

ĐẠI HỌC BÁCH KHOA HÀ NỘI



**BÁO CÁO THỰC HÀNH
LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG
LAB 05 - GUI PROGRAMMING**

Mã HP: IT3103 - Mã lớp: 744520

Sinh viên: Lê Hải Yến - MSSV: 20225780

I. New written code

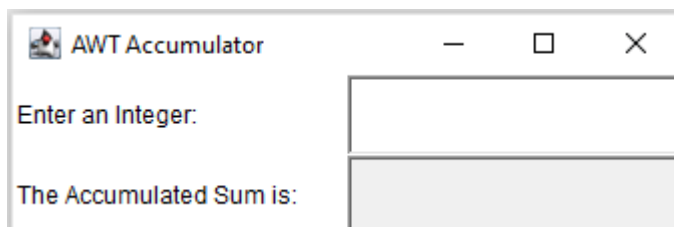
1. Swing Component

- **Branch: topic/aims-project/gui-project-swingcomponent**
- Tạo một dự án Java mới có tên là **GUIProject**, và đặt tất cả mã nguồn vào một package có tên "hust.soict ITE6.swing"

1.1. AWTAccumulator Class

```
1 package hust.soict.ITE6.swing;
2 import java.awt.*;
3 import java.awt.event.*;
4
5 public class AWTAccumulator extends Frame {
6     private TextField tfInput;
7     private TextField tfOutput;
8     private int sum = 0; // Accumulated sum, init to 0
9
10    // Constructor to setup the GUI components and event handlers;
11    public AWTAccumulator() {
12        setLayout(new GridLayout(2, 2));
13
14        add(new Label("Enter an Integer: "));
15
16        tfInput = new TextField(10);
17        add(tfInput);
18        tfInput.addActionListener(new TFInputListener());
19
20        add(new Label("The Accumulated Sum is: "));
21
22        tfOutput = new TextField(10);
23        tfOutput.setEditable(false);
24        add(tfOutput);
25
26        setTitle("AWT Accumulator");
27        setSize(350, 120);
28        setVisible(true);
29    }
30
31    public static void main(String[] args) {
32        new AWTAccumulator();
33    }
34
35    private class TFInputListener implements ActionListener {
36        @Override
37        public void actionPerformed(ActionEvent evt) {
38            int numberIn = Integer.parseInt(tfInput.getText());
39            sum += numberIn;
40            tfInput.setText("");
41            tfOutput.setText(sum+"");
42        }
43    }
44 }
45
```

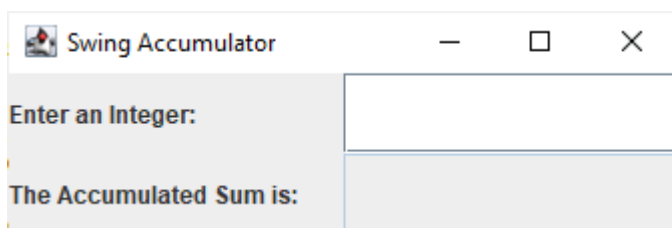
- Kết quả:



1.2. SwingAccumulator Class

```
1 package hust.soict.ITE6.swing;
2
3 import java.awt.*;
4 import java.awt.event.ActionEvent;
5 import java.awt.event.ActionListener;
6
7 import javax.swing.*;
8
9 public class SwingAccumulator extends JFrame {
10
11     private JTextField tfInput;
12     private JTextField tfOutput;
13     private int sum=0; //Accumulated sum, init to 0
14
15     // Constructor to setup the GUI components and event handlers
16     public SwingAccumulator() {
17         Container cp = getContentPane();
18         cp.setLayout(new GridLayout(2,2));
19
20         cp.add(new JLabel("Enter an Integer: "));
21
22         tfInput = new JTextField(10);
23         cp.add(tfInput);
24         tfInput.addActionListener(new TFInputListener());
25
26         cp.add(new JLabel("The Accumulated Sum is: "));
27
28         tfOutput = new JTextField(10);
29         tfOutput.setEditable(false);
30         cp.add(tfOutput);
31
32         setTitle("Swing Accumulator");
33         setSize(350, 120);
34         setVisible(true);
35     }
36     public static void main(String[] args) {
37         new SwingAccumulator();
38     }
39
40     private class TFInputListener implements ActionListener {
41         @Override
42         public void actionPerformed(ActionEvent evt) {
43             int numberIn = Integer.parseInt(tfInput.getText());
44             sum += numberIn;
45             tfInput.setText("");
46             tfOutput.setText(sum+"");
47         }
48     }
49 }
```

- Kết quả:



1.3. So sánh các thành phần giữa Swing và AWT

Lập trình với AWT và Swing có nhiều điểm tương đồng (bao gồm các thành phần container, thành phần giao diện và xử lý sự kiện). Tuy nhiên, có một số điểm khác biệt:

+ Container cấp cao nhất trong Swing và AWT: JFrame (Swing) vs Frame (AWT).

+ Tên lớp của các thành phần trong AWT và tên tương ứng trong Swing: AWT sử dụng các thành phần như Label, TextField, Button, trong khi Swing sử dụng các thành phần tương ứng là JLabel, JTextField, JButton.

2. Organizing Swing components with Layout Managers

- **Branch: `topic/aims-project/gui-project-swingcomponent-layoutmanagers`**
- Sử dụng JPanel làm container cấp thứ hai để tổ chức các thành phần

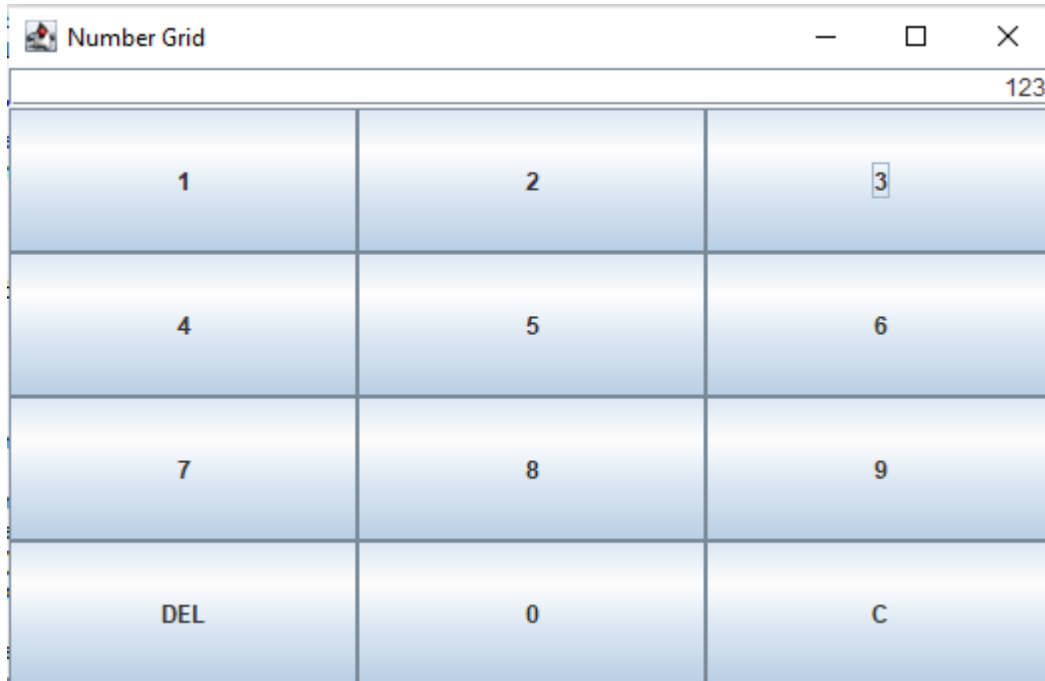
2.1. NumberGrid Class

```
1 package hust.soict.ITE6.swing;
2
3 import java.awt.*;
4 import java.awt.event.*;
5 import javax.swing.*;
6
7 public class NumberGrid extends JFrame {
8     private JButton[] btnNumbers = new JButton[10];
9     private JButton btnDelete, btnReset;
10    private JTextField tfDisplay;
11
12    public NumberGrid() {
13
14        tfDisplay = new JTextField();
15        tfDisplay.setComponentOrientation(ComponentOrientation.RIGHT_TO_LEFT);
16
17        JPanel panelButtons = new JPanel(new GridLayout(4, 3));
18        addButtons(panelButtons);
19
20        Container cp = getContentPane();
21        cp.setLayout(new BorderLayout());
22        cp.add(tfDisplay, BorderLayout.NORTH);
23        cp.add(panelButtons, BorderLayout.CENTER);
24
25        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
26        setTitle("Number Grid");
27        setSize(200, 200);
28        setVisible(true);
29    }
30
31    void addButtons(JPanel panelButtons) {
32        ButtonListener btnListener = new ButtonListener();
33        for(int i=1; i<=9; i++) {
34            btnNumbers[i] = new JButton(""+i);
35            panelButtons.add(btnNumbers[i]);
36            btnNumbers[i].addActionListener(btnListener);
37        }
38
39        btnDelete = new JButton("DEL");
40        panelButtons.add(btnDelete);
41        btnDelete.addActionListener(btnListener);
42
43        btnNumbers[0] = new JButton("0");
44        panelButtons.add(btnNumbers[0]);
45        btnNumbers[0].addActionListener(btnListener);
46
47        btnReset = new JButton("C");
48        panelButtons.add(btnReset);
49        btnReset.addActionListener(btnListener);
50    }
51
52    public static void main(String[] args) {
53        new NumberGrid();
54    }
55
56    private class ButtonListener implements ActionListener{
57        @Override
58        public void actionPerformed(ActionEvent e) {
59            String button = e.getActionCommand();
60            if (button.charAt(0) >= '0' && button.charAt(0) <= '9') {
61                tfDisplay.setText(tfDisplay.getText() + button);
62            }
63            else if (button.equals("DEL")) {
64                String deleteStr = tfDisplay.getText();
65                if (deleteStr.length()>0) {
66                    deleteStr = deleteStr.substring(0, deleteStr.length()-1);
67                }
68                tfDisplay.setText(deleteStr);
69            }
70            else if (button.equals("C")){
71                tfDisplay.setText("");
72            }
73        }
74    }
75 }
76
77 }
78 }
```

2.2. Giải thích và kết quả

- Lớp **NumberGrid** cho phép chúng ta nhập từng chữ số vào một ô hiển thị (**text field**) thông qua các nút số trên lưới. Ngoài ra, nó còn cung cấp chức năng: **Xóa chữ số cuối cùng** (DEL), **Xóa toàn bộ số** và bắt đầu lại (C)

- Các nút được thêm vào **panelButtons** thông qua phương thức **addButtons()**
- Hoàn thiện lớp bên trong **ButtonListener**
- Kết quả:



3. Create a graphical user interface for AIMS with Swing

- Trong ứng dụng **AIMS**, chúng ta sẽ triển khai ba màn hình:
 - **Màn hình “View Store”** (Xem cửa hàng) sử dụng **Swing**.
 - **Màn hình “View Cart”** (Xem giỏ hàng) sử dụng **JavaFX**.
 - **Màn hình “Update Store”** (Cập nhật cửa hàng) sử dụng **JavaFX**
- **Branch: topic/aims-project/view-store-screen**
 - Với **View Store Screen** mã nguồn sẽ được đặt trong package **hust.soict.ITE6.aims.screen** trong dự án **AimsProject**
- Đối với màn hình **View Store**, chúng ta sẽ sử dụng **BorderLayout**:
 - Ở thành phần **NORTH**, sẽ có **menu bar** và **header**
 - Ở thành phần **CENTER**, sẽ có một **panel** sử dụng **GridLayout**, mỗi ô trong lưới sẽ là một sản phẩm trong cửa hàng

3.1. View Store Screen

- Tạo StoreScreen Class:

```

1 package hust.soict ITE6.aims.screen;
2
3 import javax.swing.*;
4 import java.awt.*;
5 import java.util.*;
6
7 import hust.soict ITE6.aims.store.Store;
8 import hust.soict ITE6.aims.media.*;
9
10 public class StoreScreen extends JFrame{
11     private static Store store = new Store();
12
13     public static void initSetup() {
14         DigitalVideoDiscLHY dvd1 = new DigitalVideoDiscLHY("The Matrix", "Action", 15.50f, "Wachowskis", 136);
15         DigitalVideoDiscLHY dvd2 = new DigitalVideoDiscLHY("Inception", "Sci-Fi", 19.99f, "Christopher Nolan", 148);
16         DigitalVideoDiscLHY dvd3 = new DigitalVideoDiscLHY("The Dark Knight", "Action", 17.99f);
17         store.addMedia(dvd1);
18         store.addMedia(dvd2);
19         store.addMedia(dvd3);
20
21
22         Book book = new Book("Sherlock Holmes: The Complete Novels", "Mystery", 25.00f);
23         Book book1 = new Book("Becoming", "Biography", 30.00f);
24         Book book2 = new Book("The Great Gatsby", "Classic", 15.00f);
25         store.addMedia(book);
26         store.addMedia(book1);
27         store.addMedia(book2);
28
29
30         CompactDisc cd1 = new CompactDisc("Back In Black", "Rock", 12.99f, "AC/DC");
31         Track track1CD1 = new Track("Hells Bells", 6 * 50 + 12);
32         Track track2CD1 = new Track("Shoot to Thrill", 6*50 + 30);
33         cd1.addTrack(track1CD1);
34         cd1.addTrack(track2CD1);
35
36         CompactDisc cd2 = new CompactDisc("Lover", "Pop", 14.99f, "Taylor Swift");
37         Track track1CD2 = new Track("I Forgot That You Existed", 8 * 30);
38         Track track2CD2 = new Track("Death by a Thousand Cuts", 8 * 30 - 10);
39         cd2.addTrack(track1CD2);
40         cd2.addTrack(track2CD2);
41
42         CompactDisc cd3 = new CompactDisc("Future Nostalgia", "Pop", 16.99f, "Dua Lipa");
43         Track track1CD3 = new Track("Don't Start Now", 5 * 20 - 17);
44         Track track2CD3 = new Track("Physical", 8 * 40 + 2);
45         cd3.addTrack(track1CD3);
46         cd3.addTrack(track2CD3);
47
48         store.addMedia(cd1);
49         store.addMedia(cd2);
50         store.addMedia(cd3);
51     }
52
53     public StoreScreen(Store store) {
54         StoreScreen.store = store;
55         Container cp = getContentPane();
56         cp.setLayout(new BorderLayout());
57
58         cp.add(createNorth(), BorderLayout.NORTH);
59         cp.add(createCenter(), BorderLayout.CENTER);
60
61         setVisible(true);
62         setTitle("Store");
63         setSize(1024, 768);
64         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
65     }
66
67     public static void main(String[] args) {
68         initSetup();
69         new StoreScreen(store);
70     }
71
72     JPanel createNorth() {
73         JPanel north = new JPanel();
74         north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
75         north.add(createMenuBar());
76         north.add(createHeader());
77         return north;
78     }

```

```

79
80 JMenuBar createMenuBar() {
81     JMenu menu = new JMenu("Options");
82
83     JMenu smUpdateStore = new JMenu("Update Store");
84     smUpdateStore.add(new JMenuItem("Add Book"));
85     smUpdateStore.add(new JMenuItem("Add CD"));
86     smUpdateStore.add(new JMenuItem("Add DVD"));
87
88     menu.add(smUpdateStore);
89     menu.add(new JMenuItem("View store"));
90     menu.add(new JMenuItem("View cart"));
91
92     JMenuBar menuBar = new JMenuBar();
93     menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
94     menuBar.add(menu);
95
96     return menuBar;
97 }
98
99 JPanel createHeader() {
100     JPanel header = new JPanel();
101     header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));
102
103     JLabel title = new JLabel("AIMS");
104     title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
105     title.setForeground(Color.CYAN);
106
107     JButton cart = new JButton("View cart");
108     cart.setPreferredSize(new Dimension(100, 50));
109     cart.setMaximumSize(new Dimension(100, 50));
110
111     header.add(Box.createRigidArea(new Dimension(10, 10)));
112     header.add(title);
113     header.add(Box.createHorizontalGlue());
114     header.add(cart);
115     header.add(Box.createRigidArea(new Dimension(10, 10)));
116
117     return header;
118 }
119
120 JPanel createCenter() {
121
122     JPanel center = new JPanel();
123     center.setLayout(new GridLayout(3, 3, 2, 2));
124
125     ArrayList<Media> mediaInStore = store.getItemsInStore();
126     for (int i=0; i<9; i++) {
127         MediaStore cell = new MediaStore(mediaInStore.get(i));
128         center.add(cell);
129     }
130
131     return center;
132 }
133 }
134

```

- Thêm method **getItemsInStore** vào lớp **Store**

```

public ArrayList<Media> getItemsInStore() {
    return itemsInStore;
}

```

- Tạo **MediaStore** Class:

```

1 package hust.soict.ITE6.aims.screen;
2
3 import javax.swing.*;
4 import java.awt.*;
5 import java.awt.event.*;
6 import hust.soict.ITE6.aims.media.*;
7
8 public class MediaStore extends JPanel {
9     private Media media;
10
11     private static final String ADD_TO_CART_BUTTON_TEXT = "Add to cart";
12     private static final String PLAY_BUTTON_TEXT = "Play";
13     private static final String ADDED_TO_CART_MESSAGE = " has been added";
14

```



```

15 public MediaStore(Media media) {
16     this.media = media;
17     this.setLayout(new BorderLayout(this, BorderLayout.Y_AXIS));
18
19     JLabel title = new JLabel(media.getTitle());
20     title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 20));
21     title.setAlignmentX(CENTER_ALIGNMENT);
22
23     JLabel cost = new JLabel(String.format("%.2f$", media.getCost()));
24     cost.setAlignmentX(CENTER_ALIGNMENT);
25
26     JPanel container = new JPanel();
27     container.setLayout(new FlowLayout(FlowLayout.CENTER));
28
29     JButton addToCartButton = new JButton(ADD_TO_CART_BUTTON_TEXT);
30     addToCartButton.addActionListener(new ActionListener() {
31         public void actionPerformed(ActionEvent e) {
32             JOptionPane.showMessageDialog(MediaStore.this,
33                 media.getTitle() + ADDED_TO_CART_MESSAGE,
34                 "Add to cart",
35                 JOptionPane.INFORMATION_MESSAGE);
36         }
37     });
38     container.add(addToCartButton);
39
40     if (media instanceof Playable) {
41         JButton playButton = new JButton(PLAY_BUTTON_TEXT);
42         playButton.addActionListener(new ActionListener() {
43             public void actionPerformed(ActionEvent e) {
44                 JDialog dialog = new JDialog();
45                 dialog.setTitle(media.getTitle());
46                 dialog.setSize(400, 300);
47
48                 JLabel mediaLabel = new JLabel(media.playGUI());
49                 mediaLabel.setVerticalAlignment(JLabel.CENTER);
50                 mediaLabel.setHorizontalAlignment(JLabel.CENTER);
51
52                 JScrollPane scrollPane = new JScrollPane(mediaLabel);
53                 scrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);
54
55                 dialog.add(scrollPane);
56                 dialog.setVisible(true);
57             }
58         });
59         container.add(playButton);
60     }
61
62     this.add(Box.createVerticalGlue());
63     this.add(title);
64     this.add(cost);
65     this.add(Box.createVerticalGlue());
66     this.add(container);
67
68     this.setBorder(BorderFactory.createLineBorder(Color.BLACK));
69 }
70 }

```

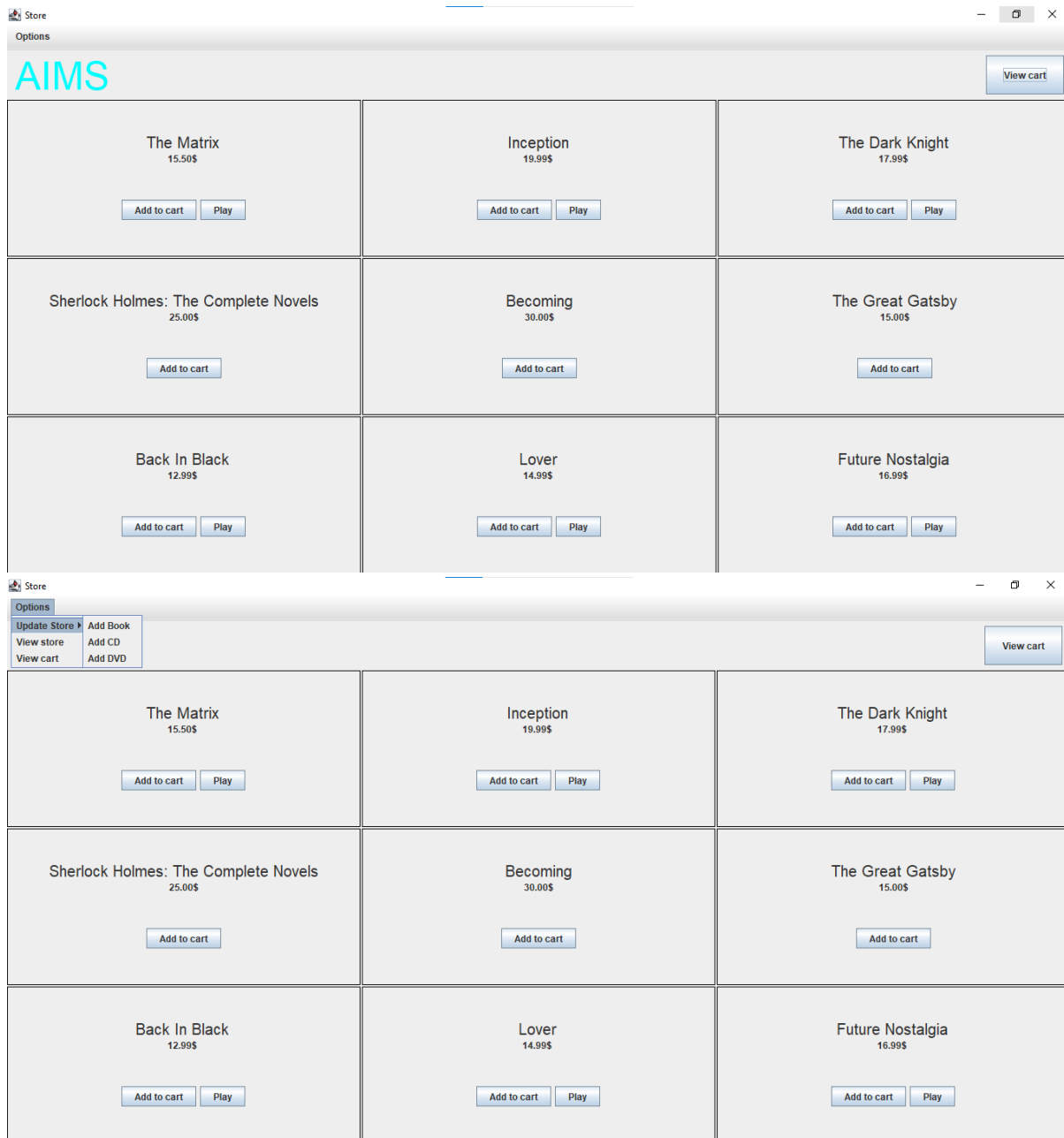
- Thêm method **playGUI()** cho lớp **Media**

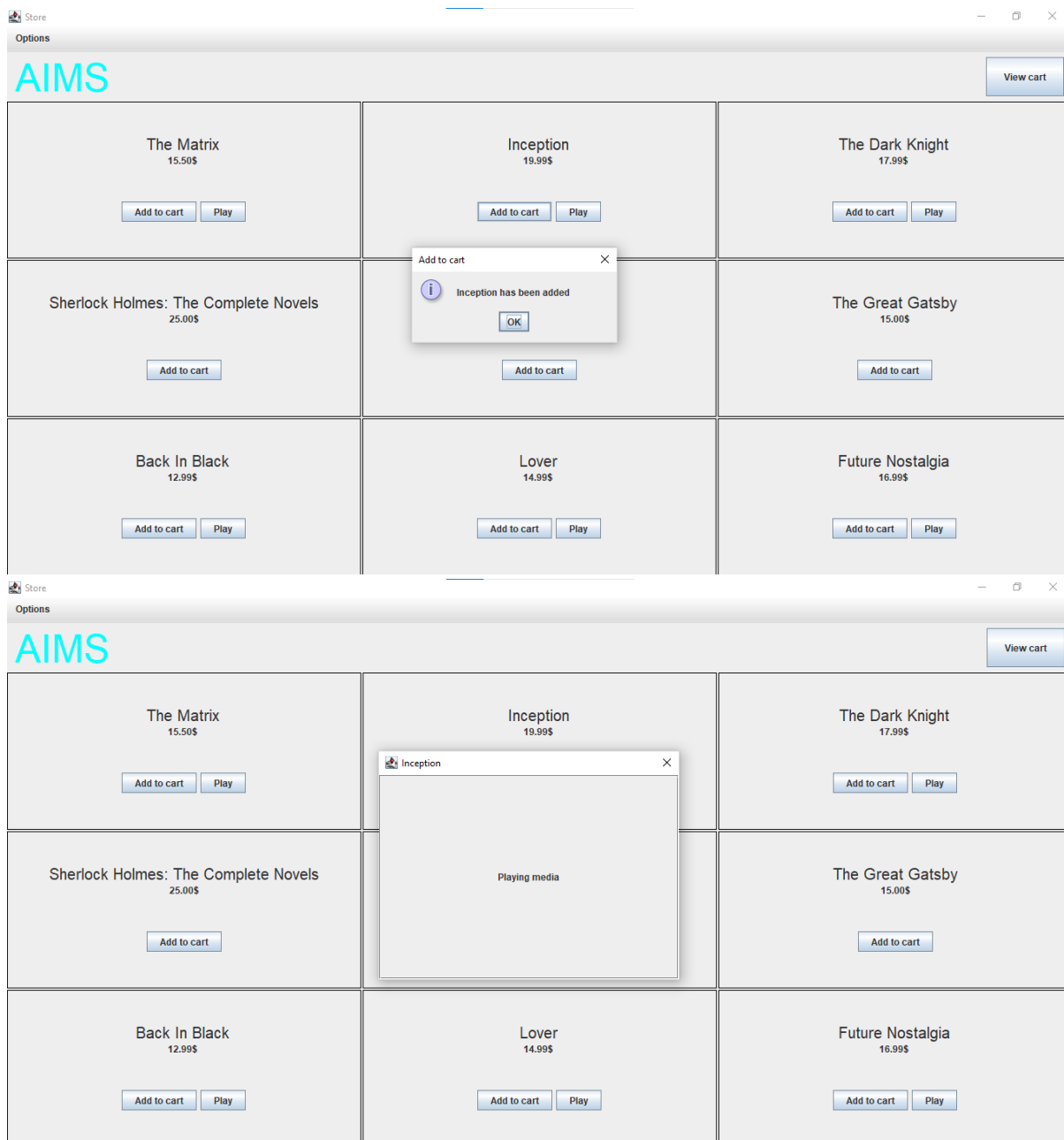
```

public String playGUI() {
    return "Playing media";
}

```

3.2. Adding more user interaction



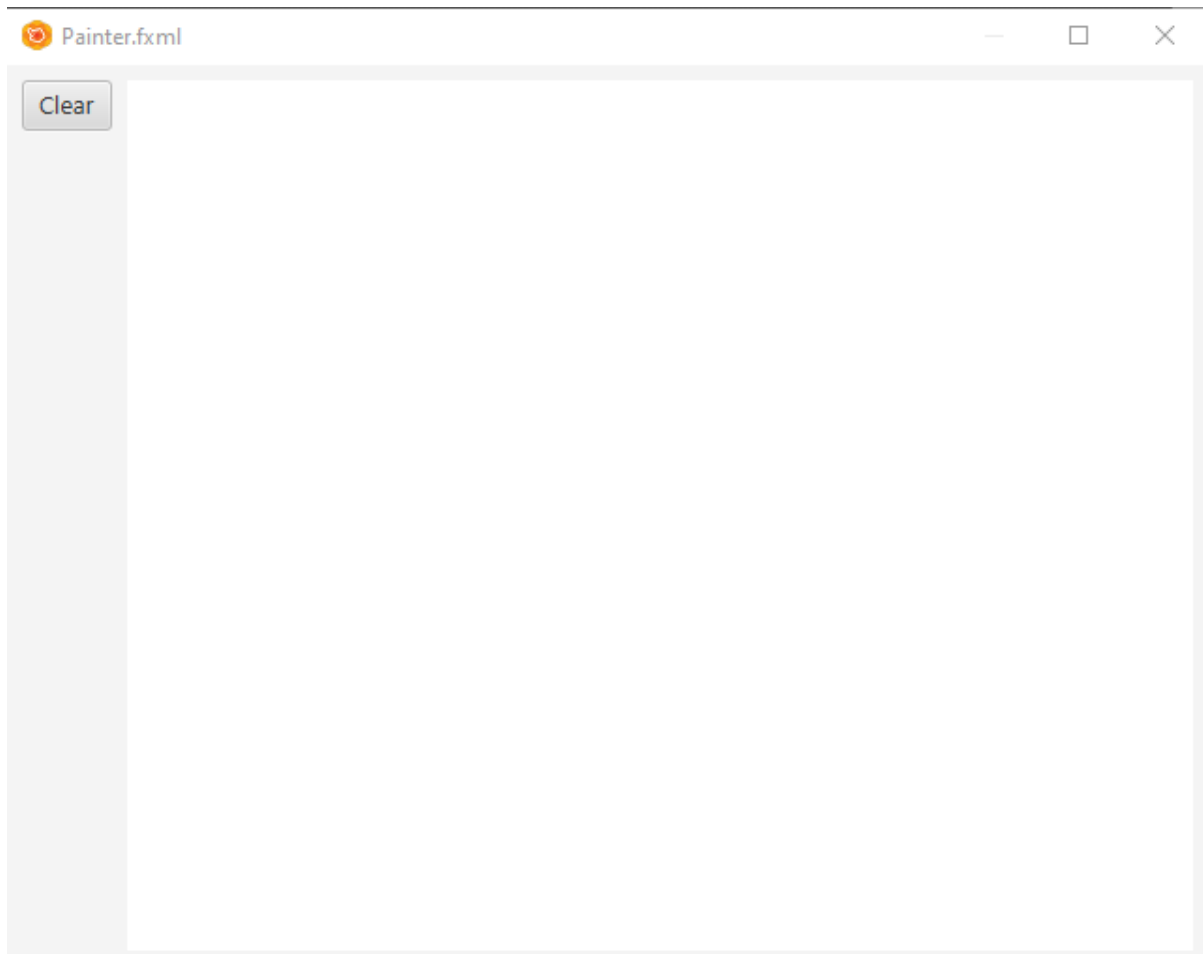


4. JavaFX API

- **Branch:** topic/gui-project/api-javafx

4.1. Create the FXML file

- Tạo file **Painter.fxml** trong **hust.soict.ITE6.javafx** và mở bằng **SceneBuilder**



4.2. Create the controller class

```
1 package hust.soict ITE6.javaafx;
2
3 import javafx.event.ActionEvent;
4 import javafx.fxml.FXML;
5 import javafx.scene.input.MouseEvent;
6 import javafx.scene.layout.Pane;
7 import javafx.scene.paint.Color;
8 import javafx.scene.shape.Circle;
9
10 public class PainterController {
11
12     @FXML
13     private Pane drawingAreaPane;
14
15     @FXML
16     void clearButtonPressed(ActionEvent event) {
17         drawingAreaPane.getChildren().clear();
18     }
19
20     @FXML
21     void drawingAreaMouseDragged(MouseEvent event) {
22         Circle newCircle = new Circle(event.getX(),
23             event.getY(), 4, Color.BLACK);
24         drawingAreaPane.getChildren().add(newCircle);
25     }
26
27 }
```

4.3. Create the application

```
1 package hust.soict.ITE6.javaafx;
2
3 import javafx.application.Application;
4 import javafx.fxml.FXMLLoader;
5 import javafx.scene.Parent;
6 import javafx.scene.Scene;
7 import javafx.stage.Stage;
8
9 public class Painter extends Application {
10
11     @Override
12     public void start(Stage stage) throws Exception {
13         Parent root = FXMLLoader.load(getClass().
14             .getResource("/hust/soict/ITE6/javaafx/Painter.fxml"));
15
16         Scene scene = new Scene(root);
17         stage.setTitle("Painter");
18         stage.setScene(scene);
19         stage.show();
20     }
21     public static void main(String[] args) {
22         Launch(args);
23     }
24 }
```

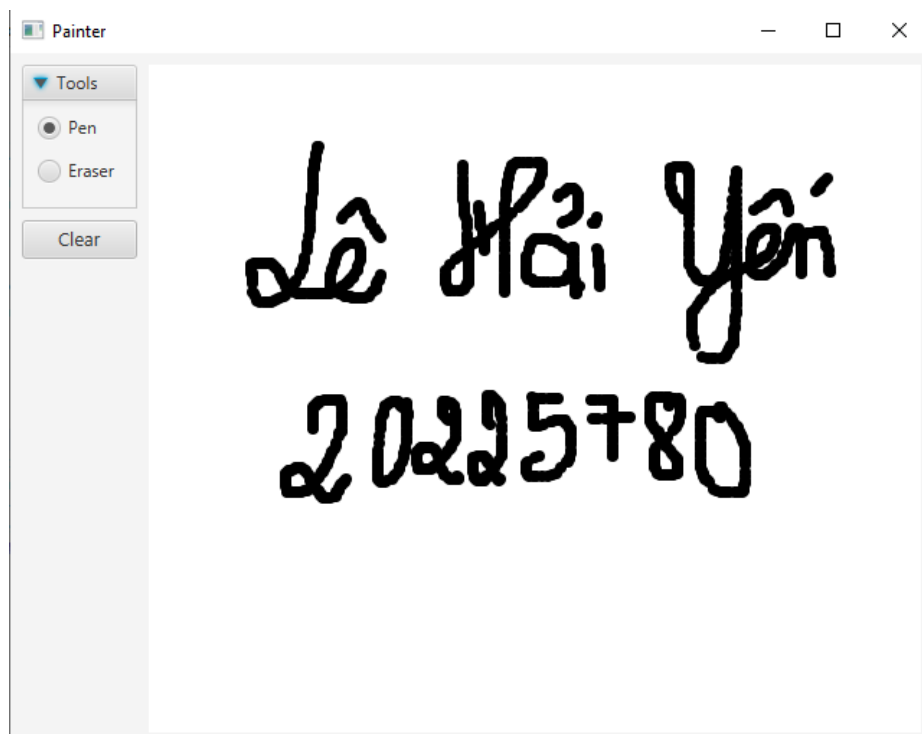
4.4. Practice exercise

- Sử dụng **TitledPane**
- **RadioButtons** tương ứng với **pen** và **eraser**
- thiết lập thuộc tính **Toggle Group** của các **RadioButton** sao cho chúng giống nhau, để chỉ một trong số chúng có thể được chọn tại một thời điểm
- Chỉnh sửa **PainterController** và kết quả thu được:

```

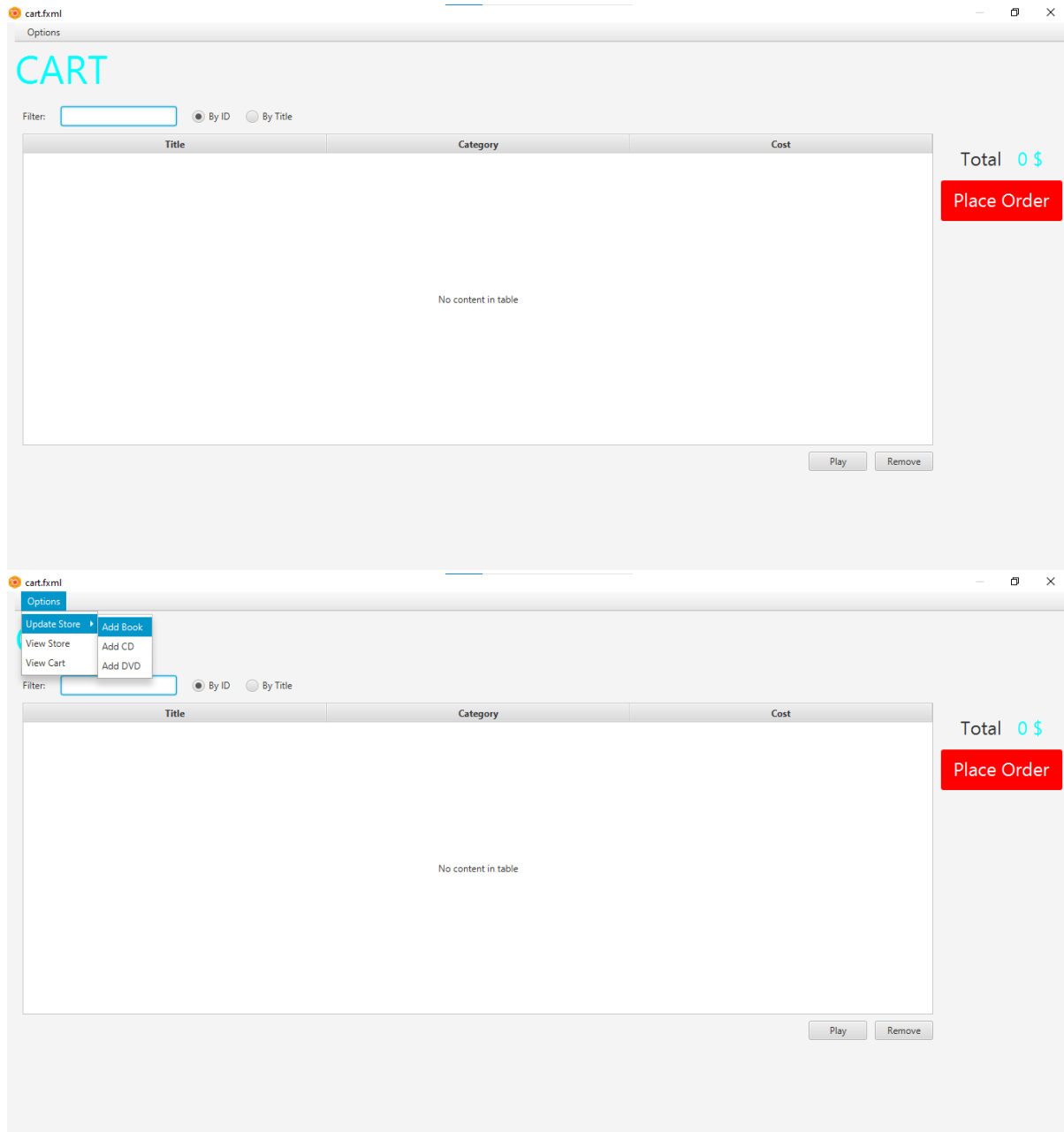
1 package hust.soict.ITE6.javafx;
2
3 import javafx.event.ActionEvent;
4 import javafx.fxml.FXML;
5 import javafx.scene.input.MouseEvent;
6 import javafx.scene.layout.Pane;
7 import javafx.scene.paint.Color;
8 import javafx.scene.shape.Circle;
9 import javafx.scene.control.RadioButton;
10 import javafx.scene.shape.Rectangle;
11 import javafx.scene.control.ToggleGroup;
12
13 public class PainterController {
14
15     @FXML
16     private Pane drawingAreaPane;
17
18     @FXML
19     private RadioButton pen;
20
21     @FXML
22     private RadioButton eraser;
23
24     @FXML
25     private ToggleGroup toolsGroup;
26
27     @FXML
28     void initialize() {
29         toolsGroup = new ToggleGroup();
30         pen.setToggleGroup(toolsGroup);
31         eraser.setToggleGroup(toolsGroup);
32         pen.setSelected(true);
33     }
34
35     @FXML
36     void clearButtonPressed(ActionEvent event) {
37         drawingAreaPane.getChildren().clear();
38     }
39
40     @FXML
41     void drawingAreaMouseDragged(MouseEvent event) {
42         Rectangle clipArea = new Rectangle(0, 0, drawingAreaPane.getWidth(), drawingAreaPane.getHeight());
43         drawingAreaPane.setClip(clipArea);
44         Color color;
45         if (pen.isSelected()) {
46             color = Color.BLACK;
47         } else if (eraser.isSelected()) {
48             color = Color.WHITE;
49         } else {
50             return;
51         }
52         Circle newCircle = new Circle(event.getX(), event.getY(), 4, color);
53         drawingAreaPane.getChildren().add(newCircle);
54     }
55
56
57 }

```



5. Setting up the View Cart Screen with ScreenBuilder

- **Branch:** topic/aims-project/view-cart-screen
- Các bài tiếp theo làm trong package **hust.soict.ITE6.aims.screen**
- Tạo **cart.fxml** trong **hust.soict.ITE6.aims.screen.view**



6. Integrating JavaFX into Swing application - The JFXPanel class

- CartScreen Class

```

public class CartScreen extends JFrame{
    private static CartLHY cart = new CartLHY();

    public CartScreen(CartLHY cart) {
        super();

        this.cart = cart;

        JFXPanel fxPanel = new JFXPanel();
        this.add(fxPanel);

        this.setTitle("Cart");
        this.setVisible(true);
        Platform.runLater(new Runnable() {
            @Override
            public void run() {
                try {
                    FXMLLoader loader = new FXMLLoader(getClass()
                        .getResource("/hust/soict/ITE6/aims/screen/view/cart.fxml"));

                    CartScreenController controller = new CartScreenController(cart);
                    loader.setController(controller);
                    Parent root = loader.load();
                    fxPanel.setScene(new Scene(root));
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

7. View the item in cart – JavaFX’s data-driven UI

- **CartScreenController Class** trong **hust.soict.ITE6.aims.screen.controller**

```

public class CartScreenController {
    private CartLHY cart;

    @FXML
    private TableView<Media> tblMedia;

    @FXML
    private TableColumn<Media, Float> colMediaCost;

    @FXML
    private TableColumn<Media, String> colMediaTitle;

    @FXML
    private TableColumn<Media, String> colMediaCategory;

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

    @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());
    }
}

```

- **Sửa Cart Class**

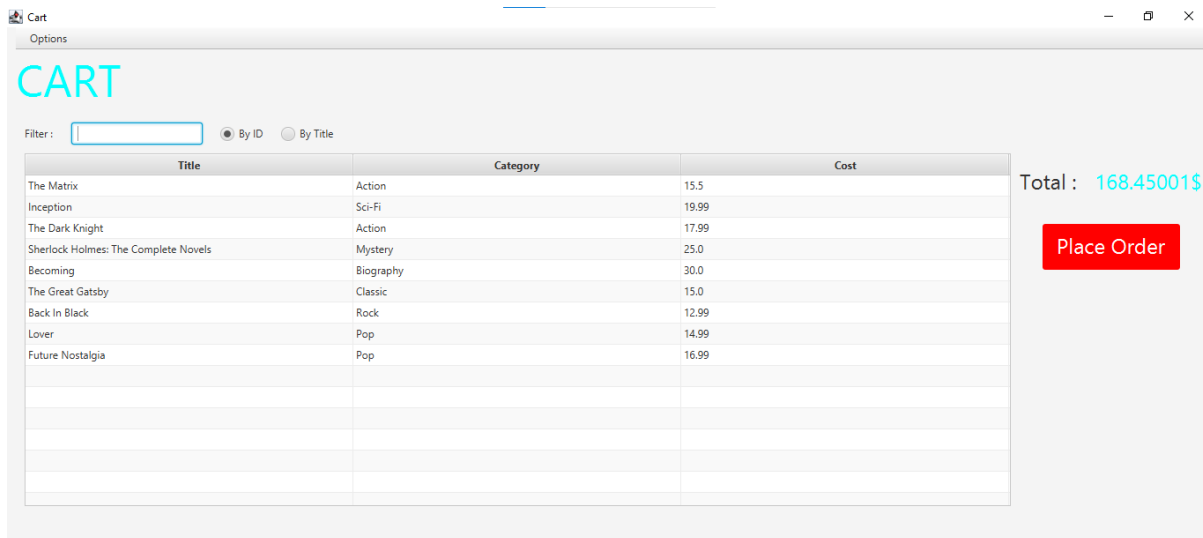
```

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

    public ObservableList<Media> getItemsOrdered(){
        return itemsOrdered;
    }
}

```

- **Kết quả:**



8. Updating buttons based on selected item in Table View – ChangeListener

- Thêm **fx:id** cho các nút (Nút **“Play”**: **btnPlay**; Nút **“Remove”**: **btnRemove**) trong **CartScreenController**

```
@FXML
private Button btnPlay;

@FXML
private Button btnRemove;

@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());

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

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

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

            private void updateButtonBar(Media media) {
                btnRemove.setVisible(true);
                if(media instanceof Playable) {
                    btnPlay.setVisible(true);
                } else {
                    btnPlay.setVisible(false);
                }
            }
        }
    );
}
```

9. Deleting a media

- Handle remove media

```

@FXML
void btnRemovePressed(ActionEvent event) {
    Media media = tblMedia.getSelectionModel().getSelectedItem();
    cart.removeMedia(media);
}

```

10. Filter items in cart – FilteredList

- Thêm thuộc tính **fx:id** cho các thành phần trong SceneBuilder và tạo ba thuộc tính tương ứng trong controller:
 - **TextField**: tfFilter.
 - **RadioButton** “Theo ID”: radioBtnFilterId.
 - **RadioButton** “Theo tiêu đề”: radioBtnFilterTitle
- Ở cuối phương thức **initialize()**, thêm một số mã để gắn một **ChangeListener** vào thuộc tính văn bản của **TextField**

```

@FXML
private TextField tfFilter;

@FXML
private RadioButton radioBtnFilterId;

@FXML
private RadioButton radioBtnFilterTitle;

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

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

        private void showFilteredMedia(String searchText) {
            FilteredList<Media> filteredMediaList = new FilteredList<>(cart.getItemsOrdered());

            if (!searchText.isEmpty() && radioBtnFilterId.isSelected()) {
                filteredMediaList.setPredicate(media -> String.valueOf(media.getId()).equals(searchText));
            } else if (!searchText.isEmpty() && radioBtnFilterTitle.isSelected()) {
                filteredMediaList.setPredicate(media -> media.getTitle().toLowerCase().contains(searchText.toLowerCase()));
            } else {
                filteredMediaList.setPredicate(null);
            }

            tblMedia.setItems(filteredMediaList);
        }
    });

```

11. Complete the Aims GUI application

11.1. Cart Screen

- Nút “Đặt hàng” (Place order)
- + Thêm method **placeOrder** trong lớp **CartLHY**

```

public String placeOrder() {
    if(itemsOrdered.isEmpty()) {
        return "Oops! It looks like your cart is empty.";
    } else {
        qtyOrdered = 0;
        itemsOrdered.clear();
        return "Success! Your order has been placed.\nYour cart is now empty.";
    }
}

```

```
@FXML
private Button placeOrder;

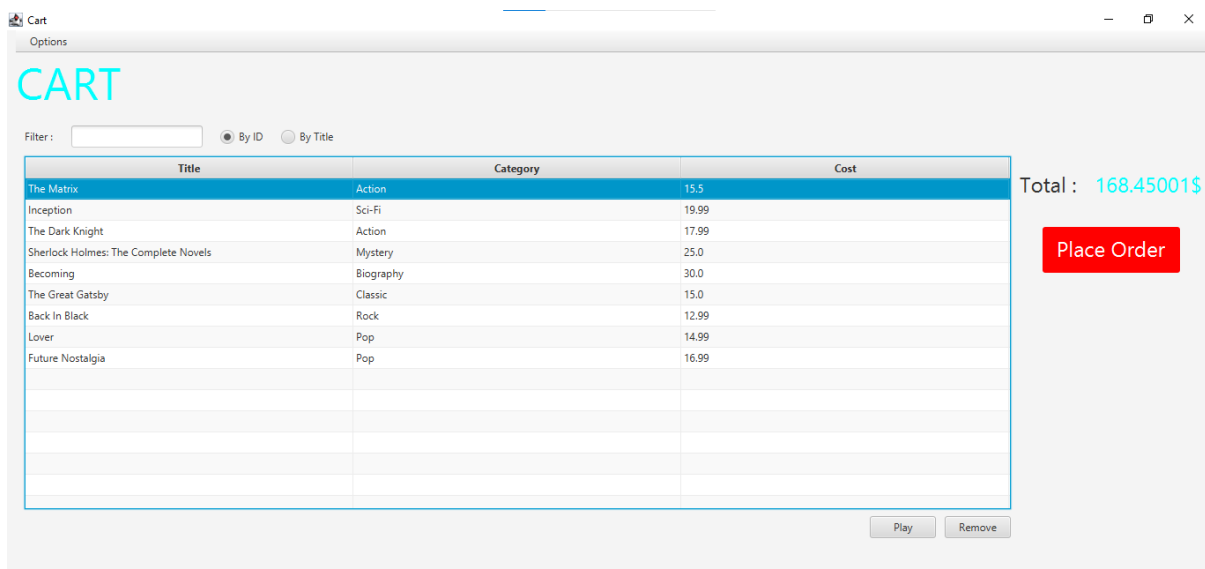
@FXML
void placeOrderPressed(ActionEvent event) {
    Alert alert = new Alert(Alert.AlertType.INFORMATION, cart.placeOrder());
    alert.setTitle("Order created");
    alert.setHeaderText(null);
    alert.showAndWait();
}
```

- Nút “Phát” (Play)

```
@FXML
void btnPlayPressed(ActionEvent event) {
    Media media = tblMedia.getSelectionModel().getSelectedItem();
    Alert alert = new Alert(Alert.AlertType.NONE, media.playGUI());
    alert.setTitle("Playing");
    alert.setHeaderText(null);
    alert.getDialogPane().getButtonTypes().add(ButtonType.OK);
    alert.showAndWait();
}
```

- Nhãn tổng chi phí phải được cập nhật khi có thay đổi trong giỏ hàng hiện tại (thêm/xóa)

```
costLabel.setText(cart.totalCost() + "$");
```



11.2. Update Store Screen

- **Branch: topic/aims-project/update-store-screen**
- Kết nối **StoreScreen** với **CartScreen**: Thêm **new Cart** vào **StoreScreen Class**; Xóa **Cart**, **initsetup** ở **CartScreen Class**

```
public class StoreScreen extends JFrame{
    private static Store store = new Store();
    private static CartLHY cart = new CartLHY();
}
```

- Ấn nút **Add to cart** thì add Media vào cart vừa được tạo ở **MediaStore Class**

```

JButton addToCartButton = new JButton(ADD_TO_CART_BUTTON_TEXT);
addToCartButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        JOptionPane.showMessageDialog(MediaStore.this,
            media.getTitle() + ADDED_TO_CART_MESSAGE,
            "Add to cart",
            JOptionPane.INFORMATION_MESSAGE);
    }
});
container.add(addToCartButton);

```

- **addMedia()** trong **CartLHY.java**

```

public String addMedia(Media media) {
    if (itemsOrdered.size() >= MAX_NUMBERS_ORDERED) {
        return "The cart is almost full!";
    } else if (itemsOrdered.contains(media)){
        return media.getTitle() + " is already in the cart!";
    } else {
        itemsOrdered.add(media);
        return "Added: " + media.getTitle();
    }
}

```

- Action **View Cart** hiện ra **CartScreen** với những cart đã được thêm

```

JButton cartLy = new JButton("View cart");
cartLy.setPreferredSize(new Dimension(100, 50));
cartLy.setMaximumSize(new Dimension(100, 50));
cartLy.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        new CartScreen(cart);
    }
});

```

- Tạo 3 lớp sau:
- AddDigitalVideoDiscToStoreScreen

```

1 package hust.soict.ITE6.aims.screen;
2
3 import javax.swing.*;
4 import java.awt.*;
5
6 import hust.soict.ITE6.aims.media.DigitalVideoDiscLHY;
7 import hust.soict.ITE6.aims.store.Store;
8
9
10 public class AddDigitalVideoDiscToStoreScreen extends JDialog {
11     private JFrame parentFrame;
12     private Store store;
13     private JTextField titleField;
14     private JTextField categoryField;
15     private JTextField priceField;
16     private JTextField directorField;
17     private JTextField lengthField;
18
19     public AddDigitalVideoDiscToStoreScreen(JFrame parent, Store store) {
20         super(parent, "Add Digital Video Disc", true);
21         this.store = store;
22         this.parentFrame = parent;
23
24         setLayout(new GridBagLayout());
25         GridBagConstraints gbc = new GridBagConstraints();
26         gbc.insets = new Insets(10, 10, 10, 10);
27         gbc.fill = GridBagConstraints.HORIZONTAL;
28
29         // Title
30         addLabel(gbc, 0, 0, "Title:");
31         titleField = createCenteredTextField();
32         addComponent(gbc, 1, 0, titleField);
33
34         // Category
35         addLabel(gbc, 0, 1, "Category:");
36         categoryField = createCenteredTextField();
37         addComponent(gbc, 1, 1, categoryField);
38
39         // Price
40         addLabel(gbc, 0, 2, "Price:");
41         priceField = createCenteredTextField();
42         addComponent(gbc, 1, 2, priceField);
43
44         // Director
45         addLabel(gbc, 0, 3, "Director:");
46         directorField = createCenteredTextField();
47         addComponent(gbc, 1, 3, directorField);
48
49         // Length
50         addLabel(gbc, 0, 4, "Length (minutes):");
51         lengthField = createCenteredTextField();
52         addComponent(gbc, 1, 4, lengthField);
53
54         // Buttons Panel
55         JPanel buttonPanel = new JPanel();
56         buttonPanel.setLayout(new FlowLayout(FlowLayout.CENTER, 10, 10));
57
58         JButton submitButton = createStyledButton("Submit");
59         submitButton.addActionListener(e -> submitDVD());
60         buttonPanel.add(submitButton);
61
62         JButton cancelButton = createStyledButton("Cancel");
63         cancelButton.addActionListener(e -> dispose());
64         buttonPanel.add(cancelButton);
65
66         gbc.gridx = 0;
67         gbc.gridy = 5;
68         gbc.gridwidth = 2;
69         add(buttonPanel, gbc);
70
71         pack();
72         setLocationRelativeTo(parent);
73         setVisible(true);
74     }
75
76     // Helper method to create centered text fields
77     private JTextField createCenteredTextField() {
78         JTextField textField = new JTextField();
79         textField.setHorizontalAlignment(JTextField.CENTER);
80         textField.setPreferredSize(new Dimension(200, 25));
81         return textField;
82     }
83
84     // Helper method to create styled buttons
85     private JButton createStyledButton(String text) {
86         JButton button = new JButton(text);
87         button.setPreferredSize(new Dimension(100, 30));
88         return button;
89     }
90
91     // Helper method to add labels with centered text
92     private void addLabel(GridBagConstraints gbc, int x, int y, String text) {
93         gbc.gridx = x;
94         gbc.gridy = y;
95         JLabel label = new JLabel(text, SwingConstants.LEFT);
96         add(label, gbc);

```

```

97     }
98
99     // Helper method to add components
100     private void addComponent(GridBagConstraints gbc, int x, int y, JComponent component) {
101         gbc.gridx = x;
102         gbc.gridy = y;
103         add(component, gbc);
104     }
105
106     private void submitDVD() {
107         // Validate input
108         if (!validateInput()) return;
109
110         try {
111             // Create DVD
112             String title = titleField.getText();
113             String category = categoryField.getText();
114             float cost = Float.parseFloat(priceField.getText());
115             String director = directorField.getText();
116             int length = Integer.parseInt(lengthField.getText());
117
118             DigitalVideoDiscLHY dvd = new DigitalVideoDiscLHY(
119                 title, category, cost, director, length
120             );
121
122             // Add to store
123             store.addMedia(dvd);
124
125             // Show success message
126             JOptionPane.showMessageDialog(this,
127                 "DVD added successfully!",
128                 "Success",
129                 JOptionPane.INFORMATION_MESSAGE);
130
131             // Refresh the store screen
132             if (parentFrame instanceof StoreScreen) {
133                 ((StoreScreen) parentFrame).refreshStoreScreen();
134             }
135
136             // Close dialog
137             dispose();
138         } catch (NumberFormatException e) {
139             JOptionPane.showMessageDialog(this,
140                 "Length must be a valid number",
141                 "Validation Error",
142                 JOptionPane.ERROR_MESSAGE);
143         }
144     }
145
146     private boolean validateInput() {
147         // Check if fields are empty
148         if (titleField.getText().isEmpty() ||
149             categoryField.getText().isEmpty() ||
150             priceField.getText().isEmpty() ||
151             directorField.getText().isEmpty() ||
152             lengthField.getText().isEmpty()) {
153             JOptionPane.showMessageDialog(this,
154                 "Please fill in all fields",
155                 "Validation Error",
156                 JOptionPane.ERROR_MESSAGE);
157             return false;
158         }
159
160         // Validate price
161         try {
162             Float.parseFloat(priceField.getText());
163         } catch (NumberFormatException e) {
164             JOptionPane.showMessageDialog(this,
165                 "Price must be a valid number",
166                 "Validation Error",
167                 JOptionPane.ERROR_MESSAGE);
168             return false;
169         }
170
171         return true;
172     }
173 }

```

- AddCompactDiscToStoreScreen

```

1 package hust.soict.ITE6.aims.screen;
2
3 import javax.swing.*;
4 import javax.swing.GroupLayout.Alignment;
5
6 import java.awt.*;
7
8 import hust.soict.ITE6.aims.media.CompactDisc;
9 import hust.soict.ITE6.aims.media.Track;
10 import hust.soict.ITE6.aims.store.Store;
11
12 public class AddCompactDiscToStoreScreen extends JDialog {
13     private Store store;
14     private JFrame parentFrame;
15     private JTextField titleField;
16     private JTextField categoryField;
17     private JTextField priceField;
18     private JTextField artistField;
19     private DefaultListModel<String> trackListModel;
20     private JList<String> trackList;
21     private JTextField trackNameField;
22     private JTextField trackLengthField;
23     private CompactDisc currentCD;
24
25     public AddCompactDiscToStoreScreen(JFrame parent, Store store) {
26         super(parent, "Add Compact Disc", true);
27         this.store = store;
28         this.parentFrame = parent;
29
30         // Use a more compact layout
31         setLayout(new GridBagLayout());
32         GridBagConstraints gbc = new GridBagConstraints();
33         gbc.insets = new Insets(5, 5, 5, 5);
34         gbc.fill = GridBagConstraints.HORIZONTAL;
35
36         // Title
37         addLabel(gbc, 0, 0, "Title:");
38         titleField = createCenteredTextField();
39         addComponent(gbc, 1, 0, titleField);
40
41         // Category
42         addLabel(gbc, 0, 1, "Category:");
43         categoryField = createCenteredTextField();
44         addComponent(gbc, 1, 1, categoryField);
45
46         // Price
47         addLabel(gbc, 0, 2, "Price:");
48         priceField = createCenteredTextField();
49         addComponent(gbc, 1, 2, priceField);
50
51         // Artist
52         addLabel(gbc, 0, 3, "Artist:");
53         artistField = createCenteredTextField();
54         addComponent(gbc, 1, 3, artistField);
55
56         // Tracks List
57         addLabel(gbc, 0, 4, "Tracks:");
58         trackListModel = new DefaultListModel<>();
59         trackList = new JList<>(trackListModel);
60         JScrollPane scrollPane = new JScrollPane(trackList);
61         scrollPane.setPreferredSize(new Dimension(200, 100));
62         addComponent(gbc, 1, 4, scrollPane);
63
64         // Track Name
65         addLabel(gbc, 0, 5, "Track Name:");
66         trackNameField = createCenteredTextField();
67         addComponent(gbc, 1, 5, trackNameField);

```

```

68
69 // Track Length
70 addLabel(gbc, 0, 6, "Track Length (seconds):");
71 trackLengthField = createCenteredTextField();
72 addComponent(gbc, 1, 6, trackLengthField);
73
74 // Add Track Button
75 JButton addTrackButton = createStyledButton("Add Track");
76 addTrackButton.addActionListener(e -> addTrack());
77 addComponent(gbc, 1, 7, addTrackButton);
78
79 // Submit Button
80 JButton submitButton = createStyledButton("Submit");
81 submitButton.addActionListener(e -> submitCD());
82 addComponent(gbc, 1, 8, submitButton);
83
84 // Cancel Button
85 JButton cancelButton = createStyledButton("Cancel");
86 cancelButton.addActionListener(e -> dispose());
87 addComponent(gbc, 1, 9, cancelButton);
88
89 pack();
90 setLocationRelativeTo(parent);
91 setVisible(true);
92 }
93
94 // Helper method to create centered text fields
95 private JTextField createCenteredTextField() {
96     JTextField textField = new JTextField();
97     textField.setHorizontalAlignment(JTextField.CENTER);
98     textField.setPreferredSize(new Dimension(200, 25));
99     return textField;
100 }
101
102 // Helper method to create styled buttons
103 private JButton createStyledButton(String text) {
104     JButton button = new JButton(text);
105     button.setPreferredSize(new Dimension(200, 30));
106     return button;
107 }
108
109 // Helper method to add labels with centered text
110 private void addLabel(GridBagConstraints gbc, int x, int y, String text) {
111     gbc.gridx = x;
112     gbc.gridy = y;
113     JLabel label = new JLabel(text, SwingConstants.CENTER);
114     add(label, gbc);
115 }
116
117 // Helper method to add components
118 private void addComponent(GridBagConstraints gbc, int x, int y, JComponent component) {
119     gbc.gridx = x;
120     gbc.gridy = y;
121     add(component, gbc);
122 }
123
124 // Rest of the methods remain the same as in the previous implementation
125 private void addTrack() {
126     String trackName = trackNameField.getText();
127     String trackLengthStr = trackLengthField.getText();
128
129     if (trackName.isEmpty() || trackLengthStr.isEmpty()) {
130         JOptionPane.showMessageDialog(this,
131             "Please enter track name and length",
132             "Track Error",
133             JOptionPane.ERROR_MESSAGE);
134     }
135     return;

```



```

135     }
136
137     try {
138         int trackLength = Integer.parseInt(trackLengthStr);
139         Track track = new Track(trackName, trackLength);
140
141         // If CD hasn't been created yet, create it first
142         if (currentCD == null) {
143             currentCD = new CompactDisc(
144                 titleField.getText(),
145                 categoryField.getText(),
146                 Float.parseFloat(priceField.getText()),
147                 artistField.getText()
148             );
149         }
150
151         // Add track to the CD
152         currentCD.addTrack(track);
153
154         // Update the track list model with track name and length
155         trackListModel.addElement(trackName + " (" + trackLength + " sec)");
156
157         // Clear track input fields
158         trackNameField.setText("");
159         trackLengthField.setText("");
160     } catch (NumberFormatException e) {
161         JOptionPane.showMessageDialog(this,
162             "Track length must be a number",
163             "Track Error",
164             JOptionPane.ERROR_MESSAGE);
165     }
166 }
167
168
169 private void submitCD() {
170     // Validate input
171     if (!validateInput()) return;
172
173     // If CD hasn't been created yet (no tracks added), create it
174     if (currentCD == null) {
175         currentCD = new CompactDisc(
176             titleField.getText(),
177             categoryField.getText(),
178             Float.parseFloat(priceField.getText()),
179             artistField.getText()
180         );
181     }
182
183     // Add to store
184     store.addMedia(currentCD);
185
186     // Show success message
187     JOptionPane.showMessageDialog(this,
188         "CD added successfully!",
189         "Success",
190         JOptionPane.INFORMATION_MESSAGE);
191
192     // Refresh the store screen
193     if (parentFrame instanceof StoreScreen) {
194         ((StoreScreen) parentFrame).refreshStoreScreen();
195     }
196
197     // Close dialog
198     dispose();
199 }
200

```

```

201 private boolean validateInput() {
202     // Check if fields are empty
203     if (titleField.getText().isEmpty() ||
204         categoryField.getText().isEmpty() ||
205         priceField.getText().isEmpty() ||
206         artistField.getText().isEmpty()) {
207         JOptionPane.showMessageDialog(this,
208             "Please fill in all fields",
209             "Validation Error",
210             JOptionPane.ERROR_MESSAGE);
211         return false;
212     }
213
214     // Validate price
215     try {
216         Float.parseFloat(priceField.getText());
217     } catch (NumberFormatException e) {
218         JOptionPane.showMessageDialog(this,
219             "Price must be a valid number",
220             "Validation Error",
221             JOptionPane.ERROR_MESSAGE);
222         return false;
223     }
224
225     // Check if tracks were added if needed
226     if (currentCD == null && trackListModel.isEmpty()) {
227         JOptionPane.showMessageDialog(this,
228             "Please add at least one track or create the CD",
229             "Validation Error",
230             JOptionPane.ERROR_MESSAGE);
231         return false;
232     }
233
234     return true;
235 }
236 }

```

- AddBookToStoreScreen

```

1 package hust.soict ITE6.aims.screen;
2
3 import javax.swing.*;
4 import java.awt.*;
5
6 import hust.soict ITE6.aims.media.Book;
7 import hust.soict ITE6.aims.store.Store;
8
9 public class AddBookToStoreScreen extends JDialog {
10     private static final long serialVersionUID = 1L;
11     private JFrame parentFrame;
12     private Store store;
13     private JTextField titleField;
14     private JTextField categoryField;
15     private JTextField priceField;
16     private JTextArea authorsArea;
17
18     public AddBookToStoreScreen(JFrame parent, Store store) {
19         super(parent, "Add Book", true);
20         this.store = store;
21         this.parentFrame = parent;
22
23         setLayout(new BorderLayout(10, 10));
24         setSize(600, 400);
25         setLocationRelativeTo(parent);
26
27         /// Create and configure panel for form fields
28         JPanel formPanel = new JPanel(new GridBagLayout());
29         formPanel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
30         GridBagConstraints gbc = new GridBagConstraints();
31         gbc.fill = GridBagConstraints.HORIZONTAL;
32         gbc.insets = new Insets(5, 5, 5, 5);
33
34         // Title Field
35         gbc.gridx = 0;
36         gbc.gridy = 0;
37         formPanel.add(new JLabel("Title:"), gbc);
38         titleField = new JTextField();
39         gbc.gridx = 1;
40         gbc.weightx = 1.0;
41         formPanel.add(titleField, gbc);
42
43         // Category Field
44         gbc.gridx = 0;
45         gbc.gridy = 1;
46         gbc.weightx = 0;
47         formPanel.add(new JLabel("Category:"), gbc);
48         categoryField = new JTextField();
49         gbc.gridx = 1;
50         gbc.weightx = 1.0;
51         formPanel.add(categoryField, gbc);
52
53         // Price Field
54         gbc.gridx = 0;
55         gbc.gridy = 2;
56         gbc.weightx = 0;
57         formPanel.add(new JLabel("Price:"), gbc);
58         priceField = new JTextField();
59         gbc.gridx = 1;
60         gbc.weightx = 1.0;
61         formPanel.add(priceField, gbc);
62
63         // Authors Field
64         gbc.gridx = 0;
65         gbc.gridy = 3;
66         gbc.weightx = 0;
67         formPanel.add(new JLabel("Authors (comma-separated):"), gbc);
68         authorsArea = new JTextArea(3, 20);
69         JScrollPane scrollPane = new JScrollPane(authorsArea);
70         gbc.gridx = 1;
71         gbc.weightx = 1.0;
72         formPanel.add(scrollPane, gbc);
73
74         add(formPanel, BorderLayout.CENTER);
75
76         // Buttons Panel
77         JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.RIGHT));
78         JButton submitButton = new JButton("Submit");
79         submitButton.addActionListener(e -> submitBook());
80         buttonPanel.add(submitButton);
81
82         JButton cancelButton = new JButton("Cancel");
83         cancelButton.addActionListener(e -> dispose());
84         buttonPanel.add(cancelButton);
85
86         add(buttonPanel, BorderLayout.SOUTH);
87
88         setVisible(true);
89     }
90
91     private void submitBook() {
92         // Validate input
93         if (!validateInput()) return;
94     }

```

```

95     // Create Book
96     String title = titleField.getText();
97     String category = categoryField.getText();
98     float price = Float.parseFloat(priceField.getText());
99
100     Book book = new Book(title, category, price);
101
102     // Add authors
103     String authorsText = authorsArea.getText();
104     if (!authorsText.isEmpty()) {
105         String[] authors = authorsText.split(",");
106         for (String author : authors) {
107             book.addAuthor(author.trim());
108         }
109     }
110
111     // Add to store
112     store.addMedia(book);
113
114     // Show success message
115     JOptionPane.showMessageDialog(this,
116         "Book added successfully!",
117         "Success",
118         JOptionPane.INFORMATION_MESSAGE);
119
120     // Refresh the store screen
121     if (parentFrame instanceof StoreScreen) {
122         ((StoreScreen) parentFrame).refreshStoreScreen();
123     }
124
125     // Close dialog
126     dispose();
127 }
128
129 private boolean validateInput() {
130     // Check if fields are empty
131     if (titleField.getText().isEmpty() ||
132         categoryField.getText().isEmpty() ||
133         priceField.getText().isEmpty()) {
134         JOptionPane.showMessageDialog(this,
135             "Please fill in all fields",
136             "Validation Error",
137             JOptionPane.ERROR_MESSAGE);
138         return false;
139     }
140
141     // Validate price
142     try {
143         Float.parseFloat(priceField.getText());
144     } catch (NumberFormatException e) {
145         JOptionPane.showMessageDialog(this,
146             "Price must be a valid number",
147             "Validation Error",
148             JOptionPane.ERROR_MESSAGE);
149         return false;
150     }
151
152     return true;
153 }
154
155 }

```

- Update StoreScreen

```

1 package hust.soict.ITE6.aims.screen;
2
3 import javax.swing.*;
4 import java.awt.*;
5 import java.util.*;
6 import java.awt.event.*;
7
8 import hust.soict.ITE6.aims.store.Store;
9 import hust.soict.ITE6.aims.cart.CartLHY;
10 import hust.soict.ITE6.aims.media.*;
11
12 public class StoreScreen extends JFrame {
13     private static final long serialVersionUID = 1L;
14     private Store store;
15     private CartLHY cart;
16     private JPanel centerPanel; // Declare centerPanel as a class field
17
18     // Constructor
19     public StoreScreen(Store store, CartLHY cart) {
20         this.store = store;
21         this.cart = cart;
22         Container cp = getContentPane();
23         cp.setLayout(new BorderLayout());
24
25         cp.add(createNorth(), BorderLayout.NORTH);
26
27         // Initialize centerPanel when creating the store screen
28         centerPanel = createCenter();
29         cp.add(centerPanel, BorderLayout.CENTER);
30
31         setVisible(true);
32         setTitle("Store");
33         setSize(1024, 768);
34     }
35
36     // Method to create the NORTH component
37     JPanel createNorth() {
38         JPanel north = new JPanel();
39         north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
40         north.add(createMenuBar());
41         north.add(createHeader());
42         return north;
43     }
44
45     // Method to create the Menu Bar
46     JMenuBar createMenuBar() {
47         JMenu menu = new JMenu("Options");
48
49         JMenu smUpdateStore = new JMenu("Update Store");
50
51         // Add Book menu item
52         JMenuItem addBookItem = new JMenuItem("Add Book");
53         addBookItem.addActionListener(e -> {
54             new AddBookToStoreScreen(this, store);
55         });
56         smUpdateStore.add(addBookItem);
57
58         // Add CD menu item
59         JMenuItem addCDItem = new JMenuItem("Add CD");
60         addCDItem.addActionListener(e -> {
61             new AddCompactDiscToStoreScreen(this, store);
62         });
63         smUpdateStore.add(addCDItem);
64
65         // Add DVD menu item
66         JMenuItem addDVDItem = new JMenuItem("Add DVD");
67         addDVDItem.addActionListener(e -> {
68             new AddDigitalVideoDiscToStoreScreen(this, store);
69         });

```

```

70     smUpdateStore.add(addDVDItem);
71
72     menu.add(smUpdateStore);
73     menu.add(new JMenuItem("View store"));
74
75     // Add View Cart menu item with action
76     JMenuItem viewCartItem = new JMenuItem("View cart");
77     viewCartItem.addActionListener(e -> {
78         new CartScreen(this.cart); // Open CartScreen with the current cart
79         dispose();
80     });
81     menu.add(viewCartItem);
82
83     JMenuBar menuBar = new JMenuBar();
84     menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
85     menuBar.add(menu);
86
87     return menuBar;
88 }
89
90 // Method to create the Header
91 JPanel createHeader() {
92     JPanel header = new JPanel();
93     header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));
94
95     JLabel title = new JLabel("AIMS");
96     title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
97     title.setForeground(Color.CYAN);
98
99     JButton cart = new JButton("View cart");
100    cart.addActionListener(e -> {
101        new CartScreen(this.cart); // Open CartScreen with the current cart
102    });
103    cart.setPreferredSize(new Dimension(100, 50));
104    cart.setMaximumSize(new Dimension(100, 50));
105
106    header.add(Box.createRigidArea(new Dimension(10, 10)));
107    header.add(title);
108    header.add(Box.createHorizontalGLue());
109    header.add(cart);
110    header.add(Box.createRigidArea(new Dimension(10, 10)));
111
112    return header;
113 }
114
115 public void refreshStoreScreen() {
116     // Remove existing center panel
117     Container cp = getContentPane();
118     cp.remove(centerPanel);
119
120     // Recreate center panel with updated store items
121     centerPanel = createCenter();
122     cp.add(centerPanel, BorderLayout.CENTER);
123
124     // Revalidate and repaint the frame
125     revalidate();
126     repaint();
127 }
128
129 JPanel createCenter() {
130     JPanel center = new JPanel();
131     center.setLayout(new GridLayout(4, 4, 2, 2));
132
133     ArrayList<Media> mediaInStore = store.getItemsInStore();
134     for (int i = 0; i < mediaInStore.size(); i++) {
135         MediaStore cell = new MediaStore(mediaInStore.get(i), cart);
136         center.add(cell);
137     }

```

```

138
139     return center;
140 }
141
142 public static void main(String[] args) {
143     // Example: Initialize Store and StoreScreen
144     Store store1 = new Store();
145     CartLHY cart = new CartLHY(); // Create a cart
146     DigitalVideoDiscLHY dvd1 = new DigitalVideoDiscLHY("The Matrix", "Action", 15.50f, "Wachowskis", 136);
147     DigitalVideoDiscLHY dvd2 = new DigitalVideoDiscLHY("Inception", "Sci-Fi", 19.99f, "Christopher Nolan", 148);
148     DigitalVideoDiscLHY dvd3 = new DigitalVideoDiscLHY("The Dark Knight", "Action", 17.99f);
149     store1.addMedia(dvd1);
150     store1.addMedia(dvd2);
151     store1.addMedia(dvd3);
152
153
154     Book book = new Book("Sherlock Holmes: The Complete Novels", "Mystery", 25.00f);
155     Book book1 = new Book("Becoming", "Biography", 30.00f);
156     Book book2 = new Book("The Great Gatsby", "Classic", 15.00f);
157     store1.addMedia(book);
158     store1.addMedia(book1);
159     store1.addMedia(book2);
160
161
162     CompactDisc cd1 = new CompactDisc("Back In Black", "Rock", 12.99f, "AC/DC");
163     Track track1CD1 = new Track("Hells Bells", 6 * 50 + 12);
164     Track track2CD1 = new Track("Shoot to Thrill", 6*50 + 30);
165     cd1.addTrack(track1CD1);
166     cd1.addTrack(track2CD1);
167
168     CompactDisc cd2 = new CompactDisc("Lover", "Pop", 14.99f, "Taylor Swift");
169     Track track1CD2 = new Track("I Forgot That You Existed", 8 * 30);
170     Track track2CD2 = new Track("Death by a Thousand Cuts", 8 * 30 - 10);
171     cd2.addTrack(track1CD2);
172     cd2.addTrack(track2CD2);
173
174     CompactDisc cd3 = new CompactDisc("Future Nostalgia", "Pop", 16.99f, "Dua Lipa");
175     Track track1CD3 = new Track("Don't Start Now", 5 * 20 - 17);
176     Track track2CD3 = new Track("Physical", 8 * 40 + 2);
177     cd3.addTrack(track1CD3);
178     cd3.addTrack(track2CD3);
179
180     store1.addMedia(cd1);
181     store1.addMedia(cd2);
182     store1.addMedia(cd3);
183
184     new StoreScreen(store1, cart);
185
186 }
187 }
188

```

- Update CartScreen

```

1 package hust.soict.ITE6.aims.screen;
2
3 import java.io.IOException;
4
5 import javax.swing.JFrame;
6 import hust.soict.ITE6.aims.media.*;
7 import hust.soict.ITE6.aims.cart.CartLHY;
8 import hust.soict.ITE6.aims.screen.controller.CartScreenController;
9 import javafx.application.Platform;
10 import javafx.embed.swing.JFXPanel;
11 import javafx.fxml.FXMLLoader;
12 import javafx.scene.Parent;
13 import javafx.scene.Scene;
14 import javax.naming.LimitExceededException;
15
16 public class CartScreen extends JFrame{
17     private static final long serialVersionUID = 1L;
18     private CartLHY cart;
19     public CartScreen(CartLHY cart) {
20         super();
21         this.cart = cart;
22
23         JFXPanel fxPanel = new JFXPanel();
24         this.add(fxPanel);
25         this.setTitle("Cart");
26         this.setSize(1024, 768);
27         this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
28         this.setVisible(true);
29
30         Platform.runLater(new Runnable() {
31             @Override
32             public void run() {
33                 try {
34                     FXMLLoader loader = new FXMLLoader(getClass().getResource("/hust/soict/ITE6/aims/screen/view/cart.fxml"));
35                     CartScreenController controller = new CartScreenController(cart);
36                     loader.setController(controller);
37                     Parent root = loader.load();
38                     fxPanel.setScene(new Scene(root));
39                 } catch (IOException e) {
40                     e.printStackTrace();
41                 }
42             }
43         });
44     }
45
46     public static void main(String[] args) {
47         CartLHY cart = new CartLHY();
48
49         DigitalVideoDiscLHY dvd1 = new DigitalVideoDiscLHY("The Matrix", "Action", 15.50f, "Wachowskis", 136);
50         DigitalVideoDiscLHY dvd2 = new DigitalVideoDiscLHY("Inception", "Sci-Fi", 19.99f, "Christopher Nolan", 148);
51         DigitalVideoDiscLHY dvd3 = new DigitalVideoDiscLHY("The Dark Knight", "Action", 17.99f);
52
53         Book book = new Book("Sherlock Holmes: The Complete Novels", "Mystery", 25.00f);
54         Book book1 = new Book("Becoming", "Biography", 30.00f);
55         Book book2 = new Book("The Great Gatsby", "Classic", 15.00f);
56
57         CompactDisc cd1 = new CompactDisc("Back In Black", "Rock", 12.99f, "AC/DC");
58         Track track1CD1 = new Track("Hells Bells", 6 * 50 + 12);
59         Track track2CD1 = new Track("Shoot to Thrill", 6 * 50 + 30);
60         cd1.addTrack(track1CD1);
61         cd1.addTrack(track2CD1);
62
63         CompactDisc cd2 = new CompactDisc("Lover", "Pop", 14.99f, "Taylor Swift");
64         Track track1CD2 = new Track("I Forgot That You Existed", 8 * 30);
65         Track track2CD2 = new Track("Death by a Thousand Cuts", 8 * 30 - 10);
66         cd2.addTrack(track1CD2);
67         cd2.addTrack(track2CD2);
68
69         CompactDisc cd3 = new CompactDisc("Future Nostalgia", "Pop", 16.99f, "Dua Lipa");
70         Track track1CD3 = new Track("Don't Start Now", 5 * 20 - 17);
71         Track track2CD3 = new Track("Physical", 8 * 40 + 2);
72         cd3.addTrack(track1CD3);
73         cd3.addTrack(track2CD3);
74
75         try {
76             cart.addMedia(dvd1);
77             cart.addMedia(dvd2);
78             cart.addMedia(dvd3);
79             cart.addMedia(book);
80             cart.addMedia(book1);
81             cart.addMedia(book2);
82             cart.addMedia(cd1);
83             cart.addMedia(cd2);
84             cart.addMedia(cd3);
85         } catch (LimitExceededException e) {
86             System.out.println(e.getMessage());
87         }
88
89         CartScreen cartScreen = new CartScreen(cart);
90         System.out.println("Hi");
91         cartScreen.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
92     }
93 }

```

- Update MediaStore


```

1 package hust.soict.ITE6.aims.screen;
2
3 import javax.naming.LimitExceededException;
4
11
12 public class MediaStore extends JPanel {
13     private static final String ADD_TO_CART_BUTTON_TEXT = "Add to cart";
14     private static final String PLAY_BUTTON_TEXT = "Play";
15     private static final String ADDED_TO_CART_MESSAGE = " has been added";
16
17     private static final long serialVersionUID = 1L;
18     private CartLHY cart; // Change this to be passed in constructor
19     private Media media;
20
21 public MediaStore(Media media, CartLHY cart) {
22     this.media = media;
23     this.cart = cart;
24     this.setLayout(new BorderLayout(this, BorderLayout.Y_AXIS));
25
26     JLabel title = new JLabel(media.getTitle());
27     title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 20));
28     title.setAlignmentX(CENTER_ALIGNMENT);
29
30     JLabel cost = new JLabel(String.format("%.2f$", media.getCost()));
31     cost.setAlignmentX(CENTER_ALIGNMENT);
32
33     JPanel container = new JPanel();
34     container.setLayout(new FlowLayout(FlowLayout.CENTER));
35
36     JButton addToCartButton = new JButton(ADD_TO_CART_BUTTON_TEXT);
37     addToCartButton.addActionListener(new ActionListener() {
38     public void actionPerformed(ActionEvent e) {
39         try {
40             String message = cart.addMedia(media);
41             JOptionPane.showMessageDialog(MediaStore.this,
42                 message,
43                 "Add to cart",
44                 JOptionPane.INFORMATION_MESSAGE);
45         } catch (LimitExceededException ex) {
46             JOptionPane.showMessageDialog(MediaStore.this,
47                 ex.getMessage(),
48                 "Error",
49                 JOptionPane.ERROR_MESSAGE);
50         }
51     }
52 });
53 container.add(addToCartButton);
54
55
56 // Play Button for Playable items (like DVD or CD)
57 if (media instanceof Playable) {
58     JButton playButton = new JButton(PLAY_BUTTON_TEXT);
59     playButton.addActionListener(e -> showPlayDialog((Playable) media));
60     container.add(playButton);
61 }
62
63 this.add(Box.createVerticalGLue());
64 this.add(title);
65 this.add(cost);
66 this.add(Box.createVerticalGLue());
67 this.add(container);
68 this.setBorder(BorderFactory.createLineBorder(Color.BLACK));
69 }
70
71 // Method to display Play information in a JDialog
72 private void showPlayDialog(Playable playableMedia) {
73     // Format the message for Playable items

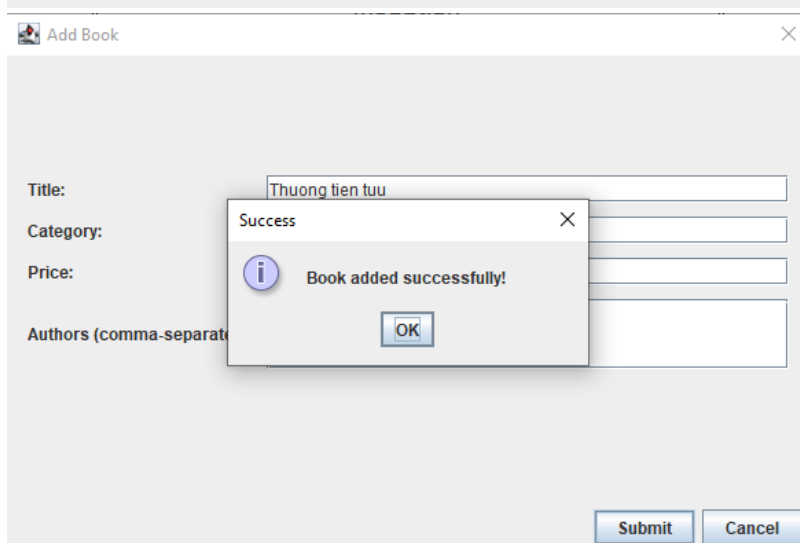
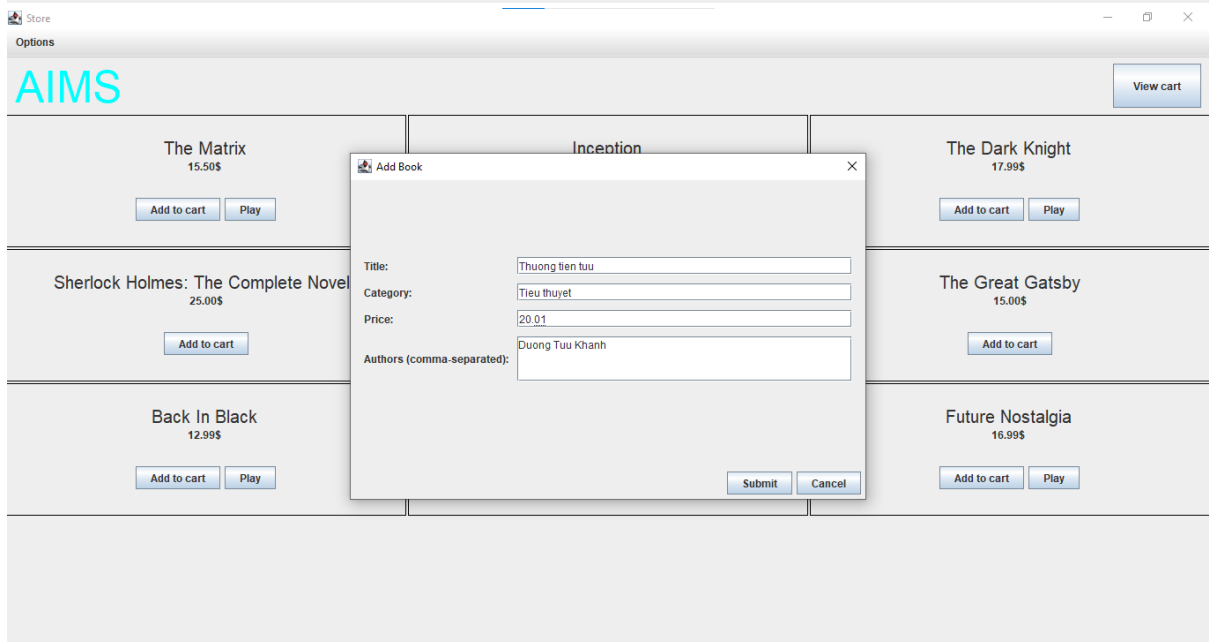
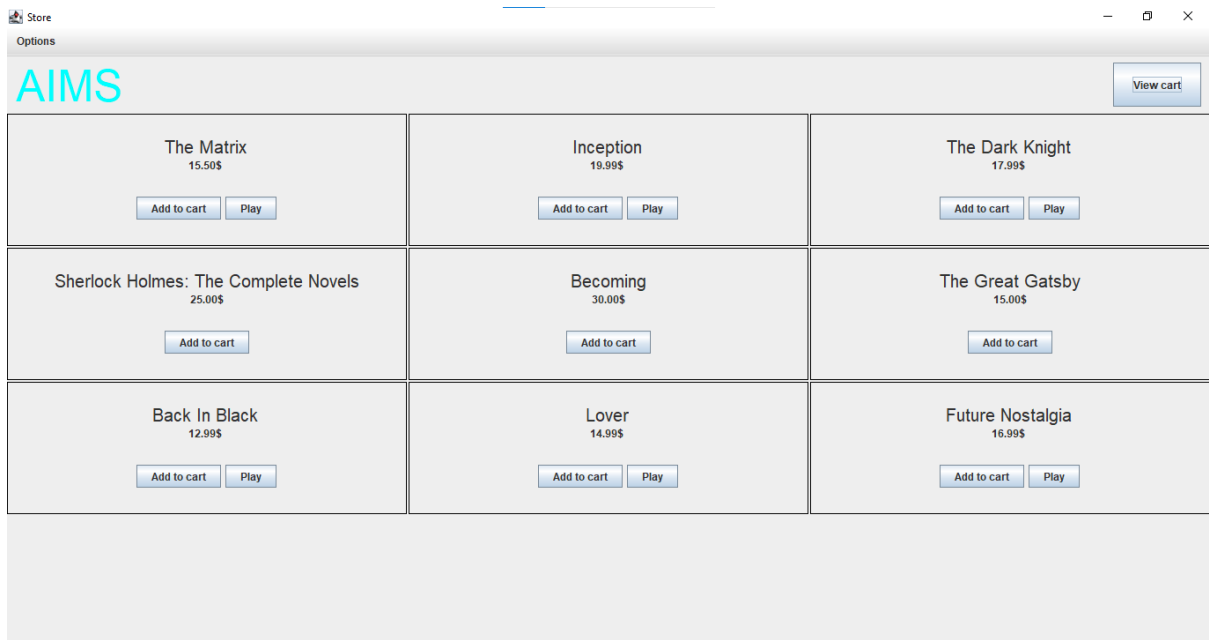
```

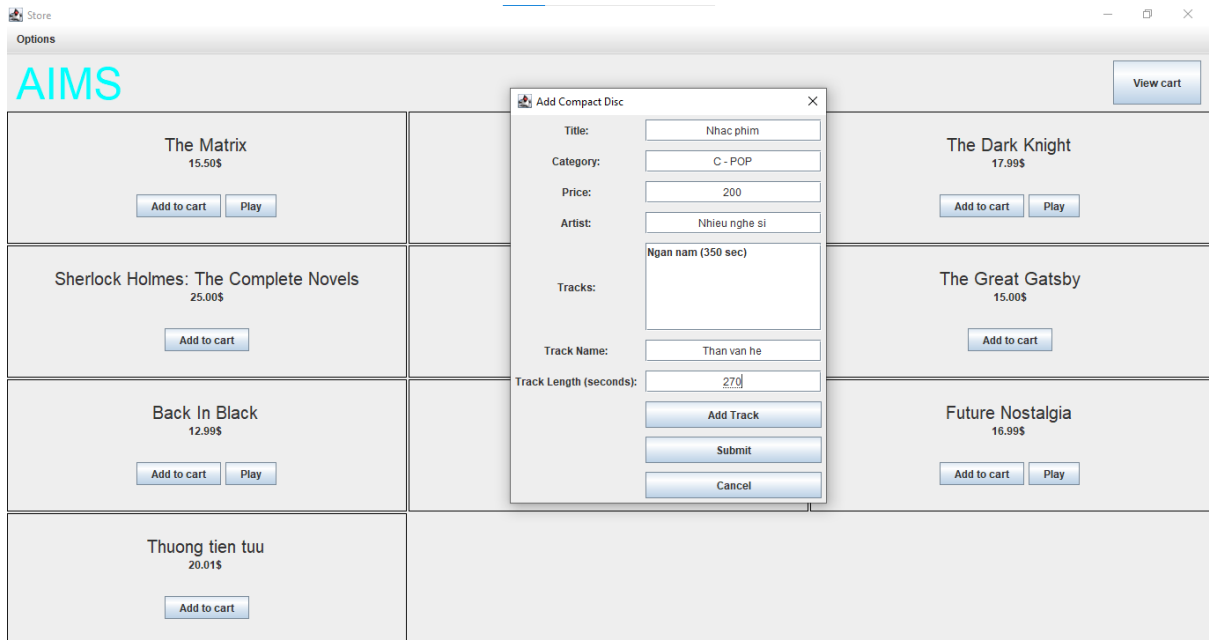
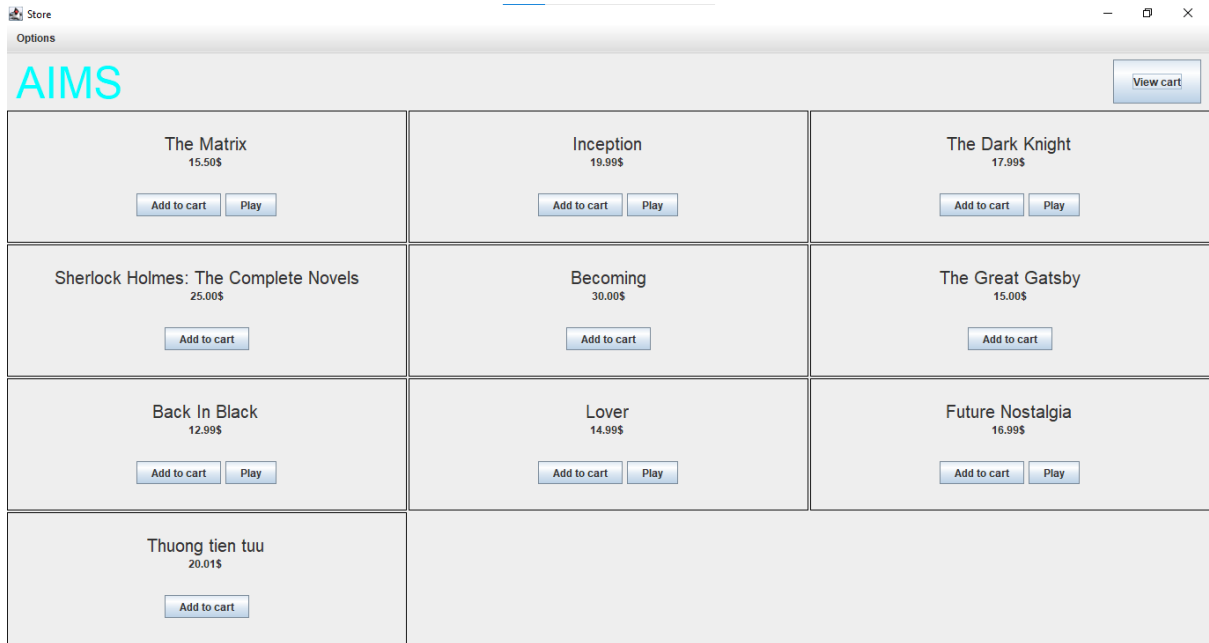
```

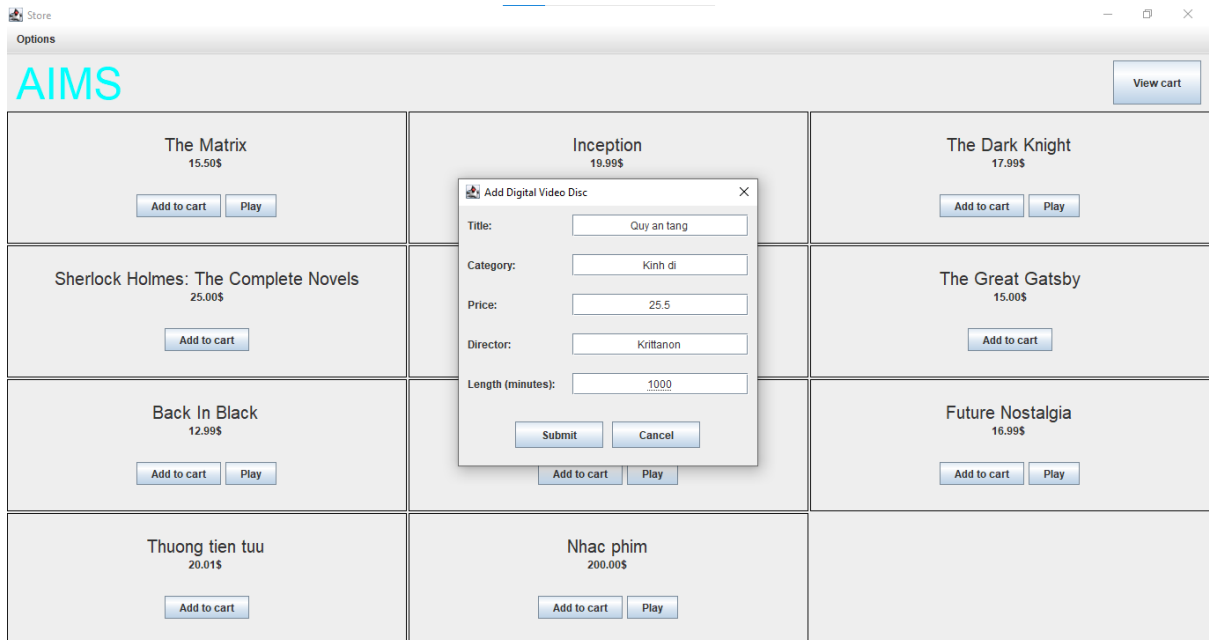
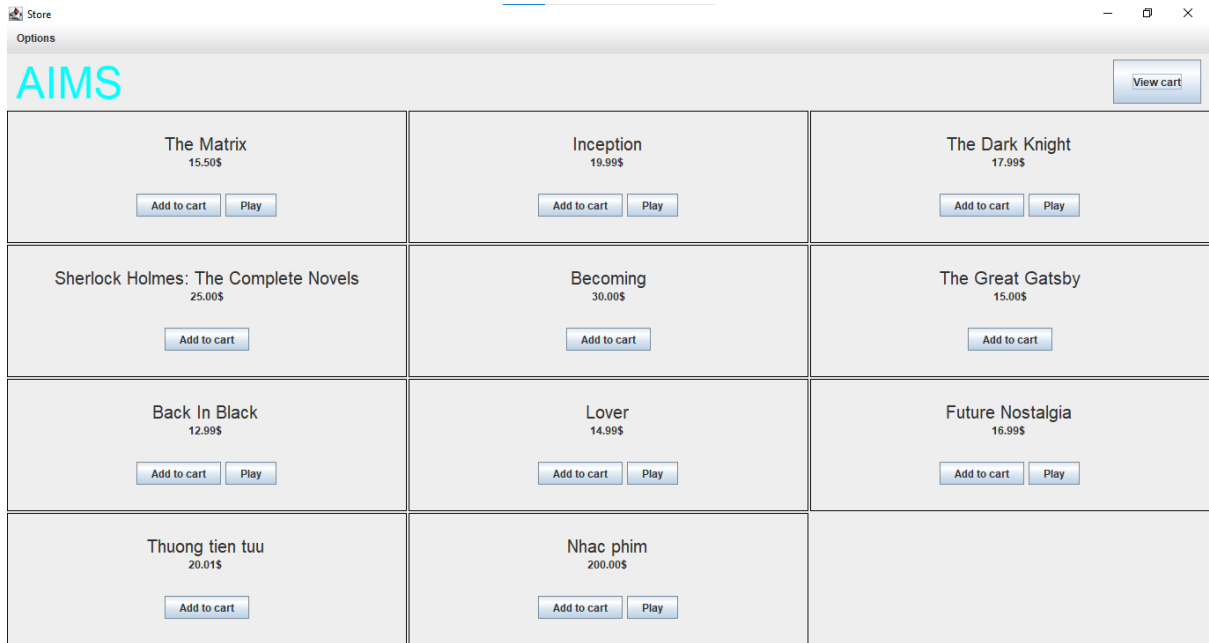
74 String playMessage = "Now Playing:\n" +
75     "Title: " + ((Media) playableMedia).getTitle() + "\n" +
76     "Length: " + ((Disc) playableMedia).getLength() + " minutes";
77
78 // Display the message in a JDialog
79 JDialog playDialog = new JDialog((JFrame) SwingUtilities.getWindowAncestor(this), "Play Media", true);
80 playDialog.setLayout(new BorderLayout());
81
82 // Create a panel to hold the JTextArea with padding
83 JPanel panel = new JPanel();
84 panel.setLayout(new BorderLayout());
85
86 // Text area for the message
87 JTextArea messageArea = new JTextArea(playMessage);
88 messageArea.setFont(new Font("Arial", Font.PLAIN, 16));
89 messageArea.setEditable(false);
90 messageArea.setWrapStyleWord(true);
91 messageArea.setLineWrap(true); // Enable word wrapping
92 messageArea.setBackground(new Color(240, 240, 240)); // Light background for readability
93 messageArea.setMargin(new Insets(10, 10, 10, 10)); // Add margin to prevent text from sticking to the edges
94 panel.add(new JScrollPane(messageArea), BorderLayout.CENTER);
95
96 playDialog.add(panel, BorderLayout.CENTER);
97
98 // Close Button with better styling
99 JPanel buttonPanel = new JPanel();
100 buttonPanel.setLayout(new FlowLayout(FlowLayout.CENTER)); // Center the button
101 JButton closeButton = new JButton("Close");
102 closeButton.setFont(new Font("Arial", Font.BOLD, 14));
103 closeButton.setBackground(new Color(60, 179, 113)); // A pleasant green background
104 closeButton.setForeground(Color.WHITE); // White text
105 closeButton.addActionListener(e -> playDialog.dispose());
106
107 buttonPanel.add(closeButton);
108
109 playDialog.add(buttonPanel, BorderLayout.SOUTH);
110
111 // Set the size and location of the dialog
112 playDialog.setSize(400, 200);
113 playDialog.setLocationRelativeTo(this);
114 playDialog.setVisible(true);
115 }

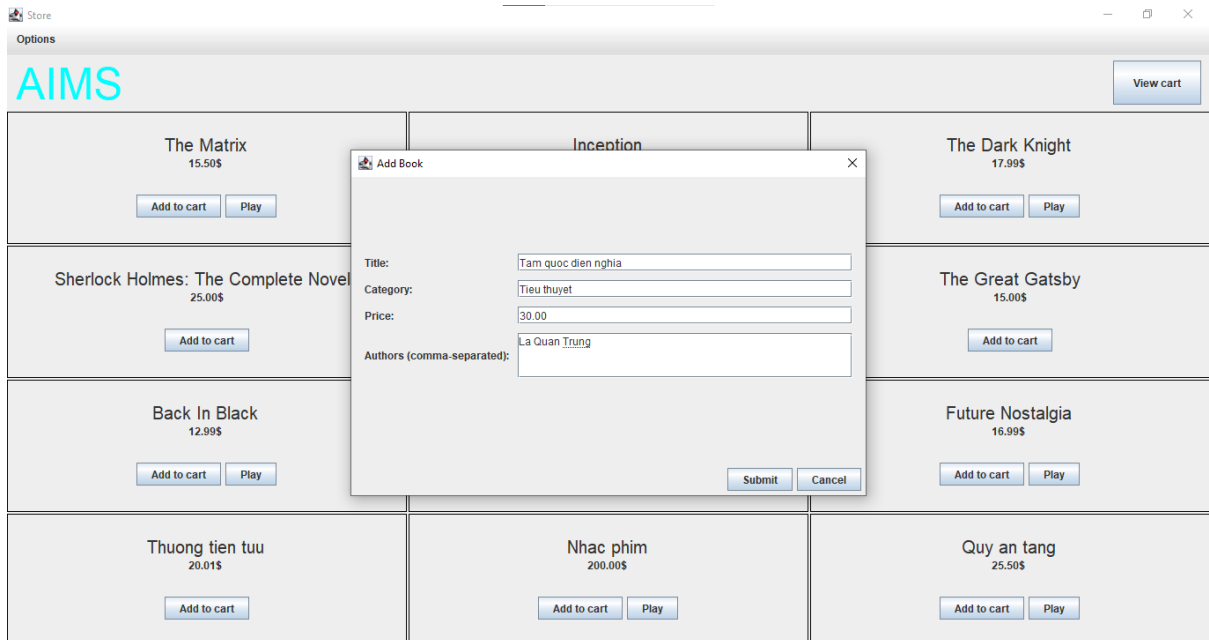
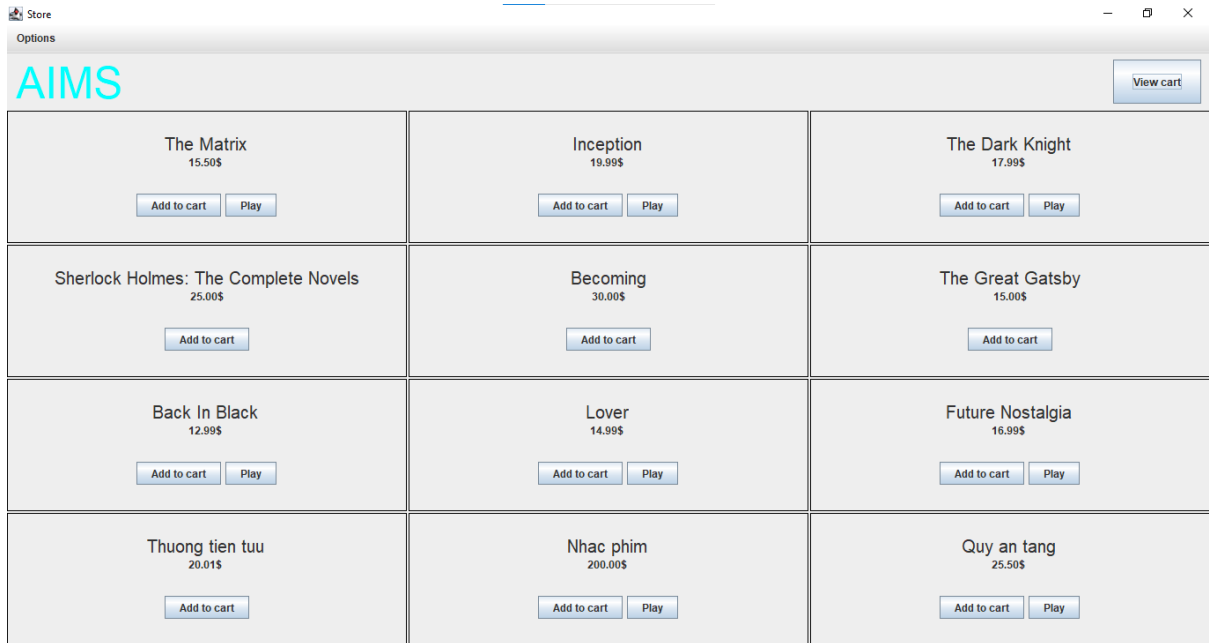
```

- Complete the Aims GUI application









Store

Options

AIMS

View cart

<div>The Matrix 15.50\$</div> <div>Add to cartPlay</div>	<div>Inception 19.99\$</div> <div>Add to cartPlay</div>	<div>The Dark Knight 17.99\$</div> <div>Add to cartPlay</div>	<div>Sherlock Holmes: The Complete No... 25.00\$</div> <div>Add to cart</div>
<div>Becoming 30.00\$</div> <div>Add to cart</div>	<div>The Great Gatsby 15.00\$</div> <div>Add to cart</div>	<div>Back In Black 12.99\$</div> <div>Add to cartPlay</div>	<div>Lover 14.99\$</div> <div>Add to cartPlay</div>
<div>Future Nostalgia 16.99\$</div> <div>Add to cartPlay</div>	<div>Thuong tien tuu 20.01\$</div> <div>Add to cart</div>	<div>Nhac phim 200.00\$</div> <div>Add to cartPlay</div>	<div>Quy an tang 25.50\$</div> <div>Add to cartPlay</div>
<div>Tam quoc dien nghia 30.00\$</div> <div>Add to cart</div>			

Cart

Options

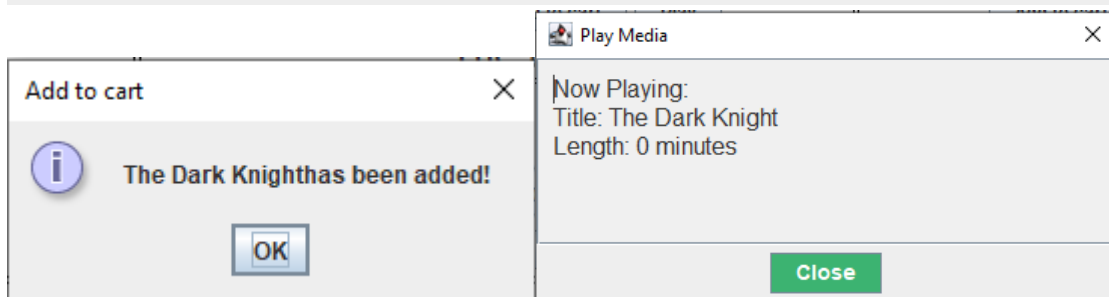
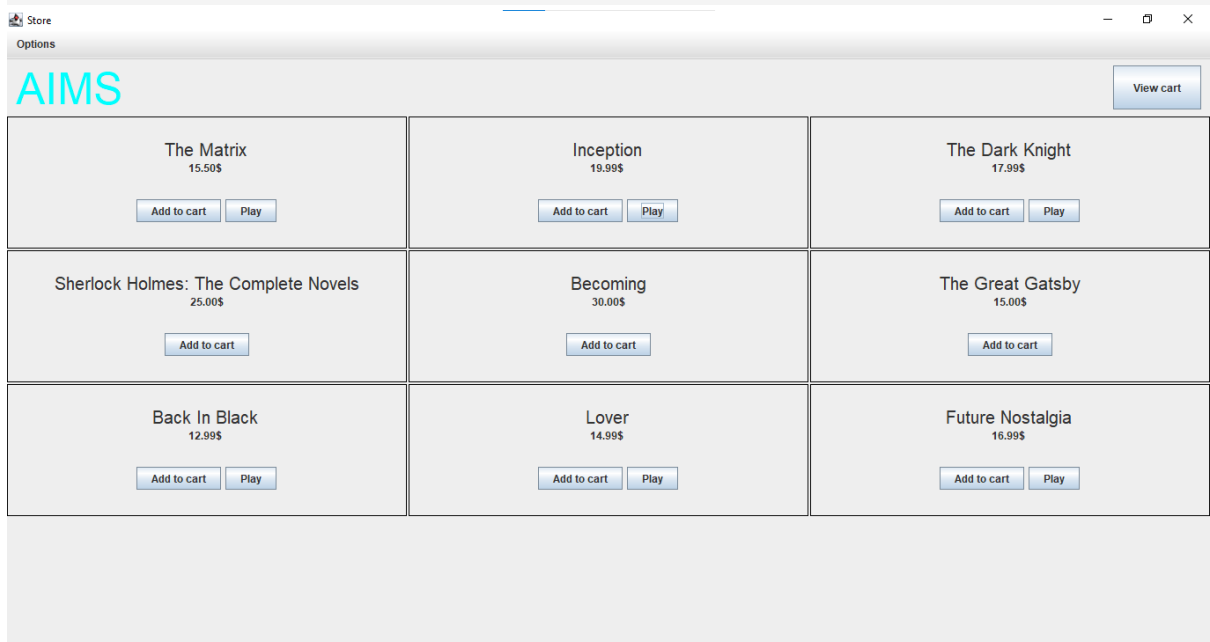
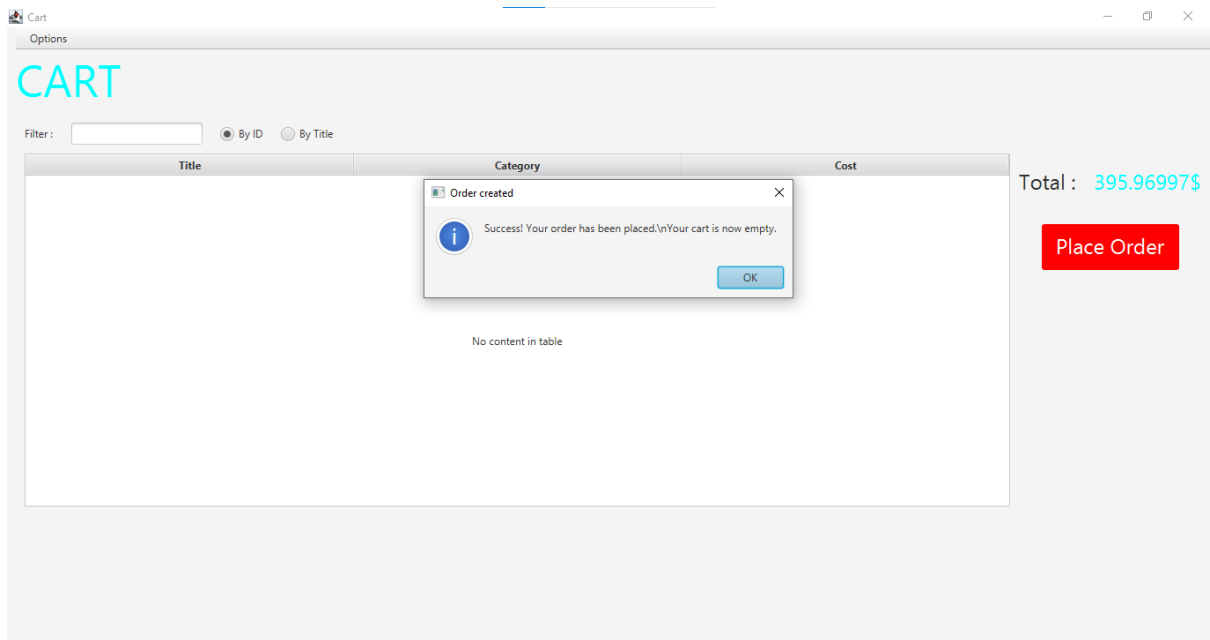
CART

Filter : ☒ By ID ☐ By Title

Title	Category	Cost
The Matrix	Action	15.5
Inception	Sci-Fi	19.99
Future Nostalgia	Pop	16.99
Tam quoc dien nghia	Tieu thuyet	30.0
Thuong tien tuu	Tieu thuyet	20.01
Nhac phim	C - POP	200.0
Lover	Pop	14.99
Sherlock Holmes: The Complete Novels	Mystery	25.0
Quy an tang	Kinh di	25.5
Back In Black	Rock	12.99
The Great Gatsby	Classic	15.0

Total : 395.96997\$

Place Order



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

- Branch: topic/aims-project/exceptions

- Cart Class


```

1 package hust.soict ITE6.aims.cart;
2 import java.util.*;
3
4 import hust.soict ITE6.aims.media.Media;
5 import javafx.collections.FXCollections;
6 import javafx.collections.ObservableList;
7
8 import javax.naming.LimitExceededException;
9
10 public class CartLHY {
11     public static final int MAX_NUMBERS_ORDERED = 20;
12     private ObservableList<Media> itemsOrdered = FXCollections.observableArrayList();
13
14     public ObservableList<Media> getItemsOrdered(){
15         return itemsOrdered;
16     }
17
18     public String addMedia(Media media) throws LimitExceededException {
19         if (itemsOrdered.size() >= MAX_NUMBERS_ORDERED) {
20             throw new LimitExceededException("ERROR: The number of media has reached its limit");
21         } else if (itemsOrdered.contains(media)){
22             return media.getTitle() + " is already in the cart!";
23         } else {
24             itemsOrdered.add(media);
25             return (media.getTitle() + "has been added! ");
26         }
27     }

```

- CartTest

```

1 package hust.soict ITE6.test.cart;
2 import hust.soict ITE6.aims.cart.CartLHY;
3
4
5 public class CartTest {
6     public static void main(String[] args) throws LimitExceededException {
7
8         CartLHY cart = new CartLHY();
9
10         DigitalVideoDiscLHY dvd1 = new DigitalVideoDiscLHY("The Lion King",
11             "Animation", 19.95f, "Roger Allers", 87);
12         cart.addMedia(dvd1);
13         DigitalVideoDiscLHY dvd2 = new DigitalVideoDiscLHY("Star War",
14             "Science Fiction", 24.95f, "George Lucas", 87);
15         cart.addMedia(dvd2);
16         DigitalVideoDiscLHY dvd3 = new DigitalVideoDiscLHY("Aladin",
17             "Animation", 18.99f);
18         cart.addMedia(dvd3);
19         DigitalVideoDiscLHY dvd4 = new DigitalVideoDiscLHY("Aladin",
20             "Animation", 18.99f);
21         cart.addMedia(dvd4);
22         cart.print();
23         cart.searchByID(1);
24         cart.searchByTitle("Star");
25         cart.searchByTitle("Baby");
26     }
27 }

```

- MediaStore

```

JButton addToCartButton = new JButton(ADD_TO_CART_BUTTON_TEXT);
addToCartButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        try {
            String message = cart.addMedia(media);
            JOptionPane.showMessageDialog(MediaStore.this,
                message,
                "Add to cart",
                JOptionPane.INFORMATION_MESSAGE);
        } catch (LimitExceededException ex) {
            JOptionPane.showMessageDialog(MediaStore.this,
                ex.getMessage(),
                "Error",
                JOptionPane.ERROR_MESSAGE);
        }
    }
});
container.add(addToCartButton);

```

- AimsLHY

```

Media media = store.search(title);
if (media != null) {
    try {
        cart.addMedia(media);
    } catch (LimitExceededException e) {
        e.printStackTrace();
    }
    foundToAdd = true;
} else {
    System.out.println("***MEDIA NOT FOUND***");
}
}

```

13. Create a class which inherits from Exception

13.1. Create new class named PlayerException

- Tạo 1 package **hust.soict.ITE6.aims.exception**, sau đó tạo lớp **PlayerException**

```

1 package hust.soict.ITE6.aims.exception;
2
3 public class PlayerException extends Exception {
4     public PlayerException(String message) {
5         super(message);
6     }
7 }

```

13.2. Raise the PlayerException in the play() method

- **Media:**

```

public String playGUI() throws PlayerException {
    return "Playing media";
}

```

- **DigitalVideoDiscLHY:**

```

public String playGUI() throws PlayerException {
    if (this.getLength() > 0) {
        return "Playing DVD: " + this.getTitle() + "\n" +
            "DVD length: " + formatDuration(this.getLength());
    } else {
        throw new PlayerException("ERROR: DVD length is non-positive!");
    }
}

```

- Track:

```

public String formatDuration(int durationInSeconds) {
    Duration duration = Duration.ofSeconds(durationInSeconds);
    return String.format("%02d:%02d", duration.toMinutes(),
        duration.minusMinutes(duration.toMinutes()).getSeconds());
}

public String playGUI() throws PlayerException {
    if (this.getLength() > 0) {
        return "Playing track: " + this.getTitle() + "\n" +
            "Track length: " + formatDuration(this.getLength());
    } else {
        throw new PlayerException("ERROR: Track length is non-positive!");
    }
}

```

13.3. Update play() in the Playable interface

```

package hust.soict.ITE6.aims.media;

import hust.soict.ITE6.aims.exception.PlayerException;

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

```

13.4. Update play() in CompactDisc

```

public String playGUI() throws PlayerException {
    if (this.getLength() > 0) {
        String output = "Playing CD: " + this.getTitle() + "\n" +
            "CD length: " + formatDuration(this.getLength()) + "\n" + "\n";
        for (Track track : tracks) {
            try {
                output += track.playGUI() + "\n";
            } catch (PlayerException e) {
                output += track.getTitle() + "\n" + e.getMessage();
            }
        }
        return output;
    } else {
        throw new PlayerException("ERROR: CD length is non-positive!");
    }
}

```

- Update **CartScreenController**

```

@FXML
void btnPlayPressed(ActionEvent event) {
    Media media = tblMedia.getSelectionModel().getSelectedItem();
    Alert alert;
    try {
        alert = new Alert(Alert.AlertType.NONE, media.playGUI());
        alert.setTitle("Playing");
        alert.setHeaderText(null);
        alert.getDialogPane().getButtonTypes().add(ButtonType.OK);
        alert.showAndWait();
    } catch (PlayerException e) {
        alert = new Alert(Alert.AlertType.ERROR, e.getMessage());
        alert.setTitle("ERROR");
        alert.setHeaderText(null);
        alert.showAndWait();
    }
}

```

14. Update the Aims class

Vì **Exception** được **throw** tại **method playGUI()** nên không ảnh hưởng lớp **Aims**.

15. Modify the equals() method of Media class

Tránh gặp **NullPointerException** và **ClassCastException**

```

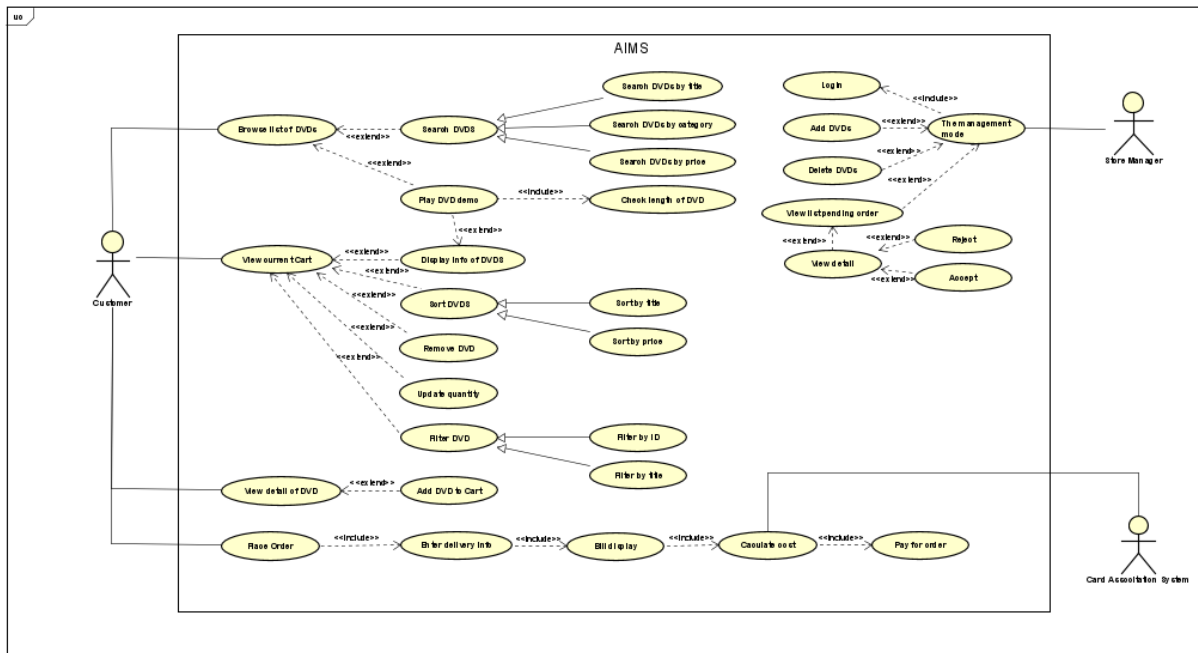
public String formatDuration(int durationInSeconds) {
    Duration duration = Duration.ofSeconds(durationInSeconds);
    return String.format("%02d:%02d", duration.toMinutes(), duration.minusMinutes(duration.toMinutes()).getSeconds());
}

@Override
public boolean equals(Object obj) {
    if (obj == this) {
        return true;
    }
    if (obj == null || !(obj instanceof Media)) {
        return false;
    }
    Media otherMedia = (Media) obj;
    return this.getTitle() != null && this.getTitle().equals(otherMedia.getTitle());
}

```

II. UML Diagram

1. Use-Case Diagram



2. Class Diagram (Final Lab05)

