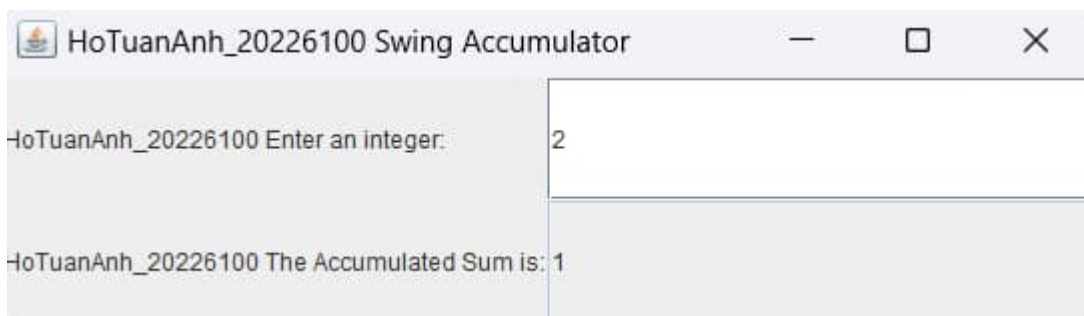
1.2 SwingAccumulator

```

src> hust> solc> rep> swing> | SwingAccumulator.java> SwingAccumulator
1 package hust.solc.itrep.swing;
2
3 import javax.swing.JFrame;
4 import javax.swing.JTextField;
5 import java.awt.*;
6 import java.awt.event.ActionEvent;
7 import java.awt.event.ActionListener;
8
9 public class SwingAccumulator extends JFrame {
10     private JTextField tfInput;
11     private JTextField tfOutput;
12     private int sum = 0;
13     public SwingAccumulator() {
14         Container cp = getContentPane();
15         cp.setLayout(new GridLayout(2,2));
16
17         cp.add(new Label("HoTuanAnh_20226100 Enter an integer: "));
18         tfInput = new JTextField(10);
19         cp.add(tfInput);
20         tfInput.addActionListener(new TFInputListener());
21
22         cp.add(new Label("HoTuanAnh_20226100 The Accumulated Sum is: "));
23         tfOutput = new JTextField(10);
24         tfOutput.setEditable(false);
25         cp.add(tfOutput);
26
27         setTitle("HoTuanAnh_20226100 Swing Accumulator");
28         setSize(350, 120);
29         setVisible(true);
30     }
31
32     // Main Program
33     public static void main(String[] args){
34         new SwingAccumulator();
35     }
36
37     private class TFInputListener implements ActionListener {
38         @Override
39         public void actionPerformed(ActionEvent evt){
40             int numberIn = Integer.parseInt(tfInput.getText());
41             sum += numberIn;
42             tfInput.setText("");
43             tfOutput.setText(sum + "");
44         }
45     }

```

Figure 1.3: Source code of SwingAccumulator



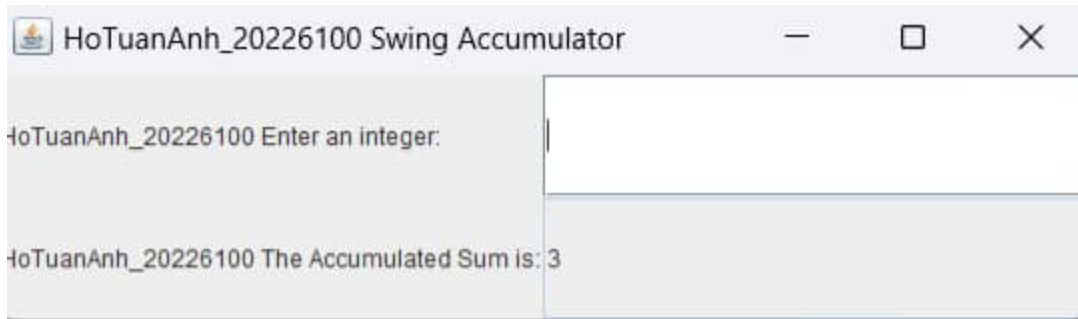


Figure 1.4: Demo of SwingAccumulator

2 Organizing Swing components with Layout Managers

2.1 Code

```
package hust.soict.itcp.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.RIGHT_TO_LEFT);
        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("HoTuanAnh_20226100 Number Grid");
        setSize(200,200);
        setVisible(true);
    }

    void addButtons(JPanel panelButtons) {
        ButtonListener btnListener = new ButtonListener();
        for (int i=1; i<=9; i++) {
            btnNumbers[i] = new JButton(""+ i);
            panelButtons.add(btnNumbers[i]);
            btnNumbers[i].addActionListener(btnListener);
        }

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

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

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

Figure 2.1: Source code of NumberGrid 1

```
package hust.soict.itcp.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.RIGHT_TO_LEFT);
        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("HoTuanAnh_20226100 Number Grid");
        setSize(200,200);
        setVisible(true);
    }

    void addButtons(JPanel panelButtons) {
        ButtonListener btnListener = new ButtonListener();
        for (int i=1;i<=9;i++) {
            btnNumbers[i] = new JButton("" + i);
            panelButtons.add(btnNumbers[i]);
            btnNumbers[i].addActionListener(btnListener);
        }

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

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

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

Figure 2.2: Source code of NumberGrid 2

2.2 Demo



Figure 2.3: Demo buttons 0-9

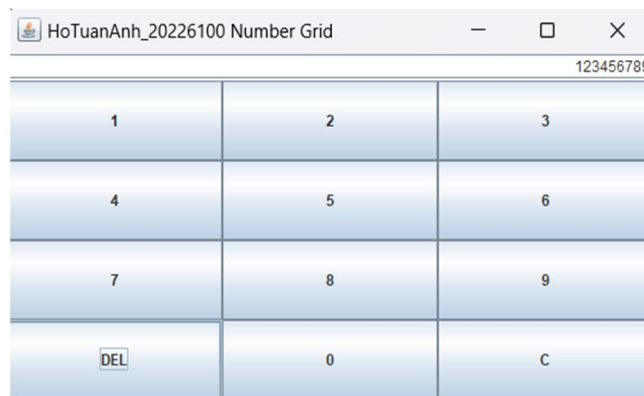


Figure 2.4: Demo DEL button

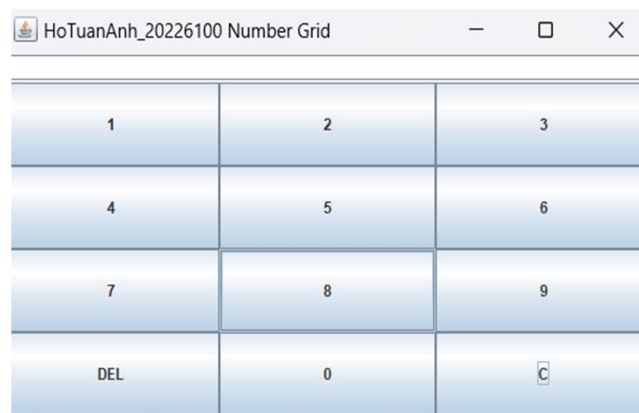


Figure 2.5: Demo C button

3 Create a graphical user interface for AIMS with Swing

3.1 Create class StoreScreen

```
src > hust > soict > itep > aims > screen > StoreScreen.java > StoreScreen > createHeader()
1 package hust.soict.itep.aims.screen;
2
3 import javax.swing.*;
4
5 import hust.soict.itep.aims.cart.Cart;
6 import hust.soict.itep.aims.media.Media;
7 import hust.soict.itep.aims.store.Store;
8
9 import java.awt.*;
10 import java.util.ArrayList;
11
12
13 public class StoreScreen extends JFrame{
14
15     private Store store;
16     private Container cp;
17     private Cart cart;
18
19     JPanel createNorth() {
20         JPanel north = new JPanel();
21         north.setLayout(new BorderLayout(north,BoxLayout.Y_AXIS));
22         north.add(createMenuBar());
23         north.add(createHeader());
24         return north;
25     }
26
27     JMenuBar createMenuBar() {
28         JMenu menu = new JMenu("Options");
29         JMenu smUpdateStore = new JMenu("Update Store");
30         smUpdateStore.add(new JMenuItem("Add Book"));
31         smUpdateStore.add(new JMenuItem("Add CD"));
32         smUpdateStore.add(new JMenuItem("Add DVD"));
33         menu.add(smUpdateStore);
34         menu.add(new JMenuItem("View store"));
35         menu.add(new JMenuItem("View cart"));
36
37         JMenuBar menuBar = new JMenuBar();
38         menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
39         menuBar.add(menu);
40         return menuBar;
41     }
42 }
```

Figure 3.1: Class StoreScreen 1

```
src > hust > soict > itep > aims > screen > StoreScreen.java > StoreScreen > createCenter()
13 public class StoreScreen extends JFrame{
14
15     JPanel createHeader() {
16         JPanel header = new JPanel();
17         header.setLayout(new BorderLayout(header,BoxLayout.X_AXIS));
18
19         JLabel title = new JLabel("AIMS");
20         title.setFont(new Font(title.getFont().getName(),Font.PLAIN,size:50));
21         title.setForeground(Color.CYAN);
22
23         JButton cart = new JButton("View cart");
24         cart.setPreferredSize(new Dimension(width:10, height:10));
25         cart.setMaximumSize(new Dimension(width:100, height:50));
26
27         header.add(Box.createRigidArea(new Dimension(width:10,height:10)));
28         header.add(title);
29         header.add(Box.createHorizontalGlue());
30         header.add(cart);
31         header.add(Box.createRigidArea(new Dimension(width:10,height:10)));
32         return header;
33     }
34
35     JPanel createCenter() {
36         JPanel center = new JPanel();
37         center.setLayout(new GridLayout(rows:3,cols:3,hgap:7,vgap:7));
38
39         ArrayList<Media> mediaStore = store.getItemsInStore();
40         for (int i = 0; i < 9; i++) {
41             MediaStore cell = new MediaStore(mediaStore.get(i));
42             center.add(cell);
43         }
44         return center;
45     }
46
47     public StoreScreen(Store store) {
48         this.store = store;
49         cp = getContentPane();
50         cp.setLayout(new BorderLayout());
51         cp.add(createNorth(),BorderLayout.NORTH);
52         cp.add(createCenter(),BorderLayout.CENTER);
53         setVisible(true);
54         setTitle("Store");
55         setSize(width:1024,height:768);
56     }
57 }
```

Figure 3.2: Class StoreScreen 2

3.2 Create class MediaStore

```
src > hust > soict > itep > aims > screen > MediaStore.java > MediaStore
1
2 package hust.soict.itep.aims.screen;
3
4 import javax.swing.*;
5 import java.awt.*;
6
7 import hust.soict.itep.aims.media.CompactDisc;
8 import hust.soict.itep.aims.media.DigitalVideoDisc;
9 import hust.soict.itep.aims.media.Media;
10 import hust.soict.itep.aims.media.Playable;
11 import hust.soict.itep.aims.media.Track;
12
13 import java.awt.event.ActionEvent;
14 import java.awt.event.ActionListener;
15
16
17
18 public class MediaStore extends JPanel {
19
20     private Media media;
21 }
```

Figure 3.7: Class MediaStore 1

```
src > hust > soict > itep > aims > screen > MediaStore.java > MediaStore > MediaStore(Media)
18 public class MediaStore extends JPanel {
23     public MediaStore(Media media) {
24
25         JLabel title = new JLabel(media.getTitle());
26         title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 20));
27         title.setAlignmentX(CENTER_ALIGNMENT);
28
29         JLabel cost = new JLabel("*** + media.getCost() + \"$");
30         cost.setAlignmentX(CENTER_ALIGNMENT);
31
32         JPanel container = new JPanel();
33         container.setLayout(new FlowLayout(FlowLayout.CENTER));
34
35         container.add(new JButton(text: "Add to cart"));
36
37         if (media instanceof Playable) {
38             JButton playButton = new JButton(text: "Play");
39             playButton.addActionListener(new ActionListener() {
40                 @Override
41                 public void actionPerformed(ActionEvent e) {
42                     JDialog playDialog = createPlayDialog(media);
43                     playDialog.setVisible(true);
44                     playDialog.setSize(width: 300, height: 200);
45                     playDialog.pack();
46                 }
47             });
48             container.add(playButton);
49         }
50
51         this.add(Box.createVerticalGlue());
52         this.add(title);
53         this.add(cost);
54         this.add(Box.createVerticalGlue());
55         this.add(container);
56         this.setBorder(BorderFactory.createLineBorder(Color.BLACK));
57     }
58
59     private JDialog createPlayDialog(Media media) {
60         JDialog playDialog = new JDialog();
61         Container container = playDialog.getContentPane();
62         playDialog.setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);
63         container.setLayout(new BorderLayout(container, BorderLayout.Y_AXIS));
64         container.add(Box.createRigidArea(new Dimension(width: 10, height: 10)));
65         if (media instanceof DigitalVideoDisc dvd) {
66             container.add(new JLabel("Playing DVD: " + dvd.getTitle()));
67             container.add(new JLabel("DVD length: " + dvd.getLength() + " min"));
68         } else if (media instanceof CompactDisc cd) {
69             container.add(new JLabel("Title: " + cd.getTitle()));
70             container.add(new JLabel("Artist: " + cd.getArtist()));
71             for (Track track : cd.getTracks()) {
72                 container.add(new JLabel("Play: " + track.getTitle() + ". Length: " + track.getLength() + " min"));
73             }
74         }
75     }
76 }
```

Figure 3.8: Class MediaStore 2

3.3 Demo

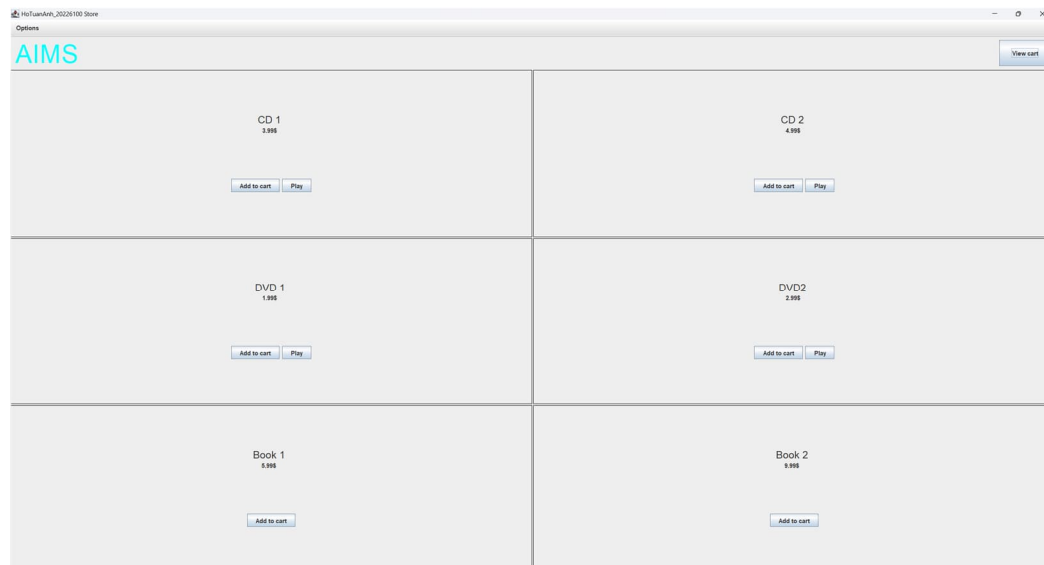


Figure 3.10: StoreScreen

Figure 3.11 Demo Add to cart button

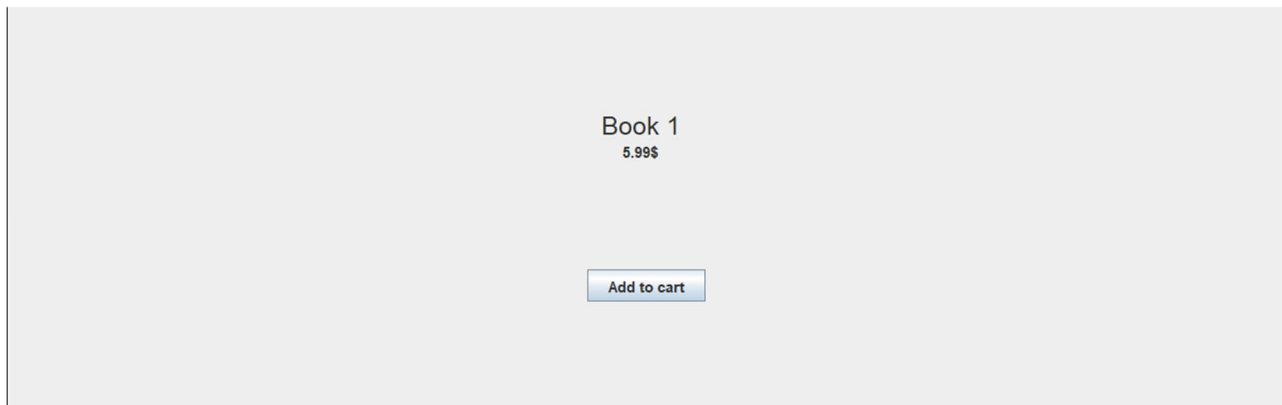
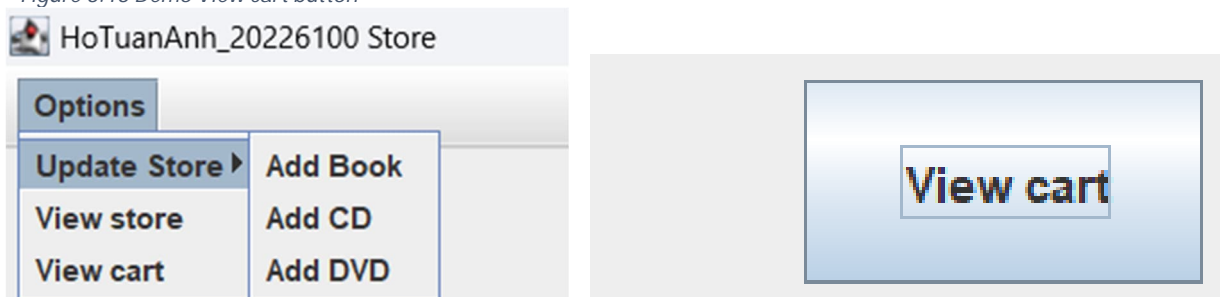


Figure 3.12 Demo Play button



Figure 3.13 Demo View cart button



4 JavaFX API

4.1 Create class Painter

```
src > hust > soict > Rep > javafx > Painter.java > Painter > start(Stage)
1 package hust.soict.itep.javafx;
2
3 import java.util.Objects;
4
5 import javafx.application.Application;
6 import javafx.fxml.FXMLLoader;
7 import javafx.scene.Parent;
8 import javafx.scene.Scene;
9 import javafx.stage.Stage;
10
11
12 public class Painter extends Application {
13
14     @Override
15     public void start(Stage stage) throws Exception {
16         try {
17             Parent root = FXMLLoader.load(Objects.requireNonNull(getClass().getResource("Painter.fxml")));
18             Scene scene = new Scene(root);
19             stage.setTitle("Painter");
20             stage.setScene(scene);
21             stage.show();
22         } catch (Exception e) {
23             System.out.println(e.toString());
24         }
25     }
26
27     public static void main(String[] args) {
28         launch(args);
29     }
30 }
```

Figure 4.1: Class Painter

4.2 Create Painter.fxml

```
src > hust > soict > Rep > javafx > Painter.fxml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <?import javafx.geometry.Insets?>
3 <?import javafx.scene.control.*?>
4 <?import javafx.scene.layout.*?>
5
6 <BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/23" xmlns:fx="
7
8     <center>
9         <AnchorPane id="drawingAreaPane" fx:id="drawingAreaPane" onMouseClicked="#drawAreaMouseClicked" onMouseDragged="#drawAreaMouseDragged" prefHeight="200.0" prefWidth="200.0" B
10     </center>
11     <left>
12         <VBox prefHeight="200.0" prefWidth="100.0" style="-fx-background-color: grey;" BorderPane.alignment="CENTER">
13             <children>
14                 <TitledPane animated="false" text="tool">
15                     <content>
16                         <AnchorPane minHeight="0.0" minWidth="0.0" prefHeight="80.0" prefWidth="82.0">
17                             <children>
18                                 <RadioButton fx:id="penRadioButton" layoutX="3.0" layoutY="10.0" mnemonicParsing="false" text="Pen" selected="true">
19                                     <toggleGroup fx:id="toggle" />
20                                 </RadioButton>
21                                 <RadioButton fx:id="eraserButton" layoutX="2.0" layoutY="40.0" mnemonicParsing="false" text="Eraser" toggleGroup="$toggle" />
22                             </children>
23                         </AnchorPane>
24                     </content>
25                 </TitledPane>
26                 <Button fx:id="clearButton" maxWidth="50000" mnemonicParsing="false" onAction="#clearButtonPressed" text="Clear" />
27             </children>
28             <padding>
29                 <Insets bottom="8.0" left="8.0" right="8.0" top="8.0" />
30             </padding>
31         </VBox>
32     </left>
33 </BorderPane>
```

Figure 4.2: Painter.fxml 1

4.3 Create class PainterController

```
src > hust > soict > itep > javafx > - PainterController.java > PainterController
1 package hust.soict.itep.javafx;
2
3 import java.util.Iterator;
4 import javafx.scene.Node;
5
6 import javafx.event.ActionEvent;
7 import javafx.fxml.FXML;
8 import javafx.scene.control.RadioButton;
9 import javafx.scene.input.MouseEvent;
10 import javafx.scene.layout.AnchorPane;
11 import javafx.scene.paint.Color;
12 import javafx.scene.shape.Circle;
13
14 public class PainterController {
15
16     @FXML
17     private RadioButton penRadioButton;
18
19     @FXML
20     private RadioButton eraserButton;
21
22     @FXML
23     private AnchorPane drawingAreaPane;
24
25     @FXML
26     void clearButtonPressed(ActionEvent event) {
27         drawingAreaPane.getChildren().clear();
28     }
29
30     @FXML
31     void drawAreaMouseDragged(MouseEvent event) {
32         if (penRadioButton.isSelected()) {
33             Circle circle = new Circle(event.getX(), event.getY(), 4, Color.BLACK);
34             drawingAreaPane.getChildren().add(circle);
35         } else if (eraserButton.isSelected()) {
36             Iterator<Node> iter = drawingAreaPane.getChildren().iterator();
37             while (iter.hasNext()) {
38                 Node c = iter.next();
39                 if (c instanceof Circle circle) {
40                     if (circle.getCenterX() ≤ event.getX() + 4 && circle.getCenterX() ≥ event.getX() - 4) {
41                         if (circle.getCenterY() ≤ event.getY() + 4 && circle.getCenterY() ≥ event.getY() - 4) iter.remove();
42                     }
43                 }
44             }
45         }
46     }
47
48 }
49
50
51 }
```

Figure 4.4: PainterController

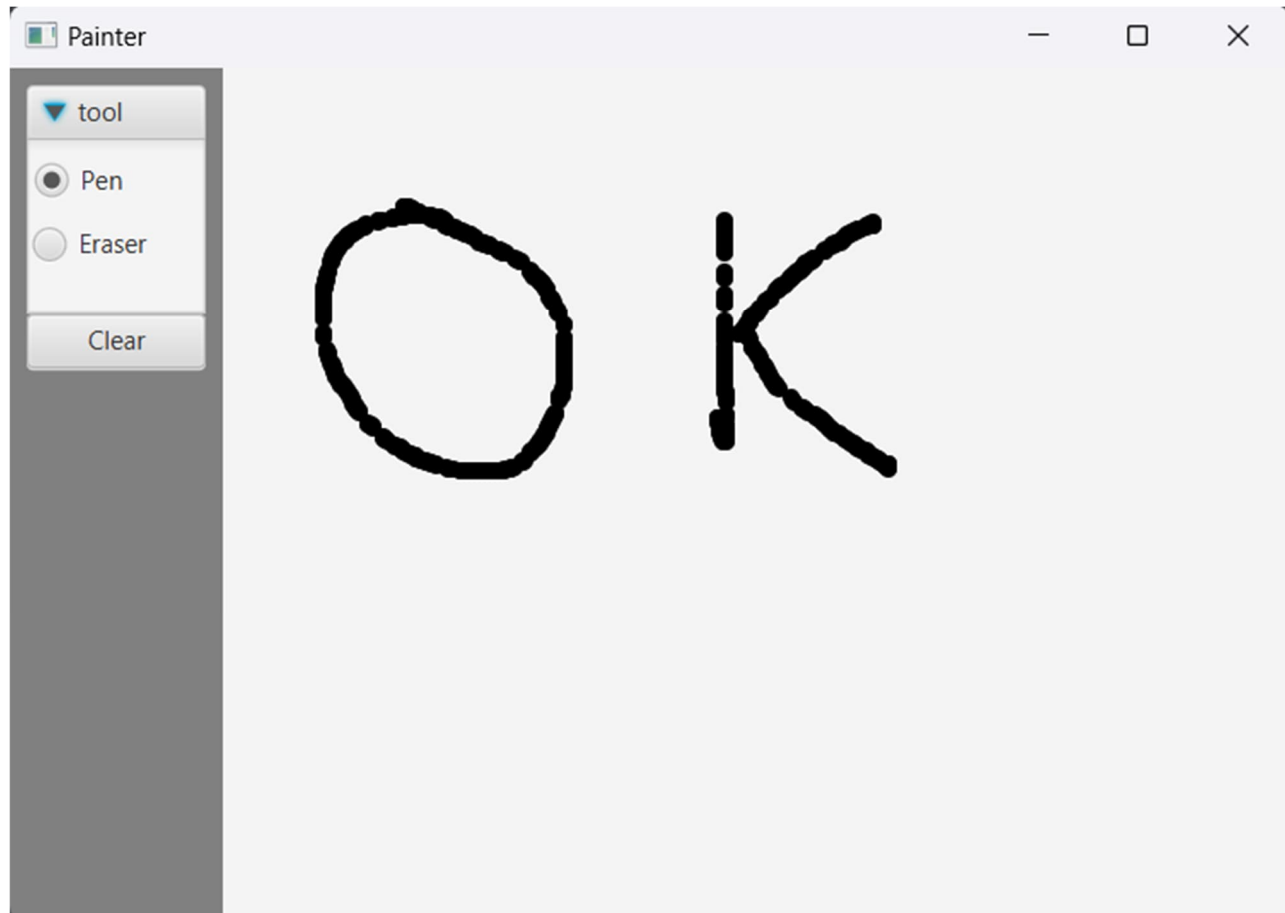


Figure 4.5: Use Pen

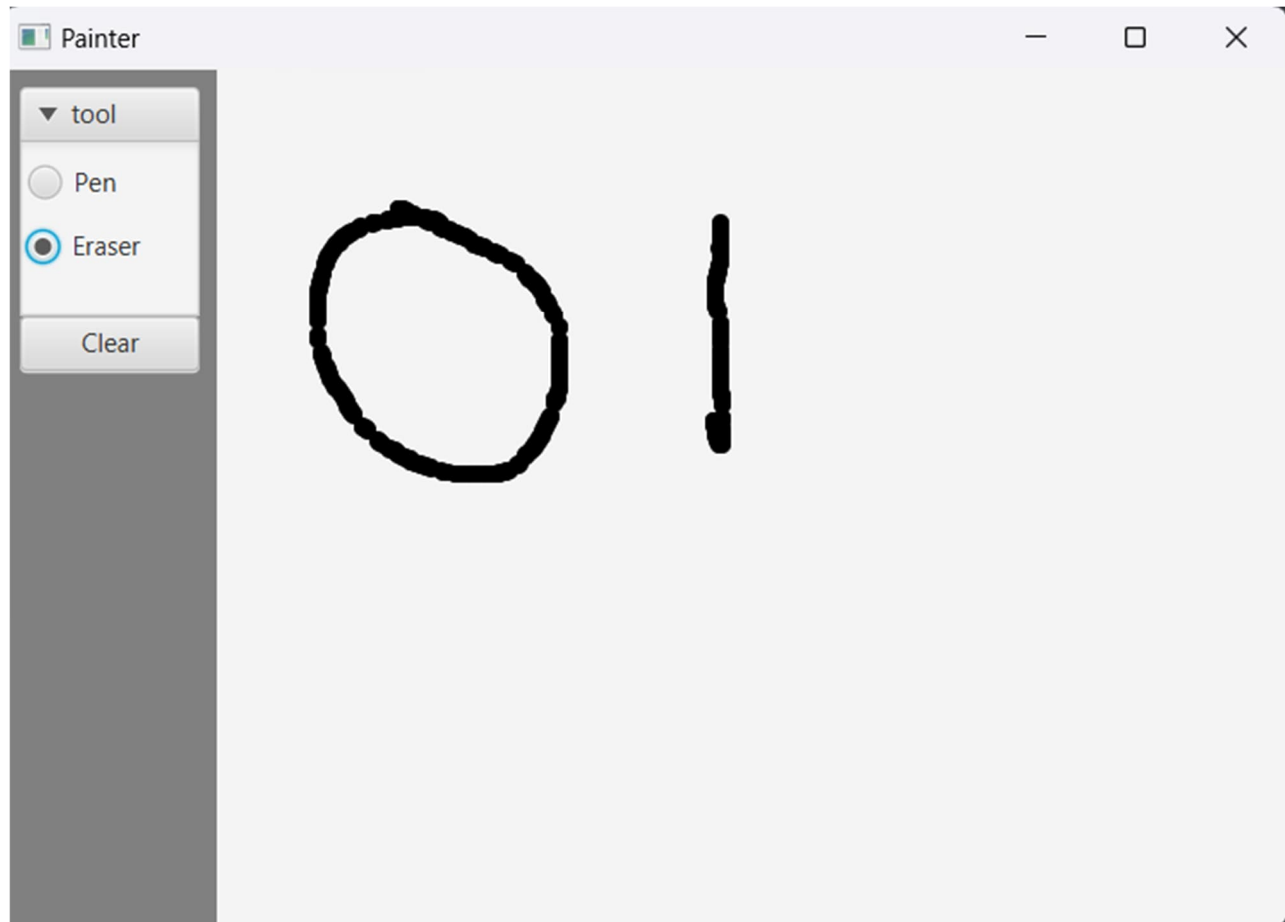


Figure 4.6: Use Eraser

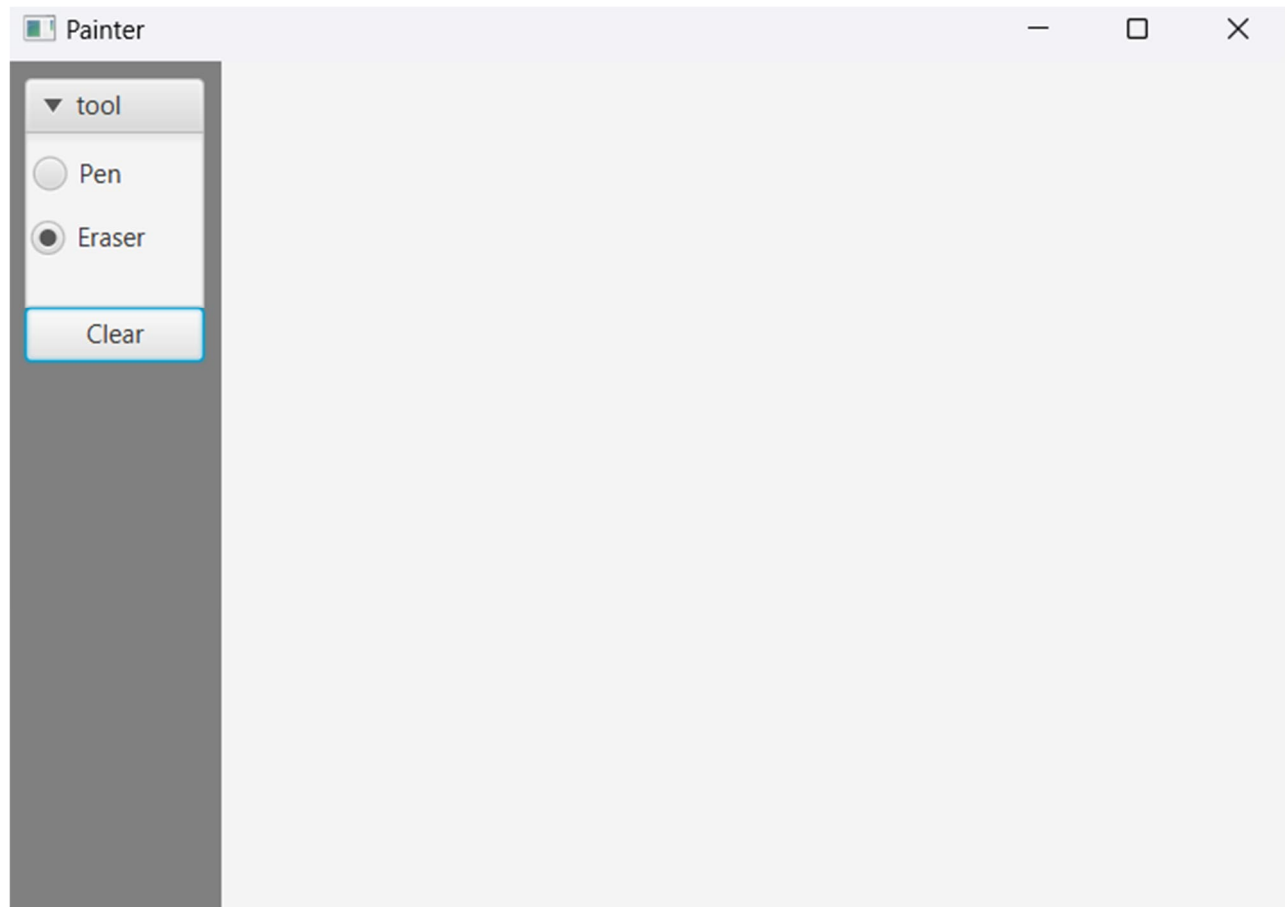


Figure 4.7: Clear button

5 View Cart Screen

5.1 Create cart.fxml

```
src > hust > soict > itep > aims > screen > fxml > cart.fxml
You, 1 second ago | 1 author (You)
1 <?xml version="1.0" encoding="UTF-8"?>
2 <?import javafx.geometry.Insets?>
3 <?import javafx.scene.control.Button?>
4 <?import javafx.scene.control.ButtonBar?>
5 <?import javafx.scene.control.Label?>
6 <?import javafx.scene.control.Menu?>
7 <?import javafx.scene.control.MenuBar?>
8 <?import javafx.scene.control.MenuItem?>
9 <?import javafx.scene.control.RadioButton?>
10 <?import javafx.scene.control.TableColumn?>
11 <?import javafx.scene.control.TableView?>
12 <?import javafx.scene.control.TextField?>
13 <?import javafx.scene.control.ToggleGroup?>
14 <?import javafx.scene.layout.BorderPane?>
15 <?import javafx.scene.layout.HBox?>
16 <?import javafx.scene.layout.VBox?>
17 <?import javafx.scene.text.Font?>
18
19 <BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="768.0" prefWidth="1024.0" xmlns="http://javafx.com/javafx/23" xmlns:fx="http://javafx.com/fxml" >
20 <top>
21 <VBox prefWidth="100.0" BorderPane.alignment="CENTER">
22 <children>
23 <MenuBar>
24 <menus>
25 <Menu mnemonicParsing="false" text="Options">
26 <items>
27 <MenuItem mnemonicParsing="false" text="Update store">
28 <items>
29 <MenuItem mnemonicParsing="false" text="Add book" />
30 <MenuItem mnemonicParsing="false" text="Add DVD" />
31 <MenuItem mnemonicParsing="false" text="Add CD" />
32 </items>
33 </MenuItem>
34 <MenuItem mnemonicParsing="false" text="DVD" />
35 <MenuItem mnemonicParsing="false" text="CD" />
36 <MenuItem mnemonicParsing="false" text="View cart">
37 </MenuItem>
38 </items>
39 </Menu>
40 </menus>
41 </MenuBar>
42 <Label text="CART" textFill="#000fff">
43 <font>
44 <Font size="50.0" />
45 </font>
46 </Label>
47 </children>
48 </VBox>
49 </top>
```

Figure 5.1: Cart.fxml 1

```
</VBox>
</top>
<center>
<VBox BorderPane.alignment="CENTER">
<padding>
<Insets left="10.0" />
</padding>
<children>
<HBox alignment="CENTER_LEFT" prefWidth="200.0" spacing="10.0">
<padding>
<Insets bottom="10.0" top="10.0" />
</padding>
<children>
<Label text="Filter" />
<TextField fx:id="tfFilter" />
<RadioButton fx:id="radioBtnFilterId" mnemonicParsing="false" selected="true" text="By ID">
<toggleGroup>
<ToggleGroup fx:id="filterCategory" />
</toggleGroup>
</RadioButton>
<RadioButton fx:id="radioBtnFilterTitle" mnemonicParsing="false" text="By Title" toggleGroup="$filterCategory" />
</children>
</HBox>
<TableView fx:id="tblMedia" prefHeight="200.0" prefWidth="200.0">
<columns>
<TableColumn fx:id="colMediaTitle" prefWidth="75.0" text="Title" />
<TableColumn fx:id="colMediaCategory" prefWidth="75.0" text="Category" />
<TableColumn fx:id="colMediaCost" prefWidth="75.0" text="Cost" />
</columns>
<columnResizePolicy>
<TableView fx:constant="CONSTRAINED_RESIZE_POLICY" />
</columnResizePolicy>
</TableView>
<ButtonBar prefHeight="40.0" prefWidth="200.0">
<buttons>
<Button fx:id="btnPlay" mnemonicParsing="false" text="Play" />
<Button fx:id="btnRemove" mnemonicParsing="false" text="Remove" />
</buttons>
</ButtonBar>
</children>
</VBox>
</center>
```

Figure 5.2: Cart.fxml 2

```

</center>
<right>
    <VBox alignment="TOP_CENTER" BorderPane.alignment="CENTER">
        <padding>
            <Insets left="10.0" right="10.0" top="50.0" />
        </padding>
        <children>
            <HBox alignment="CENTER">
                <children>
                    <Label text="Total:">
                        <font>
                            <font size="24.0" />
                        </font>
                    </Label>
                    <Label fx:id="totalPrice" text="0 $" textFill="#000fff">
                        <font>
                            <font size="24.0" />
                        </font>
                    </Label>
                </children>
            </HBox>
            <Button fx:id="btnPlaceOrder" mnemonicParsing="false" style="-fx-background-color: red;" text="Place Order" textFill="WHITE">
                <font>
                    <font size="24.0" />
                </font>
            </Button>
        </children>
    </VBox>
</right>
</BorderPane>

```

Figure 5.3: Cart.fxml 3

5.2 Create class CartScreen

```

src > hust > soict > itep > aims > screen > CartScreen.java
1 package hust.soict.itep.aims.screen;
2
3 import java.io.IOException;
4
5 import javax.swing.JFrame;
6
7 import hust.soict.itep.aims.cart.Cart;
8 import javafx.application.Platform;
9 import javafx.embed.swing.JFXPanel;
10 import javafx.fxml.FXMLLoader;
11 import javafx.scene.Parent;
12 import javafx.scene.Scene;
13
14 You, 9 hours ago · 1 author · You
15 public class CartScreen extends JFrame
16 private Cart cart;
17
18 public CartScreen(Cart cart) {
19     super();
20     this.cart = cart;
21
22     JFXPanel fxPanel = new JFXPanel();
23     this.add(fxPanel);
24     this.setTitle(title: "HoTuanAnh_20226100 Cart");
25     this.setVisible(true);
26     setSize(width: 1100, height: 768);
27
28 You, 9 hours ago · 1 author · You
29 Platform.runLater(new Runnable() {
30     @Override
31     public void run() {
32         try {
33             FXMLLoader loader = new FXMLLoader(getClass().getResource("../screen/fxml/cart.fxml"));
34             CartScreenController controller = new CartScreenController(cart);
35             loader.setController(controller);
36             Parent root = loader.load();
37             fxPanel.setScene(new Scene(root));
38         } catch (IOException e) {
39             System.err.println(e.getMessage());
40         }
41     }
42 });
43
44
45

```

Figure 5.4: CartScreen class

5.3 Create class CartScreenController

```
src > hust > soict > Reg > aims > screen > - CartScreenController.java > CartScreenController > showFilteredMedia(String)
35 public class CartScreenController {
36
37
38     private Cart cart;
39
40     @FXML
41     private Button btnPlaceOrder;
42     @FXML
43     private TextField tfFilter;
44     @FXML
45     private ToggleGroup filterCategory;
46     @FXML
47     private RadioButton radioBtnFilterId;
48     @FXML
49     private RadioButton radioBtnFilterTitle;
50     @FXML
51     private Button btnPlay;
52     @FXML
53     private Button btnRemove;
54     @FXML
55     private TableView<Media> tblMedia;
56     @FXML
57     private TableColumn<Media, String> colMediaTitle;
58     @FXML
59     private TableColumn<Media, String> colMediacategory;
60     @FXML
61     private TableColumn<Media, Float> colMediaCost;
62     @FXML
63     private Label totalPrice;
64
65
66     public CartScreenController(Cart cart) {
67         super();
68         this.cart = cart;
69     }
70
71     @FXML
72     private void initialize() {
73         colMediaTitle.setCellValueFactory(new PropertyValueFactory<Media, String>("title"));
74         colMediacategory.setCellValueFactory(new PropertyValueFactory<Media, String>("category"));
75         colMediaCost.setCellValueFactory(new PropertyValueFactory<Media, Float>("cost"));
76         tblMedia.setItems(FXCollections.observableList(this.cart.getItemsOrdered()));
77         tblMedia.setPlaceholder(new Label("No item in cart"));
78         btnPlay.setVisible(false);
79         btnRemove.setVisible(false);
80
81         btnRemove.setOnAction((EventHandler<ActionEvent>) new EventHandler<ActionEvent>() {
82             @Override
83             public void handle(ActionEvent event) {
84                 Media media = tblMedia.getSelectionModel().getSelectedItem();
85                 try {
86                     cart.removeMedia(media);
87                 } catch (Exception e) {
88                     e.printStackTrace();
89                 }
90             }
91         });
92     }
93 }
```

Figure 5.5: CartScreenController 1

```
src > hust > soict > itep > aims > screen > CartScreenController.java > CartScreenController > btnRemovePressed(ActionEvent)
25 public class CartScreenController {
26     private void initialize() {
27         btnPlaceOrder.setOnAction(new EventHandler<ActionEvent>() {
28             @Override
29             public void handle(ActionEvent event) {
30                 createPopup();
31                 cart.getItemsOrdered().clear();
32                 tblMedia.setItems(FXCollections.observableList(cart.getItemsOrdered()));
33                 totalPrice.setText(Float.toString(cart.totalCost()) + "$");
34             }
35         });
36     }
37
38     @FXML
39     void showFilteredMedia(String s) {
40         java.util.ArrayList<Media>()
41         if (filterCategory.getSelectedToggle() == Constructs an empty list with an initial capacity of ten.
42             ArrayList<Media> filterByTitle = new ArrayList<Media>();
43             for (Media item : cart.getItemsOrdered()) {
44                 if (item.getTitle().contains(s)) {
45                     filterByTitle.add(item);
46                 }
47             }
48             tblMedia.setItems(FXCollections.observableList(filterByTitle));
49         } else if (filterCategory.getSelectedToggle() == radioBtnFilterId) {
50             ArrayList<Media> filterById = new ArrayList<Media>();
51             for (Media item : cart.getItemsOrdered()) {
52                 if (item.getId() == Integer.parseInt(s)) {
53                     filterById.add(item);
54                 }
55             }
56             tblMedia.setItems(FXCollections.observableList(filterById));
57         }
58     }
59
60     @FXML
61     void createPopup() {
62         Stage popupwindow = new Stage();
63         popupwindow.initModality(Modality.APPLICATION_MODAL);
64         popupwindow.setTitle("Place order");
65
66         Label label1 = new Label("You have place your order !");
67         label1.setFont(Font.font("San Serif", FontWeight.BOLD, 14));
68         Label label2 = new Label("Your bill total is " + Float.toString(cart.totalCost()) + "$");
69         Button button1 = new Button("OK !");
70         label2.setTextFill(Color.RED);
71         button1.setOnAction(e -> popupwindow.close());
72         VBox layout = new VBox(10);
73         layout.getChildren().addAll(label1, label2, button1);
74         layout.setAlignment(Pos.CENTER);
75         Scene scene1 = new Scene(layout, 300, 200);
76         popupwindow.setScene(scene1);
77         popupwindow.show();
78     }
79 }
```

Figure 5.6: CartScreenController 2

5.4 Demo

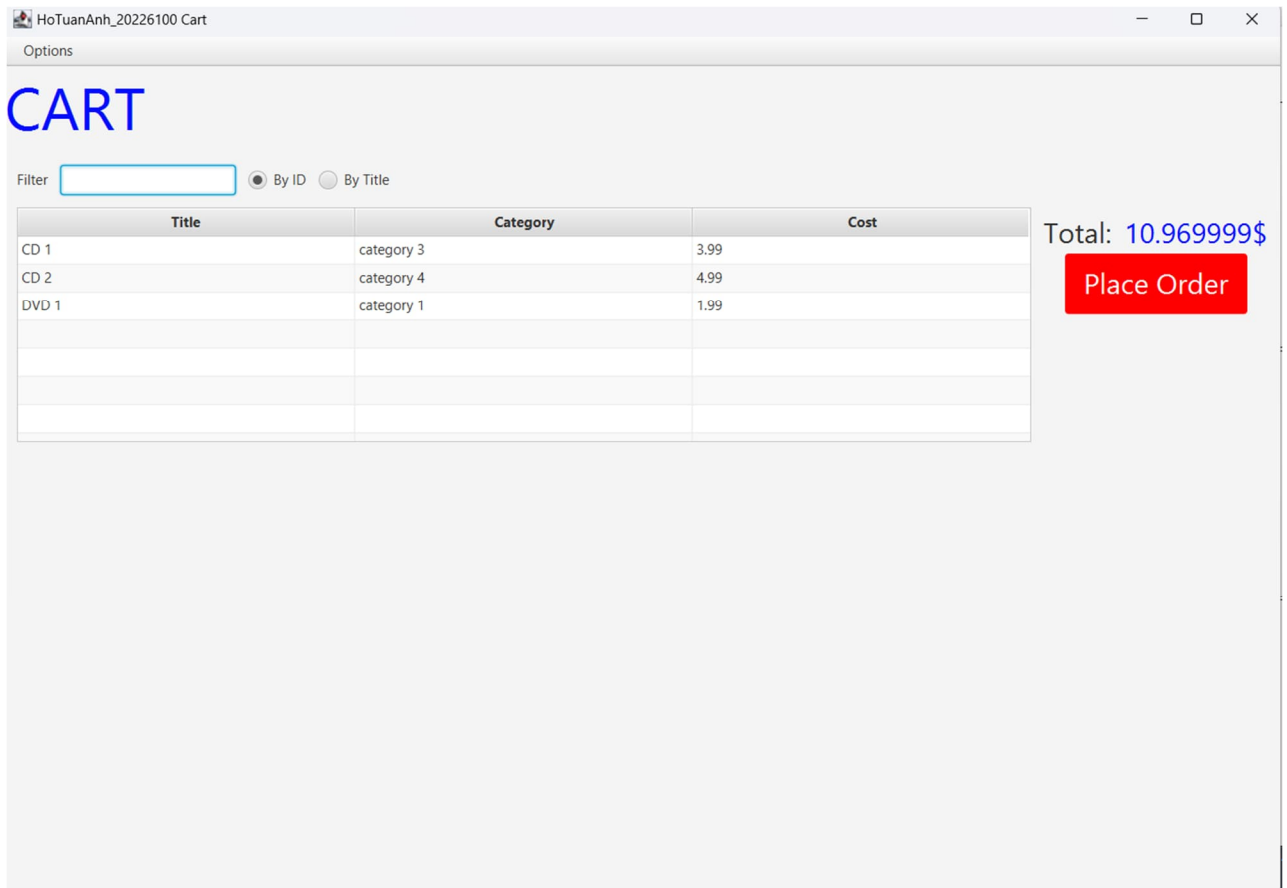


Figure 5.7: Demo CartScreen

6 Updating buttons based on selected item in TableView – ChangeListener

6.1 Edit class CartScreenController



Figure 6.1: CartScreenController 1

6.2 Demo

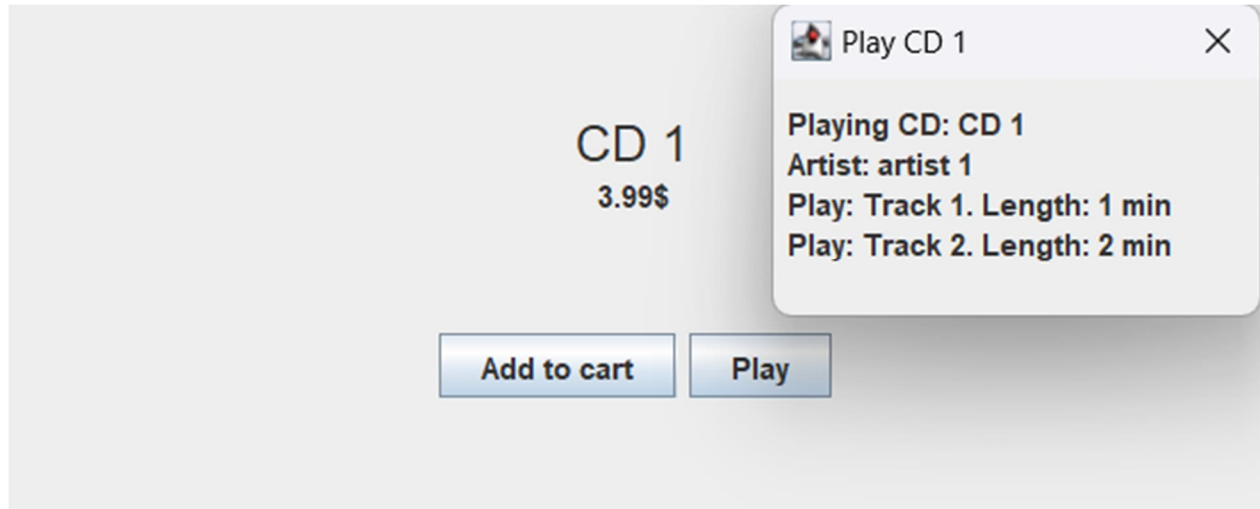


Figure 6.3: Demo media playable

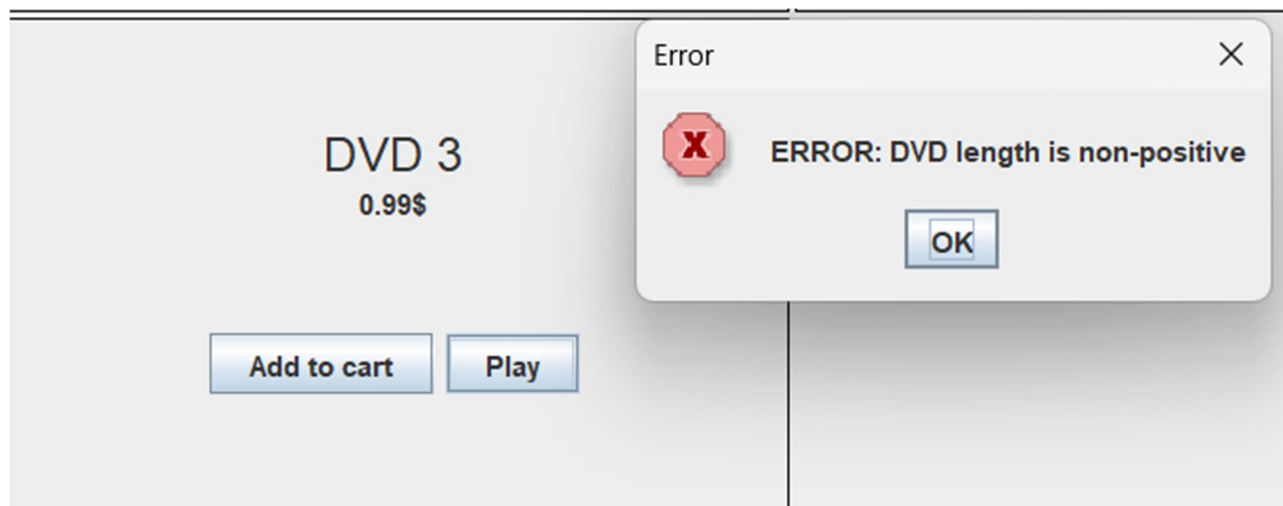


Figure 6.4: Demo media unplayable

7 Deleting a media

7.1 Code

```
@FXML
void btnRemovePressed(ActionEvent event) {
    Media media = tblMedia.getSelectionModel().getSelectedItem();
    try {
        cart.removeMedia(media);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

Figure 7.1: btnRemovePressed Method

2.1 Demo



Figure 7.2: button Rem

Complete the Aims GUI application

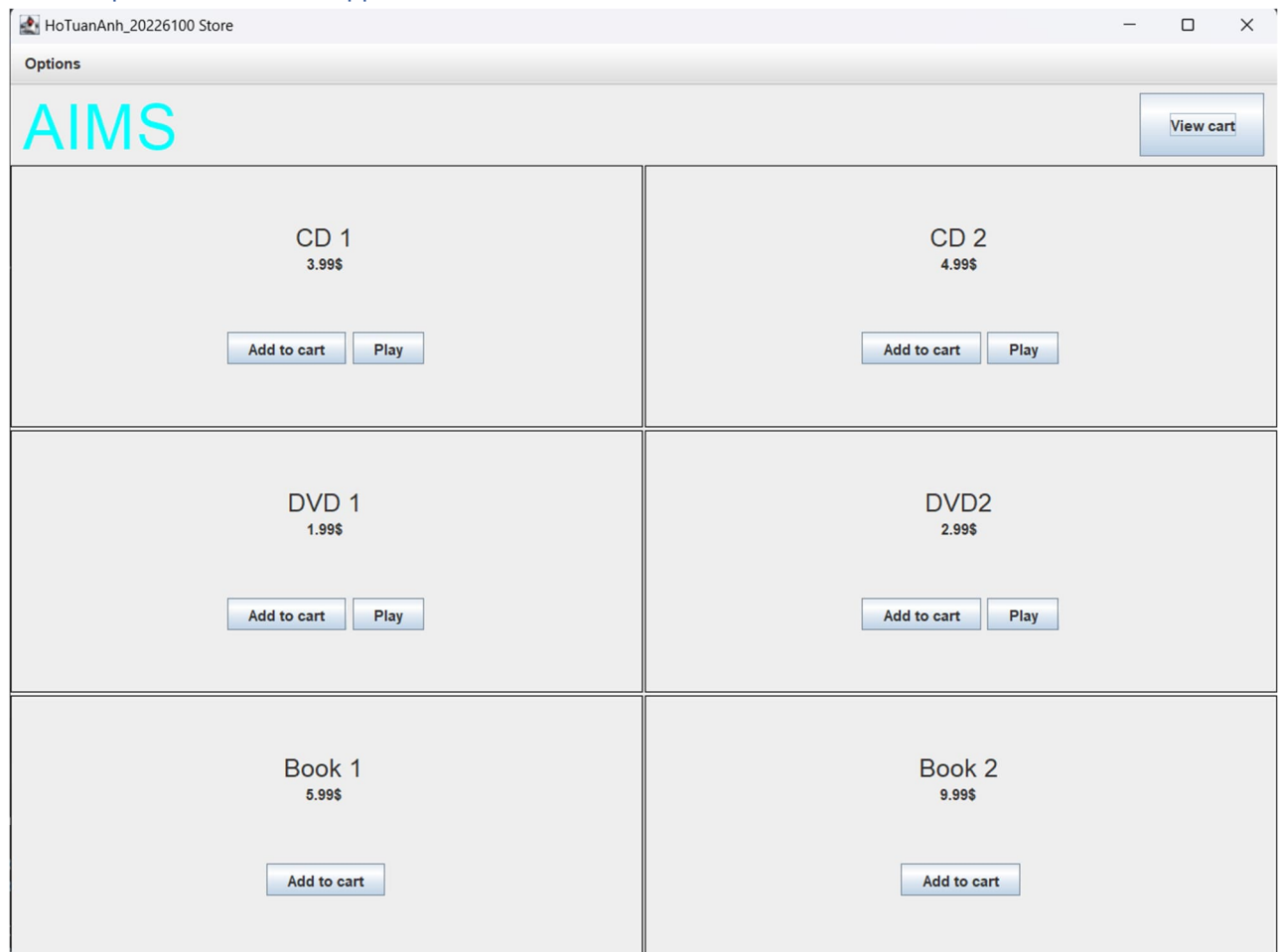
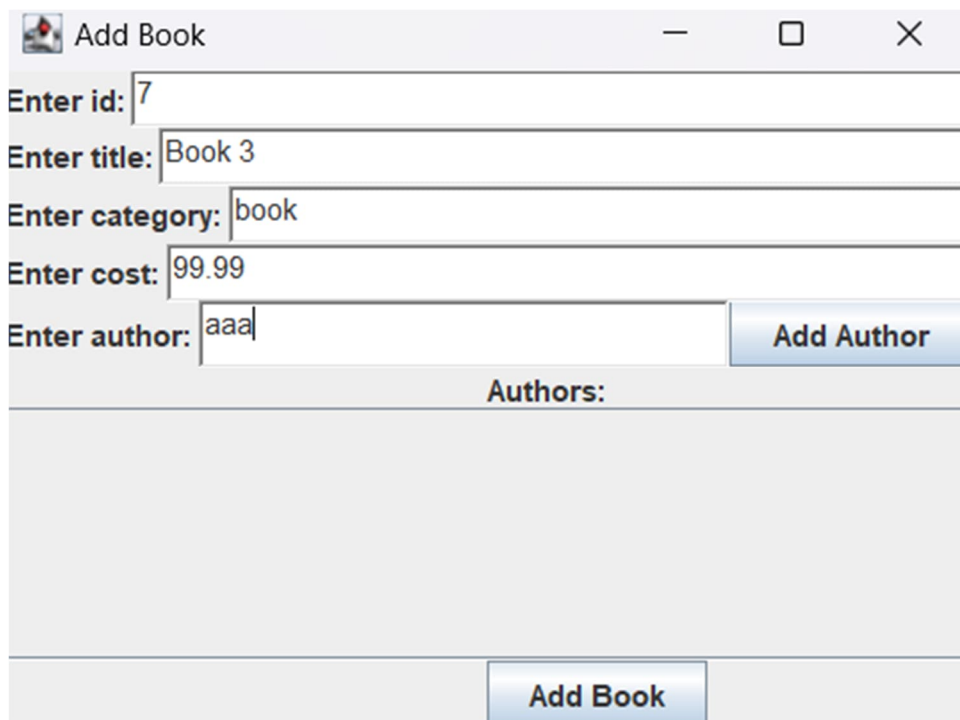


Figure 8.1: Store before add book



The screenshot shows a window titled "Add Book" with a standard Windows-style title bar (minimize, maximize, close buttons). The window contains the following elements:

- Enter id:** A text input field containing the value "7".
- Enter title:** A text input field containing the value "Book 3".
- Enter category:** A text input field containing the value "book".
- Enter cost:** A text input field containing the value "99.99".
- Enter author:** A text input field containing the value "aaa". To the right of this field is a button labeled "Add Author".
- Authors:** A label positioned above a large, empty rectangular area, likely intended for a list of authors.
- Add Book:** A button located at the bottom right of the window.

Figure 8.2: Add book

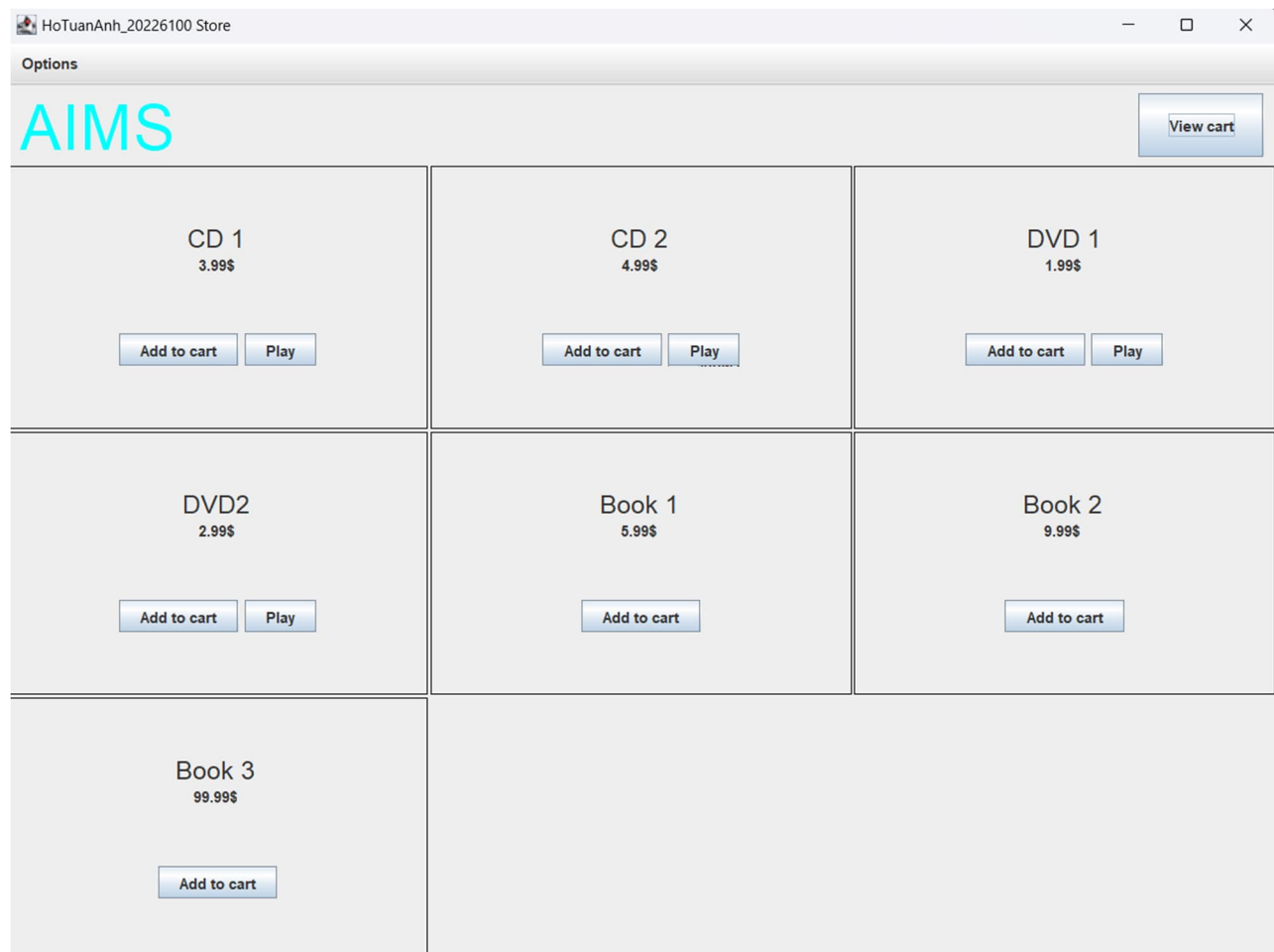


Figure 8.3: Store after add book

The screenshot shows a Java Swing window titled "Add Compact Disc". The window contains several input fields for adding a new CD:

- Enter id:** 8
- Enter title:** CD 3
- Enter category:** cd
- Enter cost:** 15.99
- Enter director:** bbb
- Enter artist:** xxx

Below these fields is a button labeled "Add Track". Underneath the button is a list of tracks, each with its own "Track title" and "Length" input fields, and a "Remove" button:

- Track title: t1 Length: 1 Remove
- Track title: t2 Length: 2 Remove
- Track title: t3 Length: 3 Remove

The window has a standard title bar with minimize, maximize, and close buttons. At the bottom of the window, there is a blue bar with the text "Add Compact Disc".

Figure 8.4: Add CD

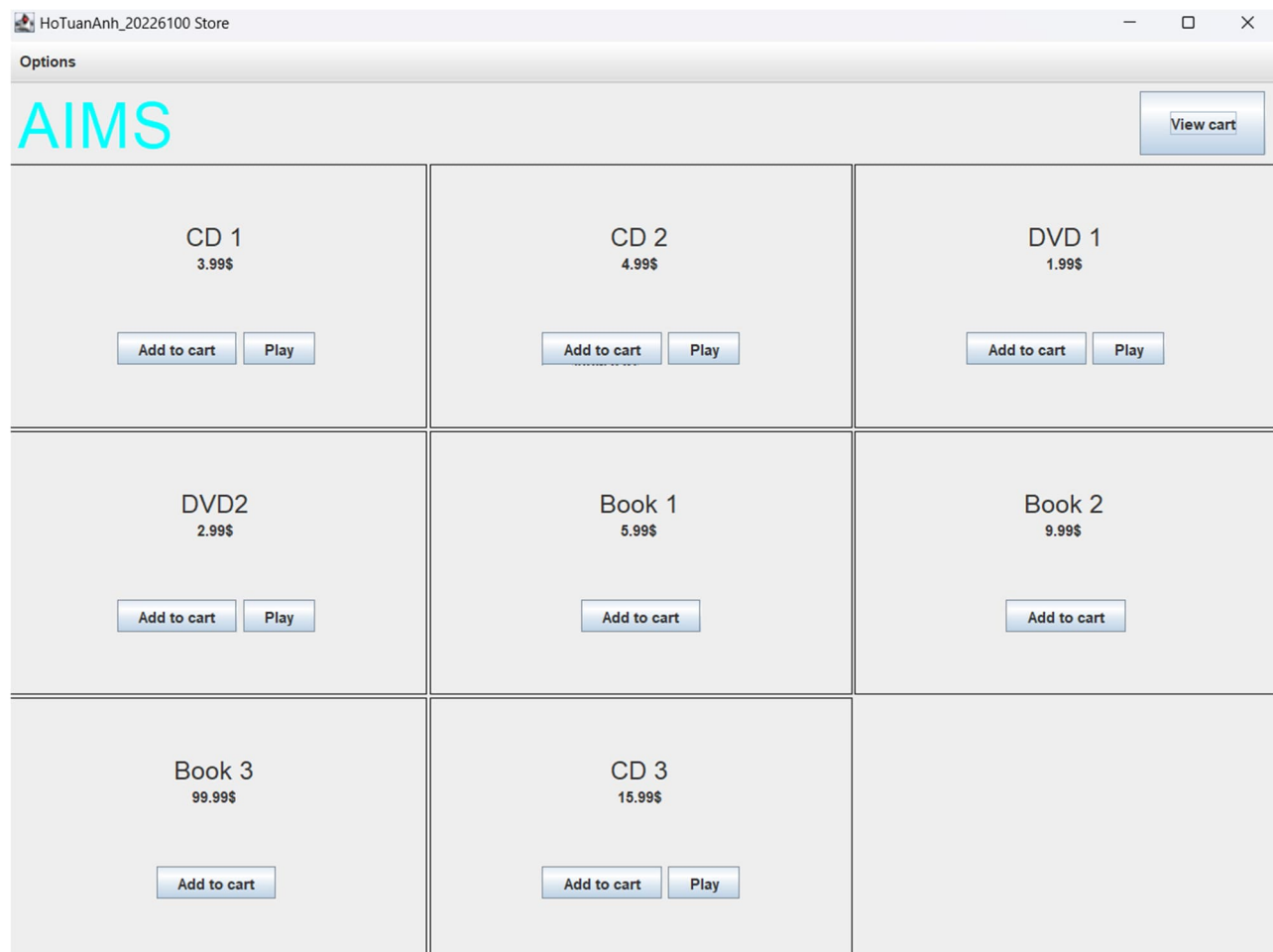
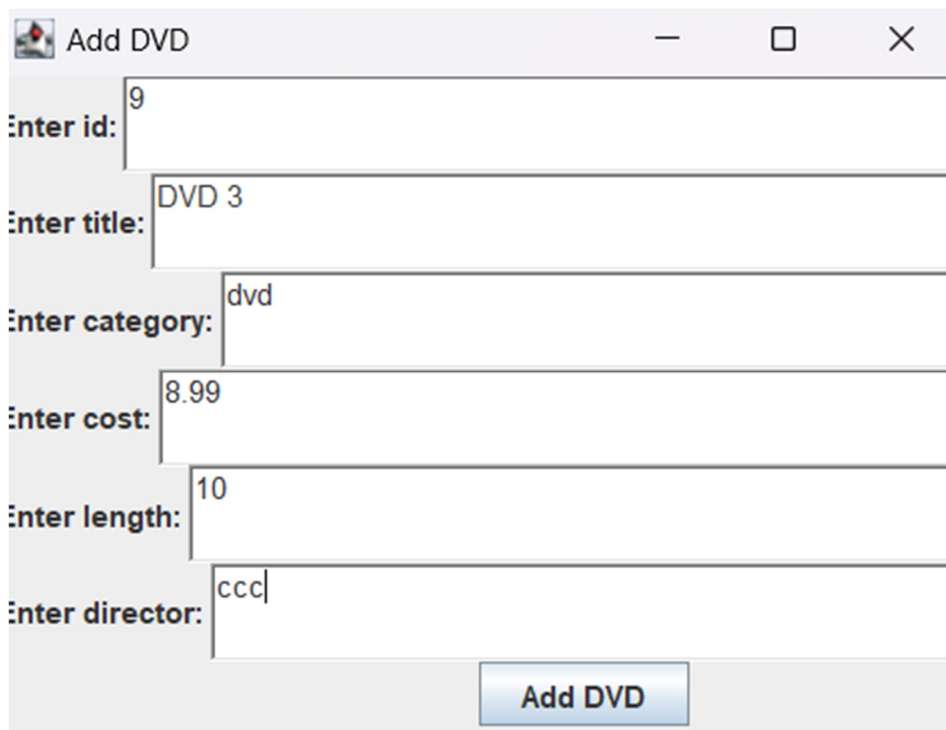


Figure 8.5: Store after add CD



The image shows a Java Swing window titled "Add DVD". The window contains six text input fields and one button. The fields are labeled "Enter id:", "Enter title:", "Enter category:", "Enter cost:", "Enter length:", and "Enter director:". The values entered in the fields are "9", "DVD 3", "dvd", "8.99", "10", and "ccc" respectively. The "Add DVD" button is located at the bottom right of the window.

Field Label	Value
Enter id:	9
Enter title:	DVD 3
Enter category:	dvd
Enter cost:	8.99
Enter length:	10
Enter director:	ccc

Figure 8.6 Add DVD

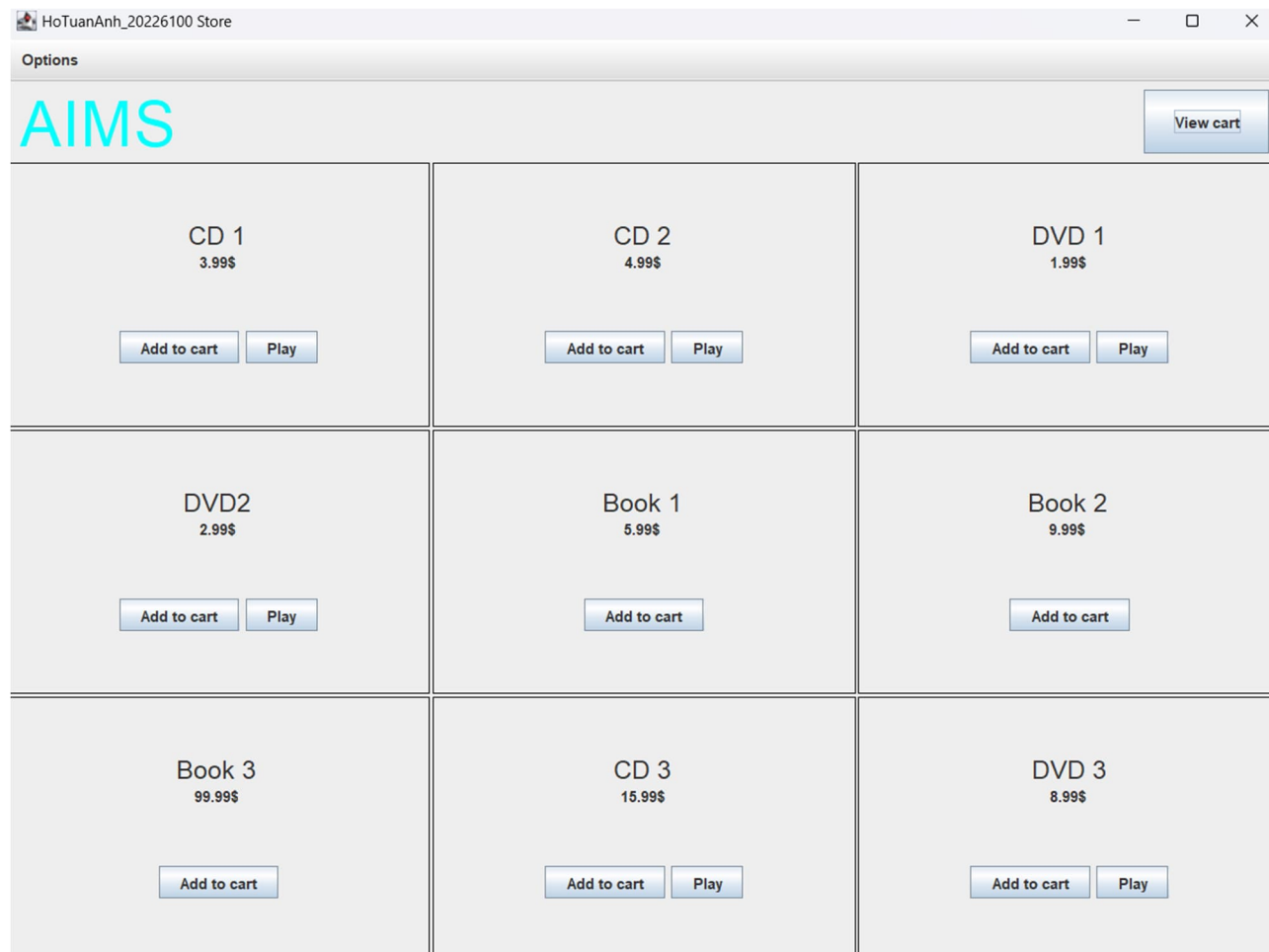


Figure 8. 7: Store after add DVD

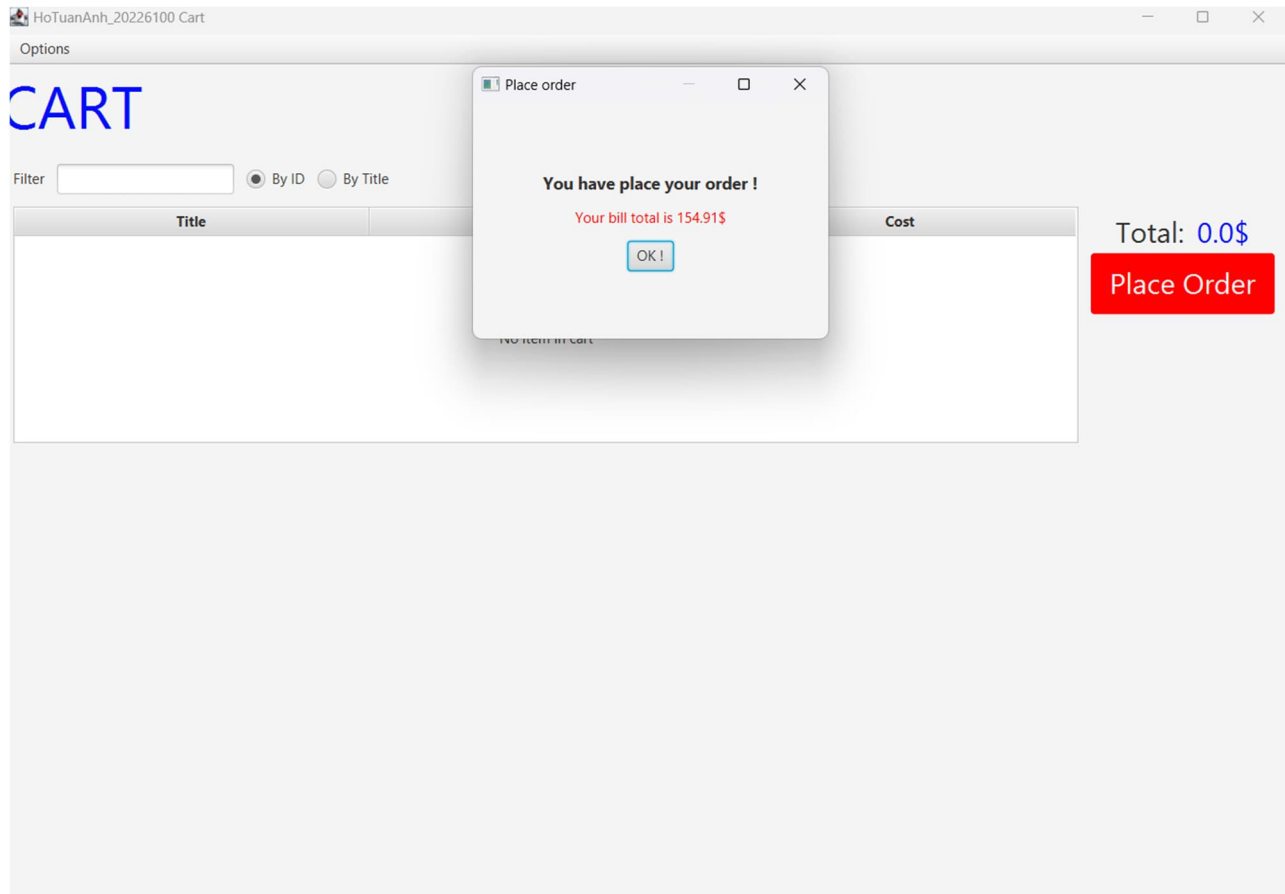
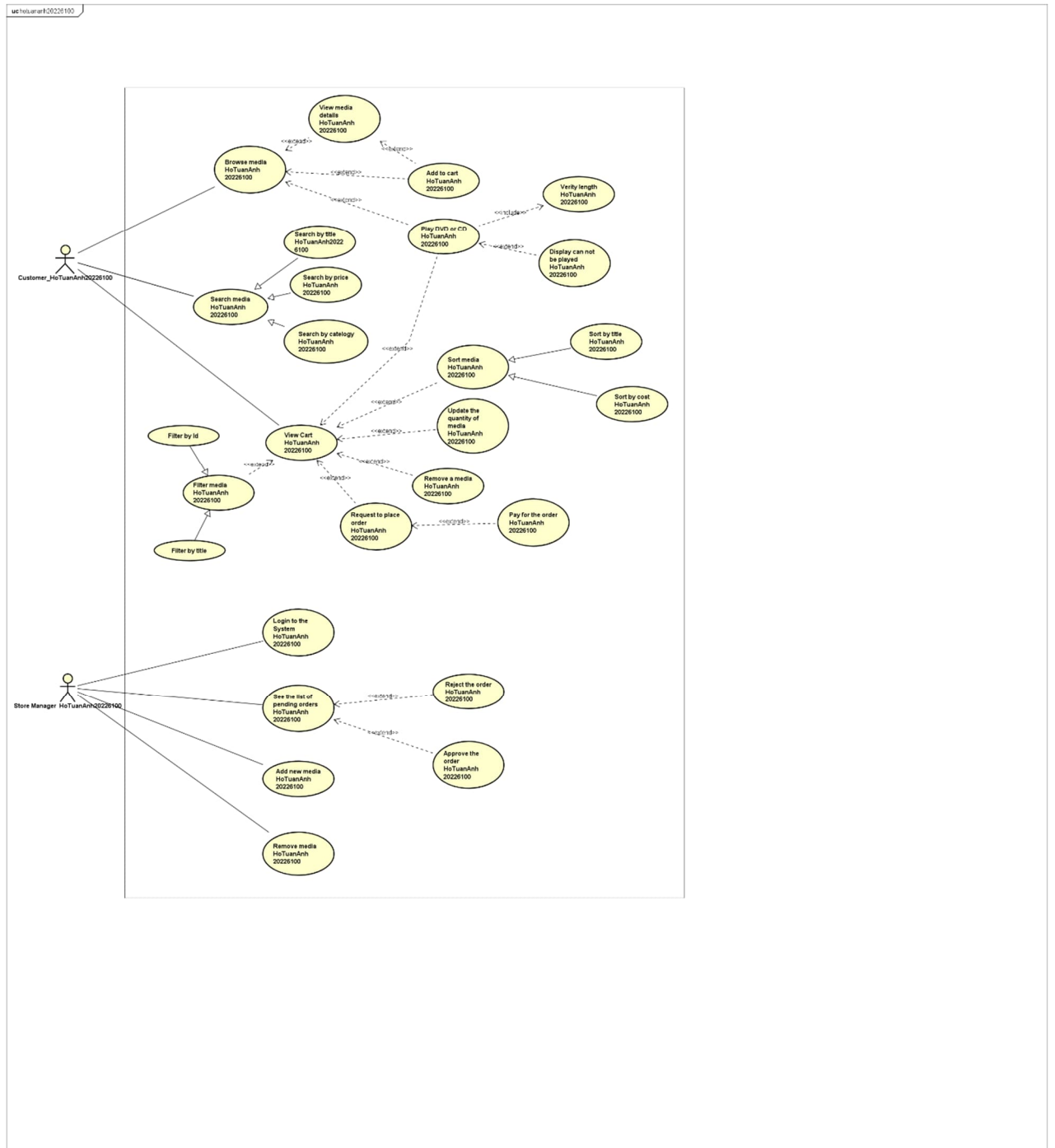


Figure 8.8: Cart

```
src > hust > soict > itep > aims > exception > PlayerException.java > PlayerException
You, 9 hours ago | 1 author (You)
1 package hust.soict.itep.aims.exception;
2
3 You, 9 hours ago | 1 author (You)
4 public class PlayerException extends Exception {
5
6     public PlayerException(String message) {
7         super(message);
8     }
9     You, 9 hours ago • update lab5
10 }
11
```

Figure 8.9: Exception

3. Usecase Diagram



4. Class Diagram

