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

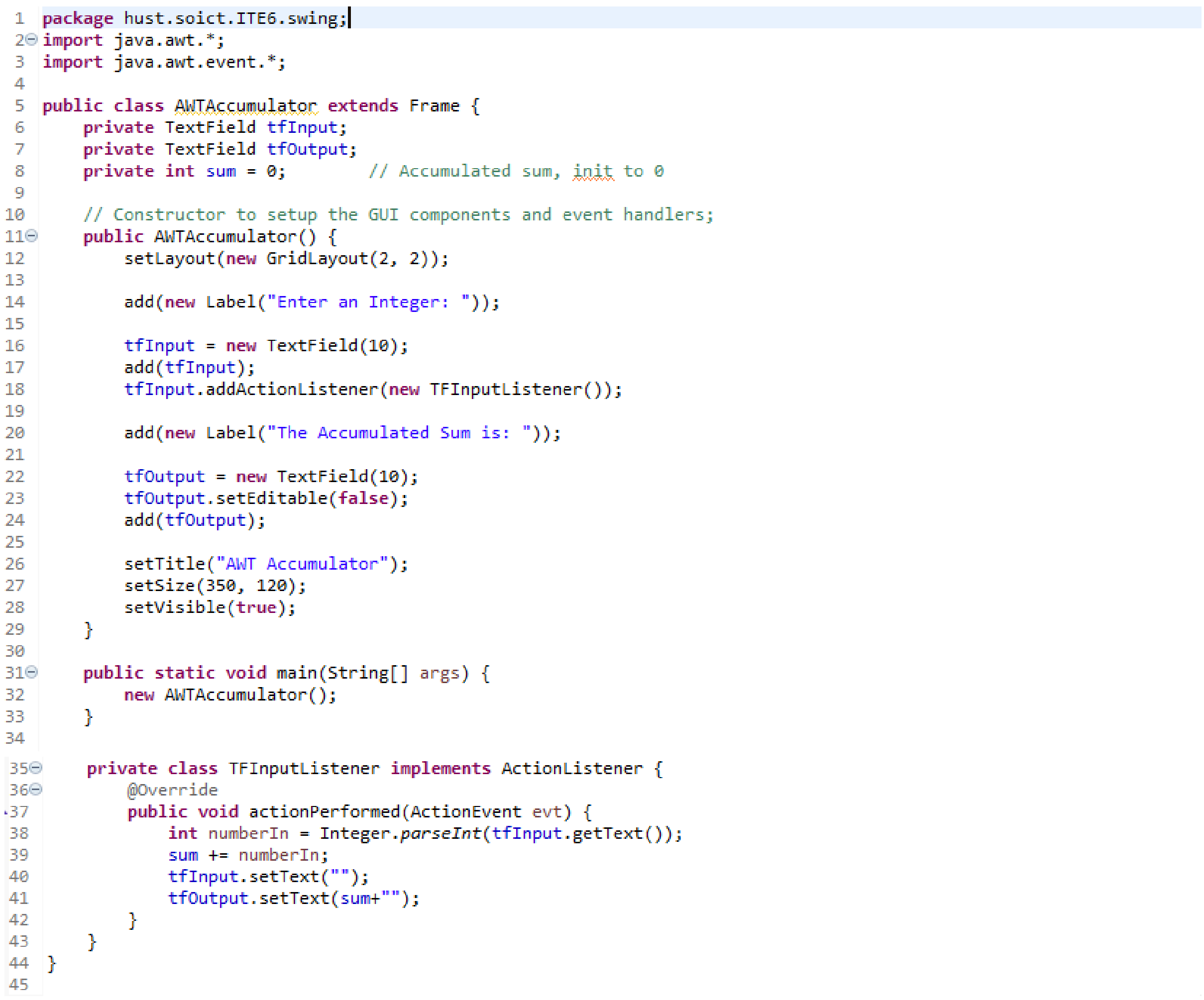
Sinh viên: Lê Hải Yến - MSSV: 20225780

I. New written code

1. Swing Component

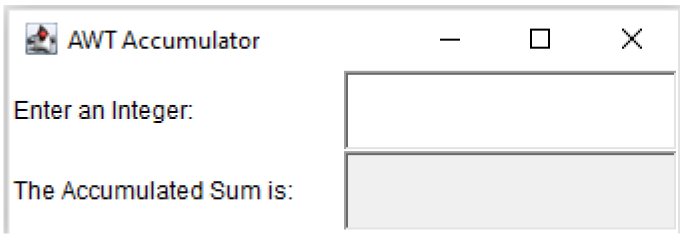
- **Branch: topic/aims-project/gui-project-swingcomponent**
- Tạo một dự án Java mới có tên là **GUIProject**, và đặt tất cả mã nguồn vào một package có tên “hust.soict.ITE6.swing”

1.1. AWTAccumulator Class



```java
package hust.soict.ITE6.swing;
import java.awt.*;
import java.awt.event.*;

public class AWTAccumulator extends Frame {
    private TextField tfInput;
    private TextField tfOutput;
    private int sum = 0;        // Accumulated sum, init to 0

    // Constructor to setup the GUI components and event handlers;
    public AWTAccumulator() {
        setLayout(new GridLayout(2, 2));

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

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

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

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

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

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

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

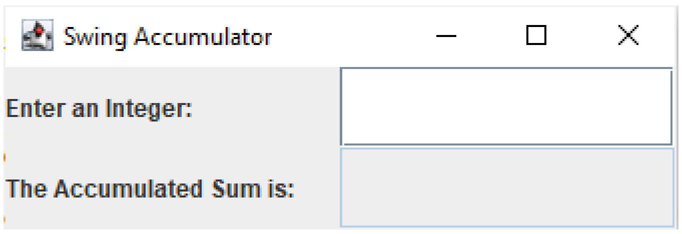
- Kết quả:



1.2. SwingAccumulator Class

```java
package hust.soict.ITE6.swing;

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

import javax.swing.*;

public class SwingAccumulator extends JFrame {

    private JTextField tfInput;
    private JTextField tfOutput;
    private int sum=0;          //Accumulated sum, init to 0

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

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

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

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

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

        setTitle("Swing Accumulator");
        setSize(350, 120);
        setVisible(true);
    }
    public static void main(String[] args) {
        new SwingAccumulator();
    }

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

- 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:

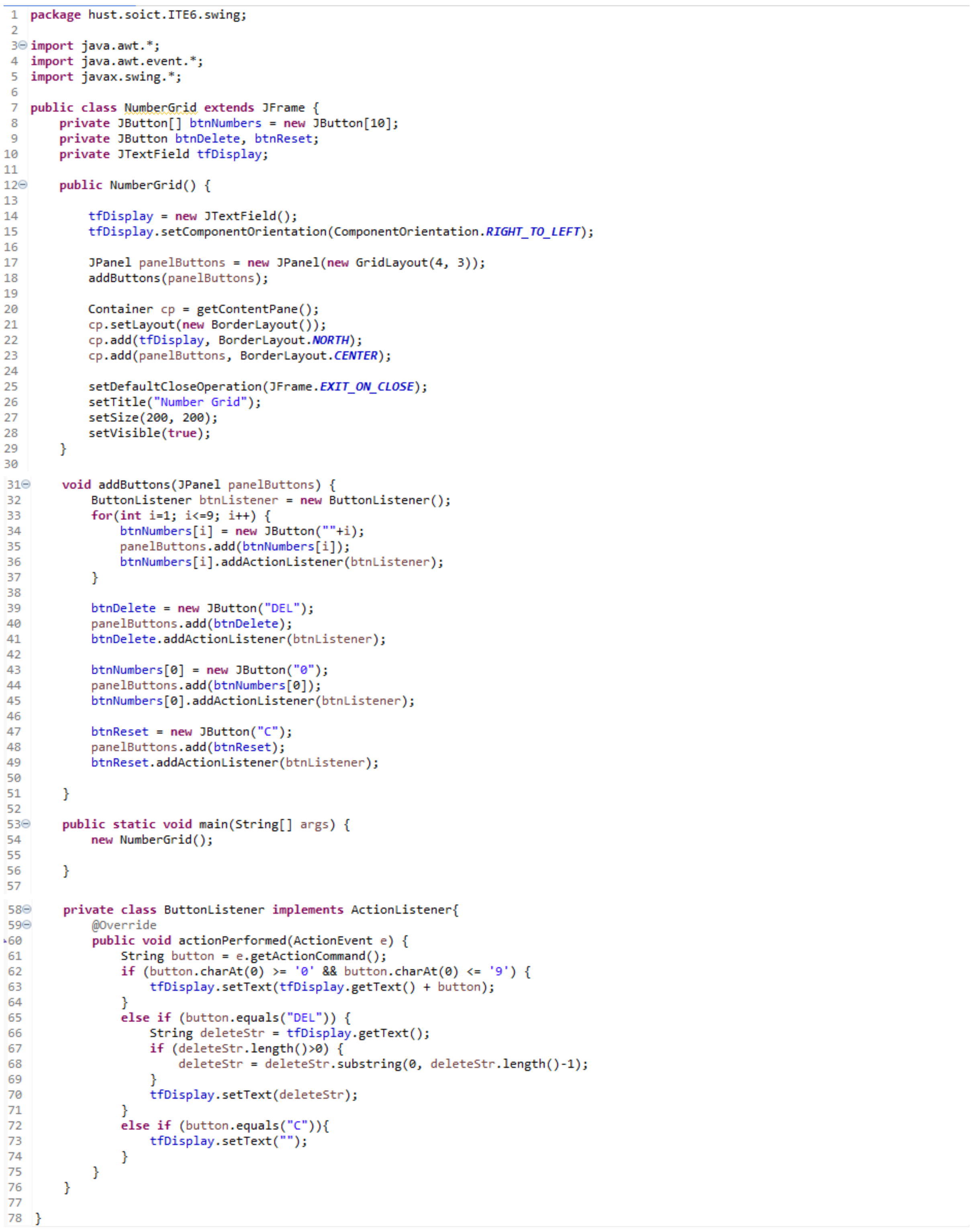
+ Container cấp cao nhất trong Swing và AWT: JFrame (Swing) vs Frame (AWT).

+ 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



```java
package hust.soict.ITE6.swing;

import java.awt.*;
import java.awt.event.*;
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("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);

    }

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

    }

    private class ButtonListener implements ActionListener{
        @Override
        public void actionPerformed(ActionEvent e) {
            String button = e.getActionCommand();
            if (button.charAt(0) >= '0' && button.charAt(0) <= '9') {
                tfDisplay.setText(tfDisplay.getText() + button);
            }
            else if (button.equals("DEL")) {
                String deleteStr = tfDisplay.getText();
                if (deleteStr.length()>0) {
                    deleteStr = deleteStr.substring(0, deleteStr.length()-1);
                }
                tfDisplay.setText(deleteStr);
            }
            else if (button.equals("C")){
                tfDisplay.setText("");
            }
        }
    }

}
```

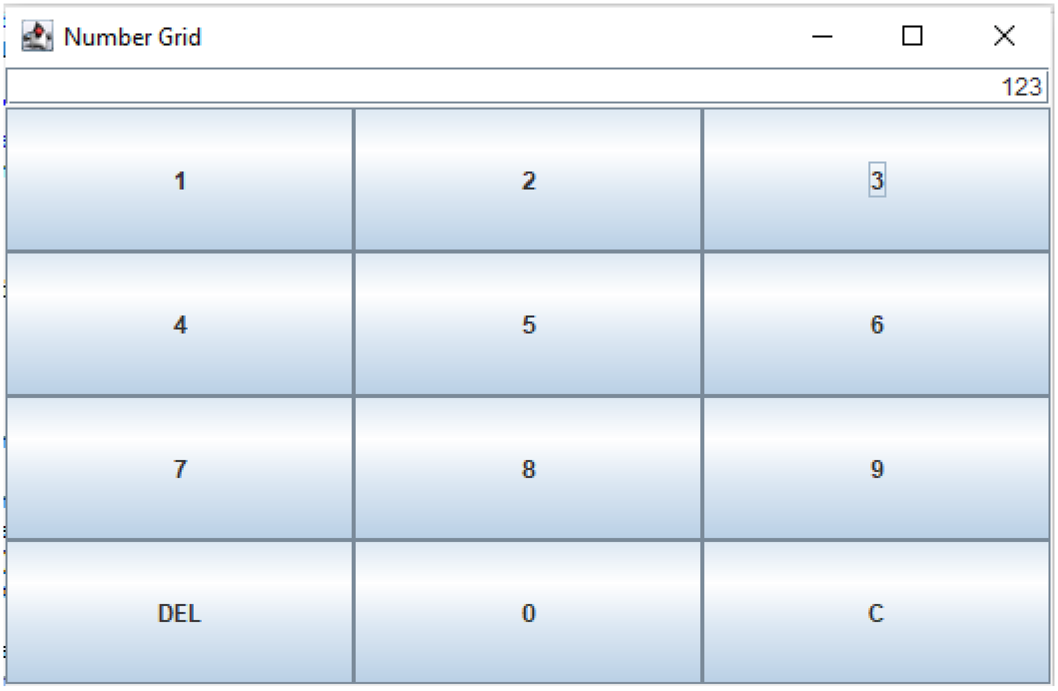
2.2. Giải thích và kết quả

- 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:

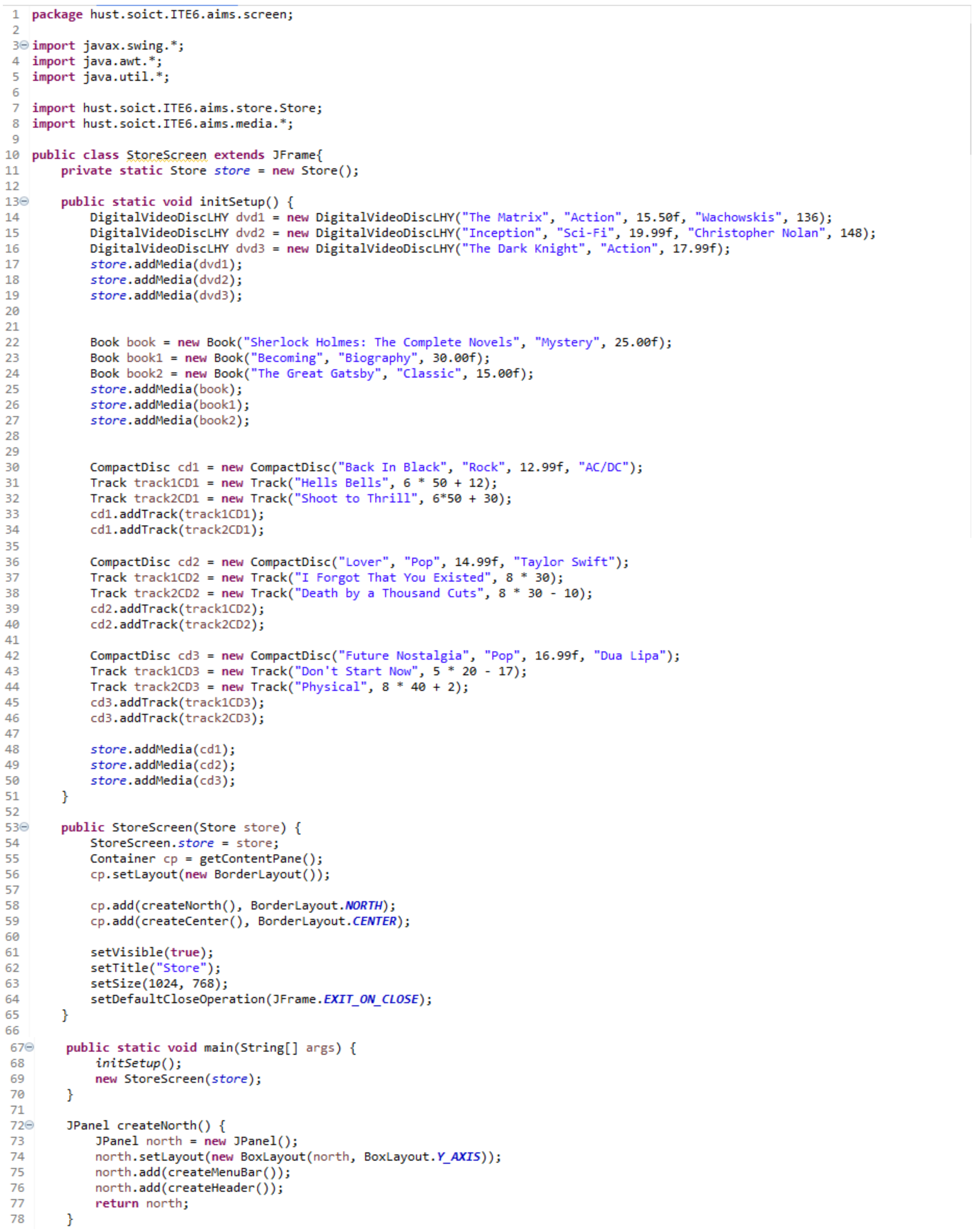


3. Create a graphical user interface for AIMS with Swing

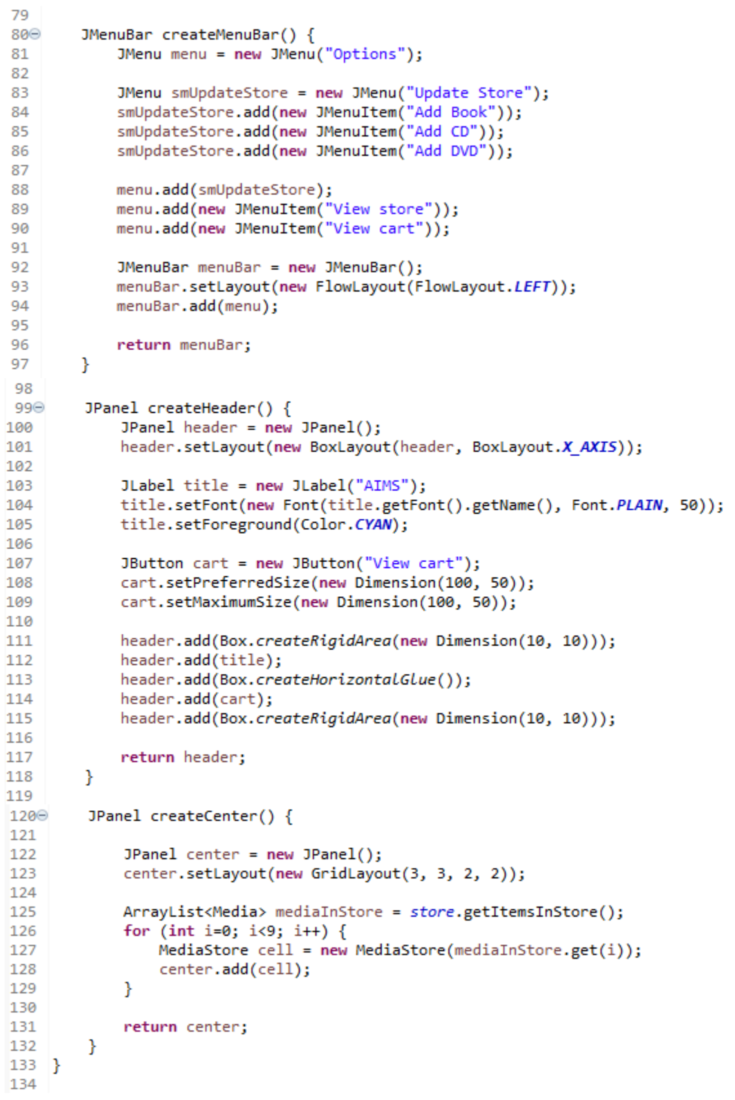
- Trong ứng dụng **AIMS**, chúng ta sẽ triển khai ba màn hình:
- **Màn hình "View Store"** (Xem cửa hàng) sử dụng **Swing**.
- **Màn hình "View Cart"** (Xem giỏ hàng) sử dụng **JavaFX**.
- **Màn hình "Update Store"** (Cập nhật cửa hàng) sử dụng **JavaFX**
- **Branch: topic/aims-project/view-store-screen**
- 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:

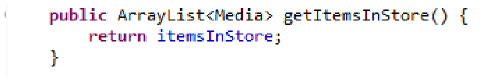


```java
package hust.soict.ITE6.aims.screen;

import javax.swing.*;
import java.awt.*;
import java.util.*;

import hust.soict.ITE6.aims.store.Store;
import hust.soict.ITE6.aims.media.*;

public class StoreScreen extends JFrame{
    private static Store store = new Store();

    public static void initSetup() {
        DigitalVideoDiscLHY dvd1 = new DigitalVideoDiscLHY("The Matrix", "Action", 15.50f, "Wachowskis", 136);
        DigitalVideoDiscLHY dvd2 = new DigitalVideoDiscLHY("Inception", "Sci-Fi", 19.99f, "Christopher Nolan", 148);
        DigitalVideoDiscLHY dvd3 = new DigitalVideoDiscLHY("The Dark Knight", "Action", 17.99f);
        store.addMedia(dvd1);
        store.addMedia(dvd2);
        store.addMedia(dvd3);


        Book book = new Book("Sherlock Holmes: The Complete Novels", "Mystery", 25.00f);
        Book book1 = new Book("Becoming", "Biography", 30.00f);
        Book book2 = new Book("The Great Gatsby", "Classic", 15.00f);
        store.addMedia(book);
        store.addMedia(book1);
        store.addMedia(book2);


        CompactDisc cd1 = new CompactDisc("Back In Black", "Rock", 12.99f, "AC/DC");
        Track track1CD1 = new Track("Hells Bells", 6 * 50 + 12);
        Track track2CD1 = new Track("Shoot to Thrill", 6*50 + 30);
        cd1.addTrack(track1CD1);
        cd1.addTrack(track2CD1);

        CompactDisc cd2 = new CompactDisc("Lover", "Pop", 14.99f, "Taylor Swift");
        Track track1CD2 = new Track("I Forgot That You Existed", 8 * 30);
        Track track2CD2 = new Track("Death by a Thousand Cuts", 8 * 30 - 10);
        cd2.addTrack(track1CD2);
        cd2.addTrack(track2CD2);

        CompactDisc cd3 = new CompactDisc("Future Nostalgia", "Pop", 16.99f, "Dua Lipa");
        Track track1CD3 = new Track("Don't Start Now", 5 * 20 - 17);
        Track track2CD3 = new Track("Physical", 8 * 40 + 2);
        cd3.addTrack(track1CD3);
        cd3.addTrack(track2CD3);

        store.addMedia(cd1);
        store.addMedia(cd2);
        store.addMedia(cd3);
    }

    public StoreScreen(Store store) {
        StoreScreen.store = store;
        Container cp = getContentPane();
        cp.setLayout(new BorderLayout());

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

        setVisible(true);
        setTitle("Store");
        setSize(1024, 768);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

     public static void main(String[] args) {
         initSetup();
         new StoreScreen(store);
     }

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



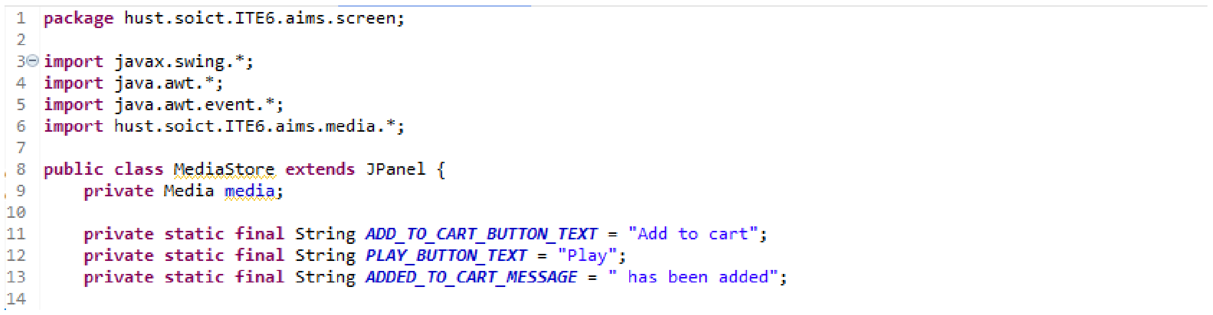
```java

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

        JMenu smUpdateStore = new JMenu("Update Store");
        smUpdateStore.add(new JMenuItem("Add Book"));
        smUpdateStore.add(new JMenuItem("Add CD"));
        smUpdateStore.add(new JMenuItem("Add DVD"));

        menu.add(smUpdateStore);
        menu.add(new JMenuItem("View store"));
        menu.add(new JMenuItem("View cart"));

        JMenuBar menuBar = new JMenuBar();
        menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
        menuBar.add(menu);

        return menuBar;
    }

    JPanel createHeader() {
        JPanel header = new JPanel();
        header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));

        JLabel title = new JLabel("AIMS");
        title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
        title.setForeground(Color.CYAN);

        JButton cart = new JButton("View cart");
        cart.setPreferredSize(new Dimension(100, 50));
        cart.setMaximumSize(new Dimension(100, 50));

        header.add(Box.createRigidArea(new Dimension(10, 10)));
        header.add(title);
        header.add(Box.createHorizontalGlue());
        header.add(cart);
        header.add(Box.createRigidArea(new Dimension(10, 10)));

        return header;
    }

    JPanel createCenter() {

        JPanel center = new JPanel();
        center.setLayout(new GridLayout(3, 3, 2, 2));

        ArrayList<Media> mediaInStore = store.getItemsInStore();
        for (int i=0; i<9; i++) {
            MediaStore cell = new MediaStore(mediaInStore.get(i));
            center.add(cell);
        }

        return center;
    }
}

```

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

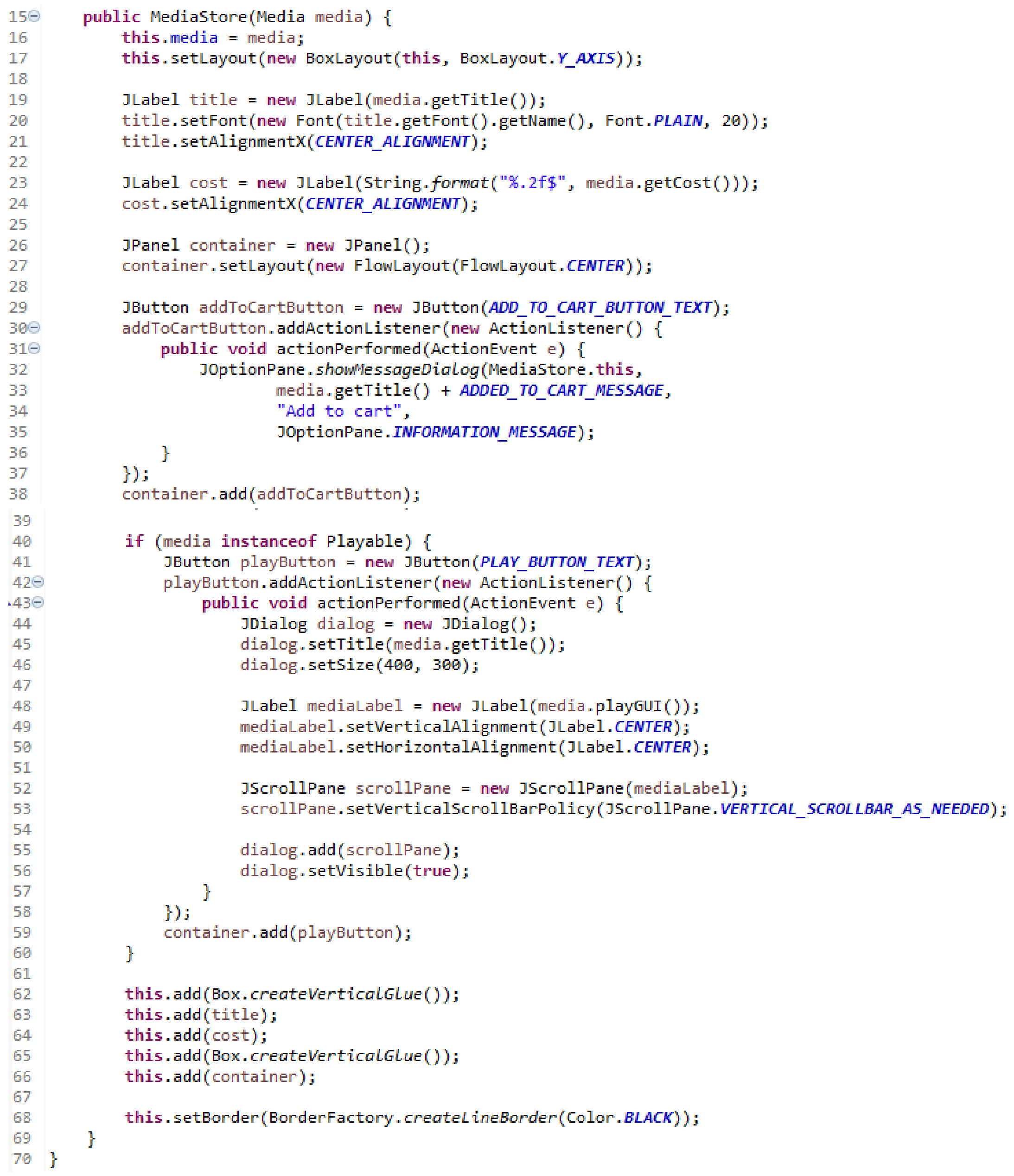


```java
public ArrayList<Media> getItemsInStore() {
    return itemsInStore;
}
```

- Tạo MediaStore Class:

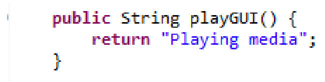


```java
package hust.soict.ITE6.aims.screen;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import hust.soict.ITE6.aims.media.*;

public class MediaStore extends JPanel {
    private Media media;

    private static final String ADD_TO_CART_BUTTON_TEXT = "Add to cart";
    private static final String PLAY_BUTTON_TEXT = "Play";
    private static final String ADDED_TO_CART_MESSAGE = " has been added";

```



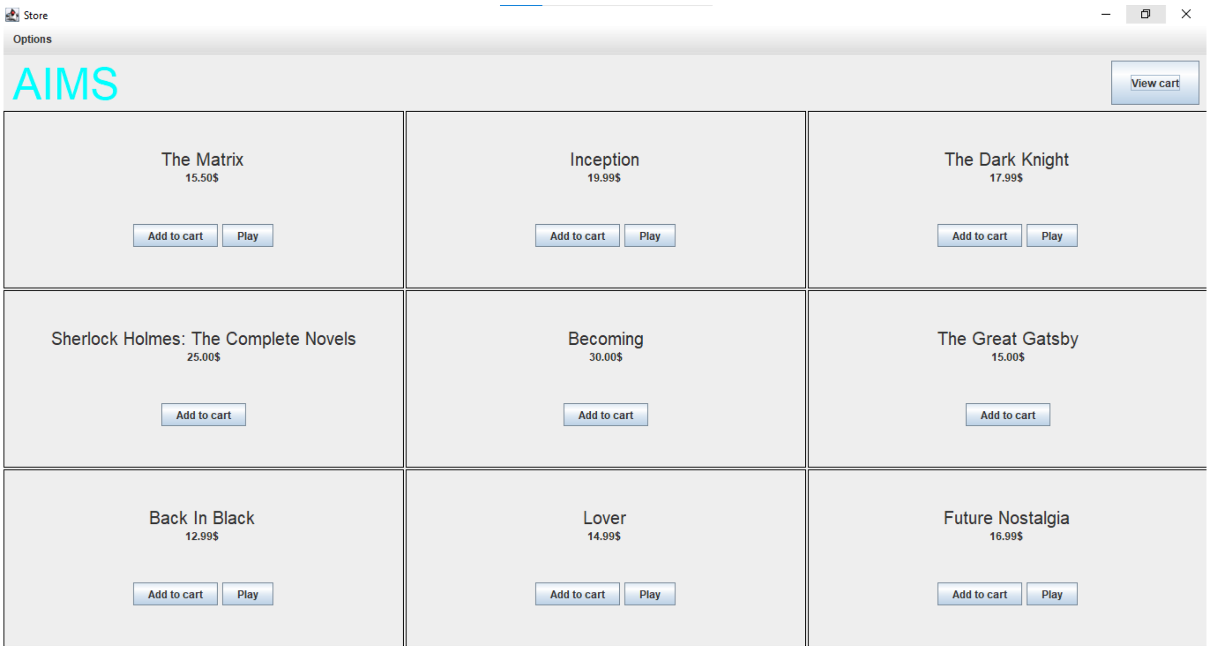
```java
    public MediaStore(Media media) {
        this.media = media;
        this.setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));

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

        JLabel cost = new JLabel(String.format("%.2f$", media.getCost()));
        cost.setAlignmentX(CENTER_ALIGNMENT);

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

        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);

        if (media instanceof Playable) {
            JButton playButton = new JButton(PLAY_BUTTON_TEXT);
            playButton.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                    JDialog dialog = new JDialog();
                    dialog.setTitle(media.getTitle());
                    dialog.setSize(400, 300);

                    JLabel mediaLabel = new JLabel(media.playGUI());
                    mediaLabel.setVerticalAlignment(JLabel.CENTER);
                    mediaLabel.setHorizontalAlignment(JLabel.CENTER);

                    JScrollPane scrollPane = new JScrollPane(mediaLabel);
                    scrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);

                    dialog.add(scrollPane);
                    dialog.setVisible(true);
                }
            });
            container.add(playButton);
        }

        this.add(Box.createVerticalGlue());
        this.add(title);
        this.add(cost);
        this.add(Box.createVerticalGlue());
        this.add(container);

        this.setBorder(BorderFactory.createLineBorder(Color.BLACK));
    }
}
```

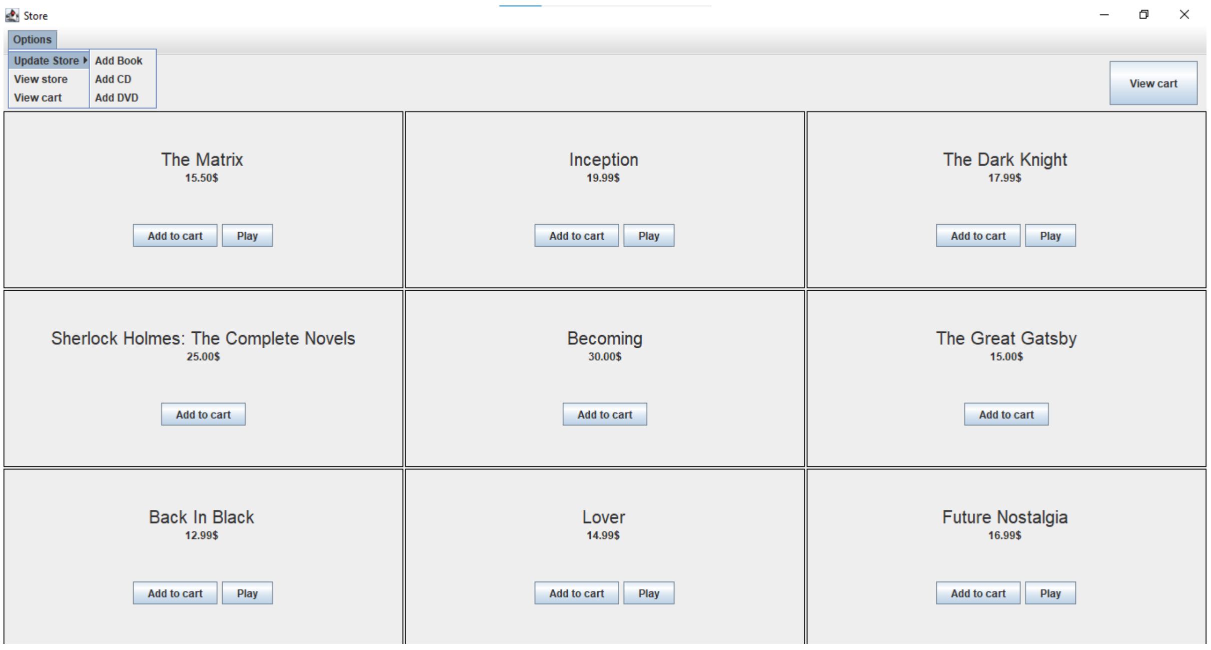
- Thêm method **playGUI()** cho lớp **Media**

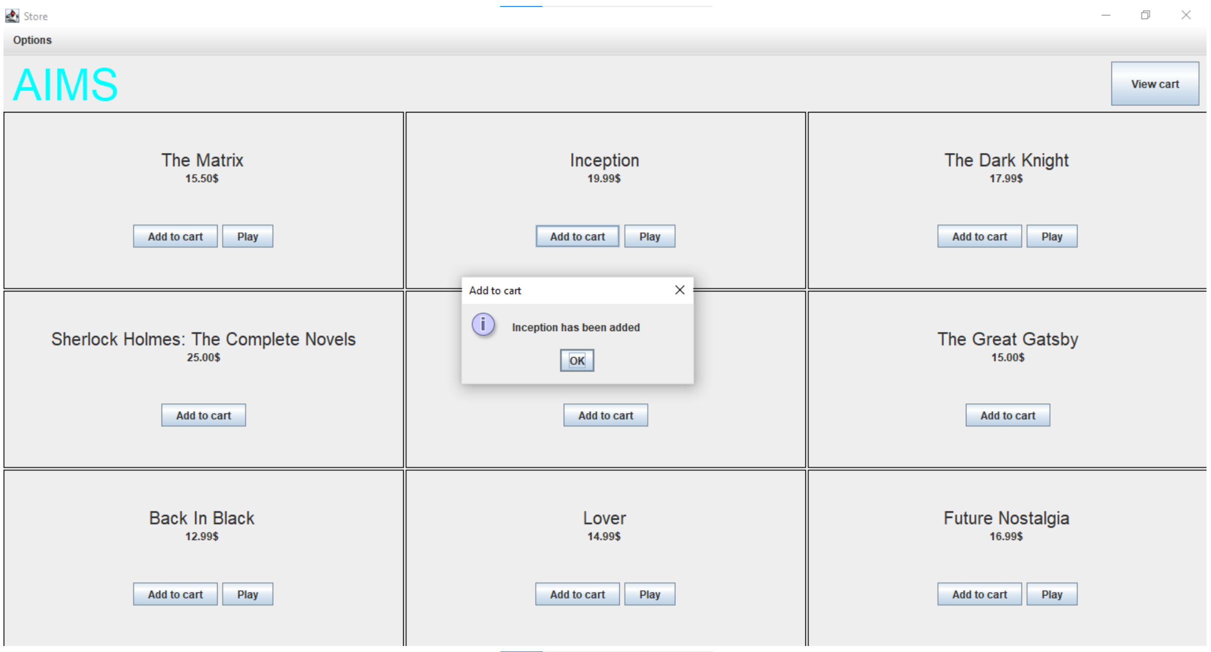


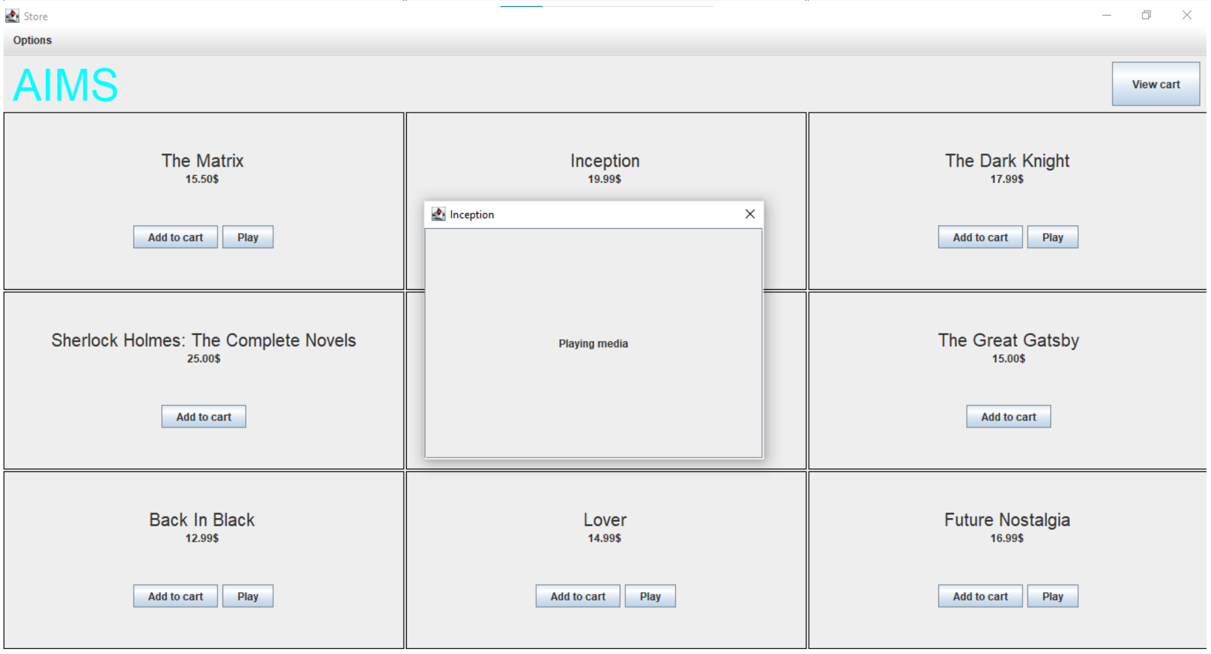
```java
public String playGUI() {
    return "Playing media";
}
```

3.2. Adding more user interaction







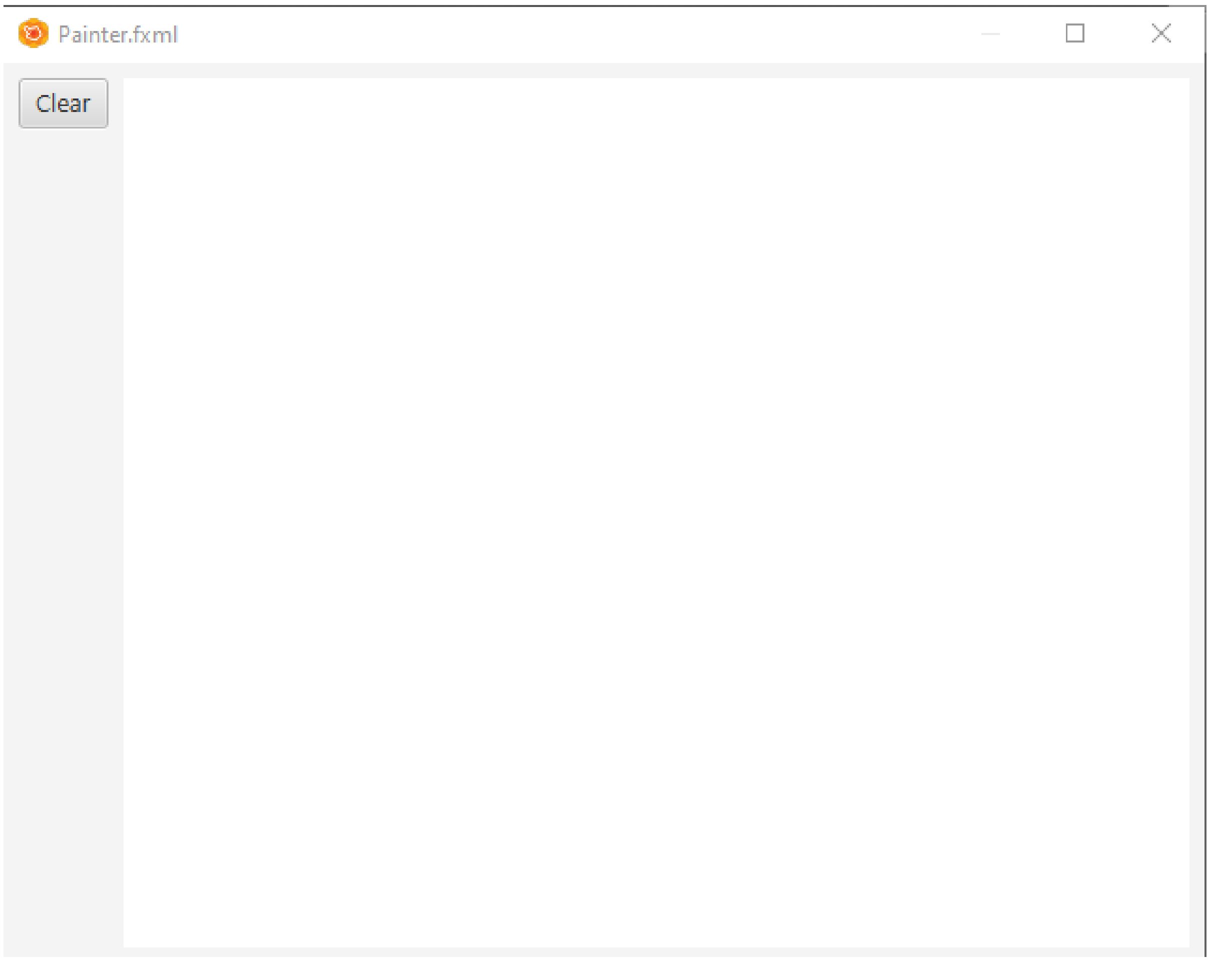


4. JavaFX API

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

4.1. Create the FXML file

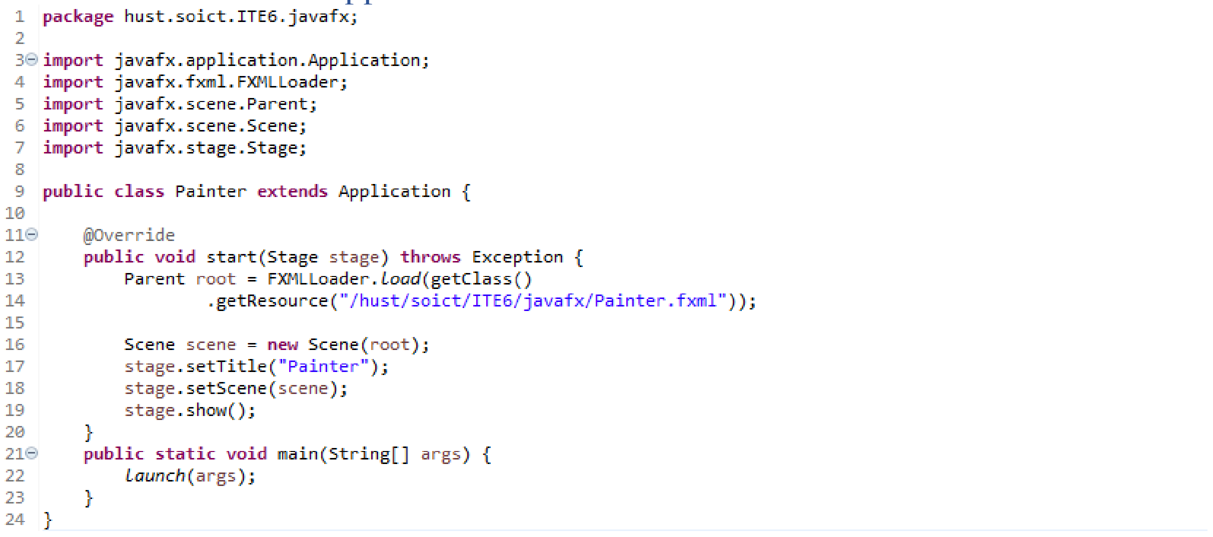
- Tạo file **Painter.fxml** trong **hust.soict.ITE6.javafx** và mở bằng **SceneBuilder**



4.2. Create the controller class

```java
package hust.soict.ITE6.javafx;

import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;

public class PainterController {

    @FXML
    private Pane drawingAreaPane;

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

    @FXML
    void drawingAreaMouseDragged(MouseEvent event) {
        Circle newCircle = new Circle(event.getX(),
                event.getY(), 4, Color.BLACK);
        drawingAreaPane.getChildren().add(newCircle);
    }

}
```

4.3. Create the application



```java
package hust.soict.ITE6.javafx;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class Painter extends Application {

    @Override
    public void start(Stage stage) throws Exception {
        Parent root = FXMLLoader.load(getClass()
                .getResource("/hust/soict/ITE6/javafx/Painter.fxml"));

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

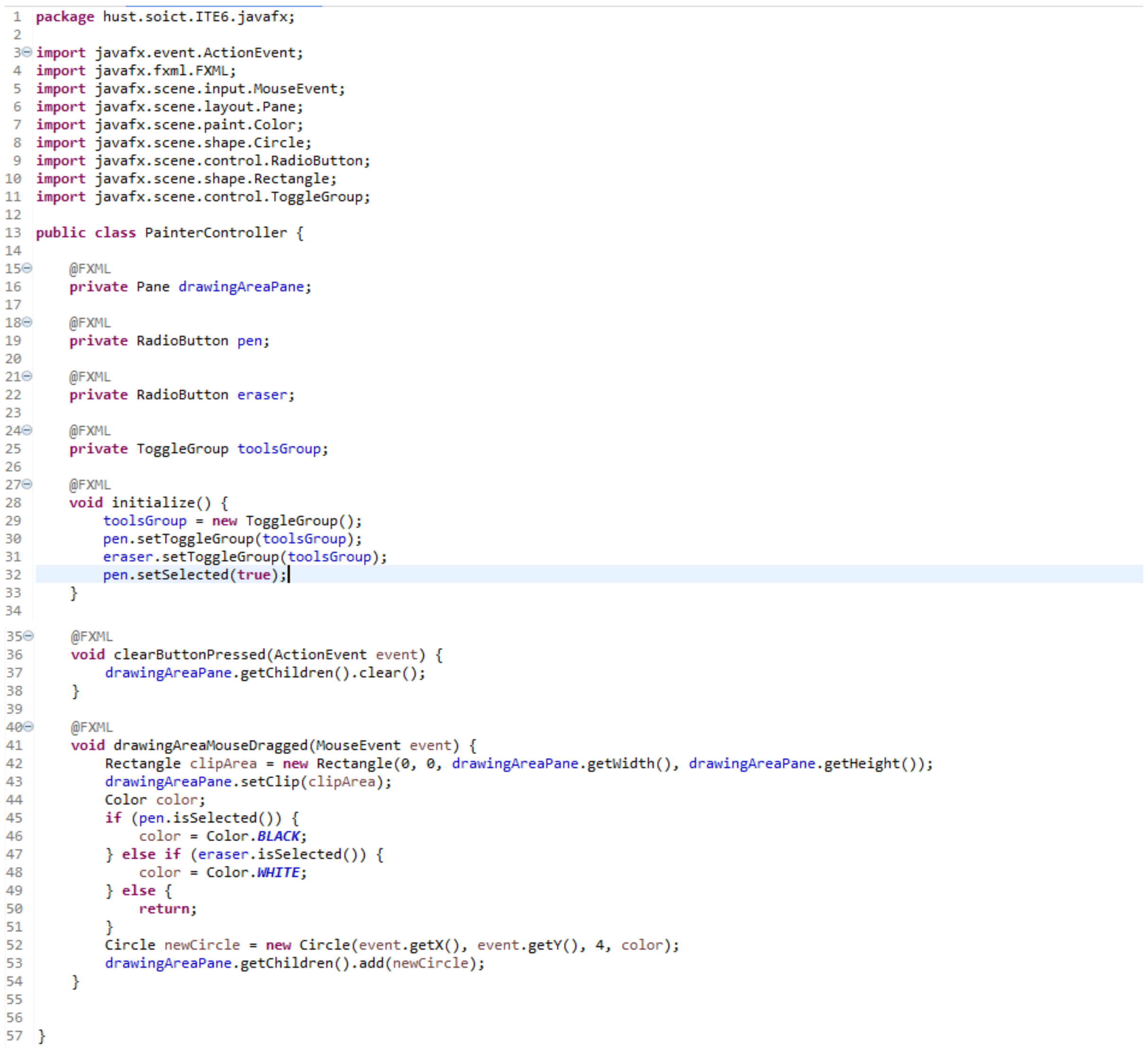
4.4. Practice exercise

- Sử dụng **TitledPane**

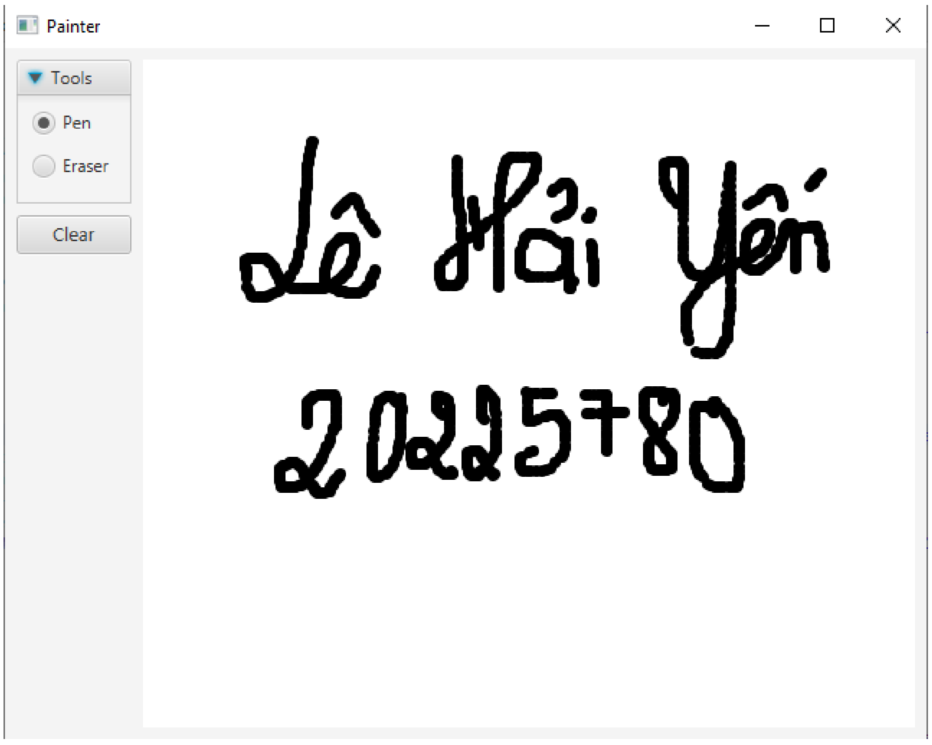
- **RadioButtons** tương ứng với **pen** và **eraser**

- 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:

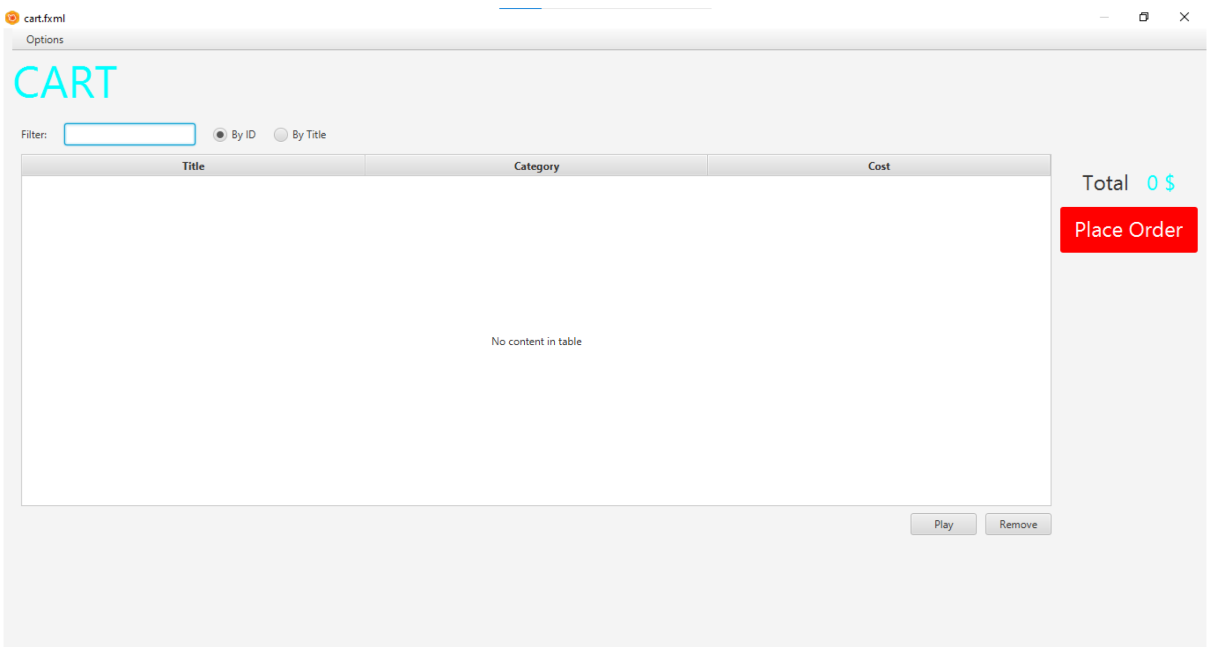


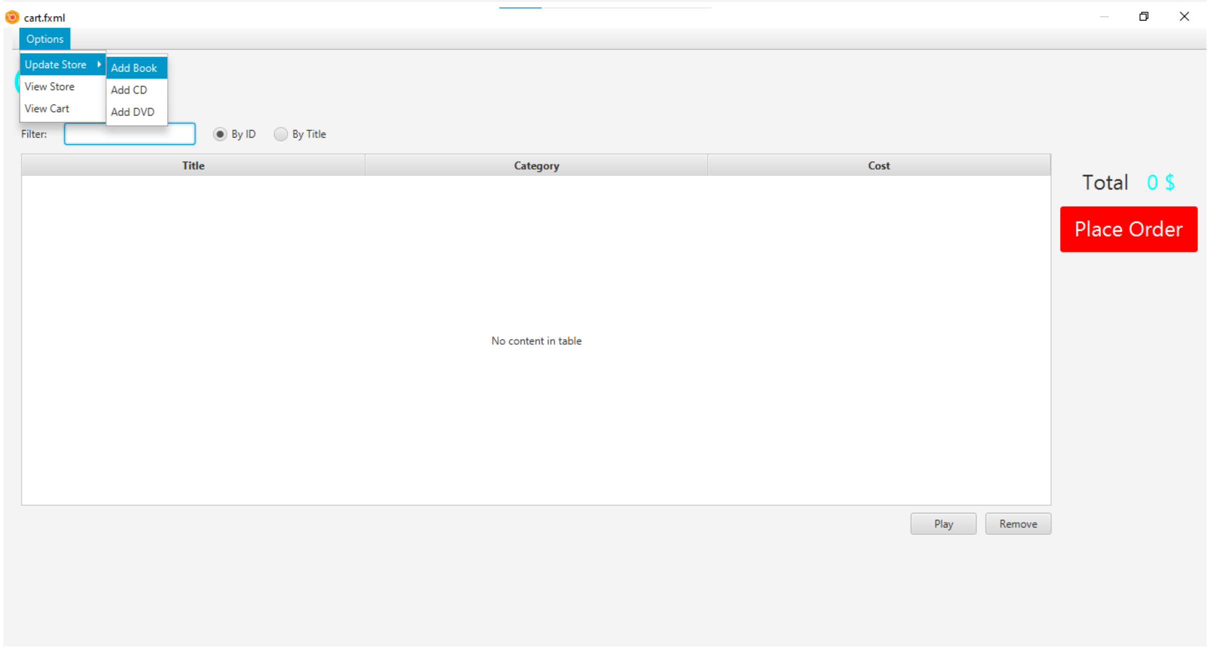
```java
package hust.soict.ITE6.javafx;

import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;
import javafx.scene.control.RadioButton;
import javafx.scene.shape.Rectangle;
import javafx.scene.control.ToggleGroup;

public class PainterController {

    @FXML
    private Pane drawingAreaPane;

    @FXML
    private RadioButton pen;

    @FXML
    private RadioButton eraser;

    @FXML
    private ToggleGroup toolsGroup;

    @FXML
    void initialize() {
        toolsGroup = new ToggleGroup();
        pen.setToggleGroup(toolsGroup);
        eraser.setToggleGroup(toolsGroup);
        pen.setSelected(true);
    }

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

    @FXML
    void drawingAreaMouseDragged(MouseEvent event) {
        Rectangle clipArea = new Rectangle(0, 0, drawingAreaPane.getWidth(), drawingAreaPane.getHeight());
        drawingAreaPane.setClip(clipArea);
        Color color;
        if (pen.isSelected()) {
            color = Color.BLACK;
        } else if (eraser.isSelected()) {
            color = Color.WHITE;
        } else {
            return;
        }
        Circle newCircle = new Circle(event.getX(), event.getY(), 4, color);
        drawingAreaPane.getChildren().add(newCircle);
    }


}
```



5. Setting up the View Cart Screen with ScreenBuilder

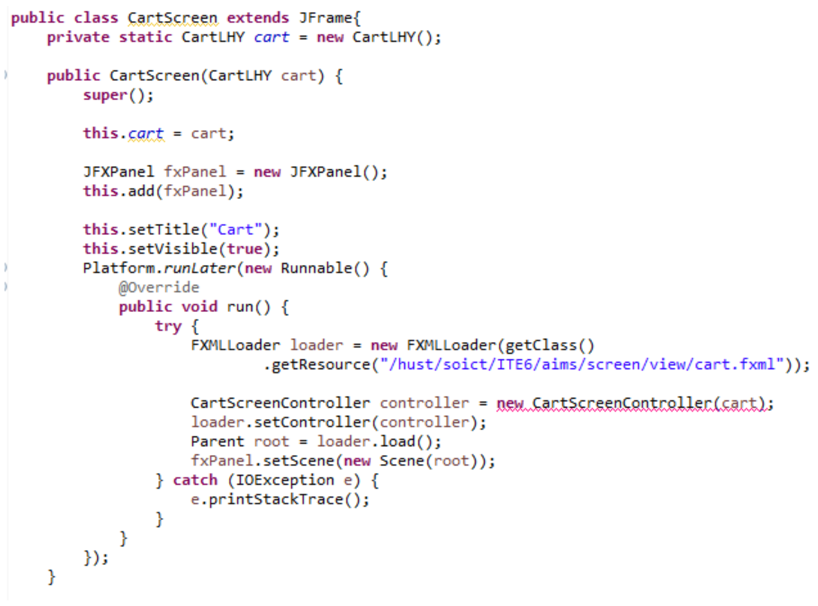
- **Branch: topic/aims-project/view-cart-screen**
- Các bài tiếp theo làm trong package **hust.soict.ITE6.aims.screen**
- Tạo **cart.fxml** trong **hust.soict.ITE6.aims.screen.view**





6. Integrating JavaFX into Swing application - The JFXPanel class

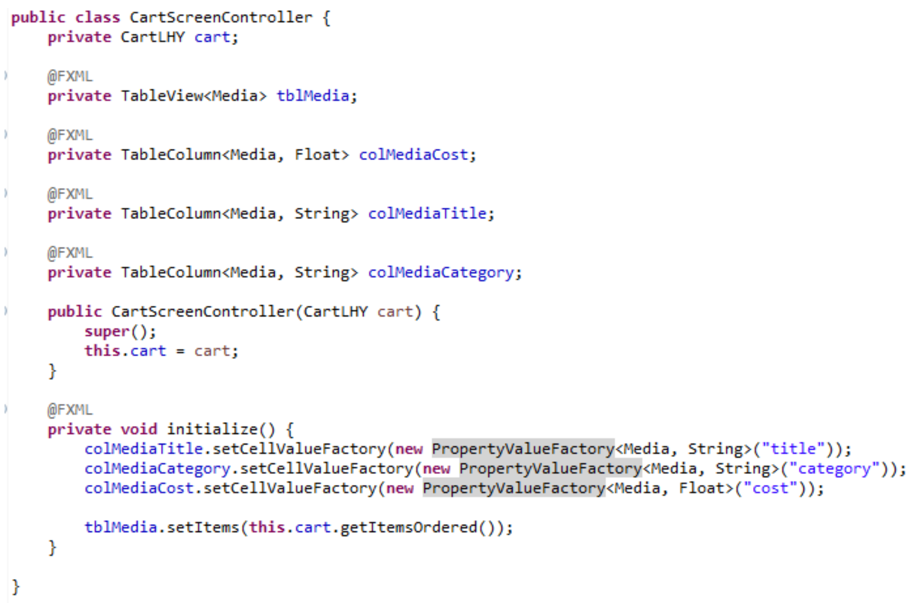
- CartScreen Class



```java
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();
                }
            }
        });
    }
```

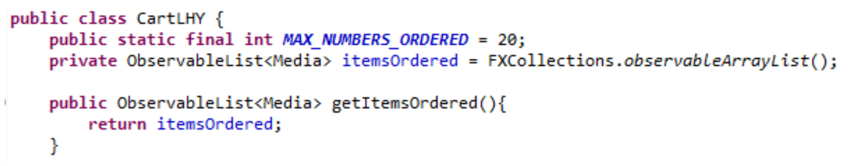
7. View the item in cart – JavaFX's data-driven UI

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



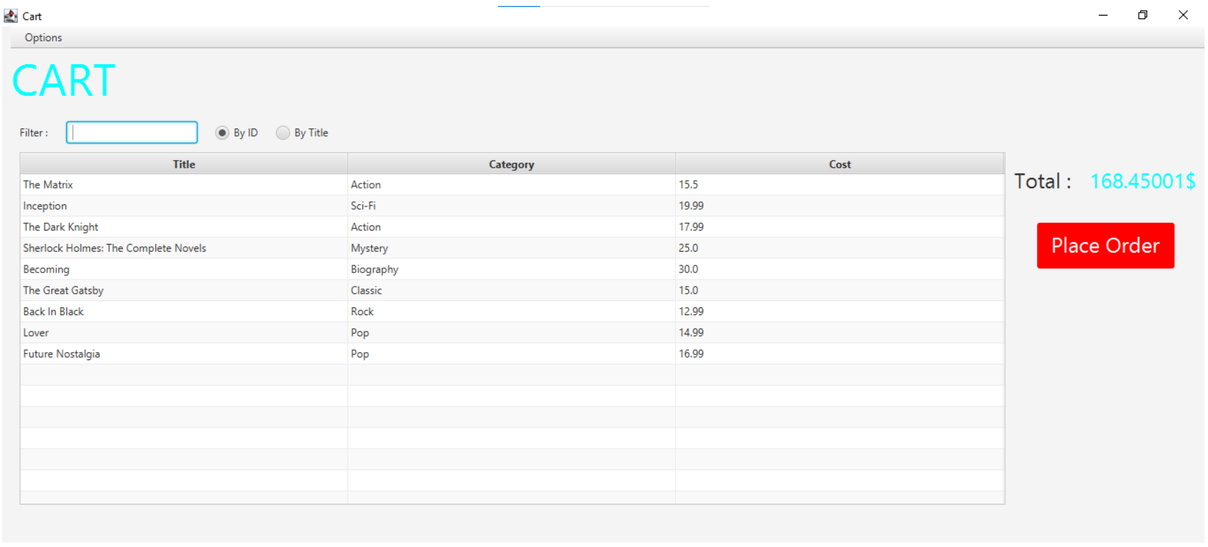
```java
public class CartScreenController {
    private CartLHY cart;

    @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**



```java
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:



Title	Category	Cost
The Matrix	Action	15.5
Inception	Sci-Fi	19.99
The Dark Knight	Action	17.99
Sherlock Holmes: The Complete Novels	Mystery	25.0
Becoming	Biography	30.0
The Great Gatsby	Classic	15.0
Back In Black	Rock	12.99
Lover	Pop	14.99
Future Nostalgia	Pop	16.99

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**

```java
@FXML
private Button btnPlay;

@FXML
private Button btnRemove;
```

```java
@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

```java
@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**

```java
@FXML
private TextField tfFilter;

@FXML
private RadioButton radioBtnFilterId;

@FXML
private RadioButton radioBtnFilterTitle;
```

```java
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**

```java
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.";
    }
}
```

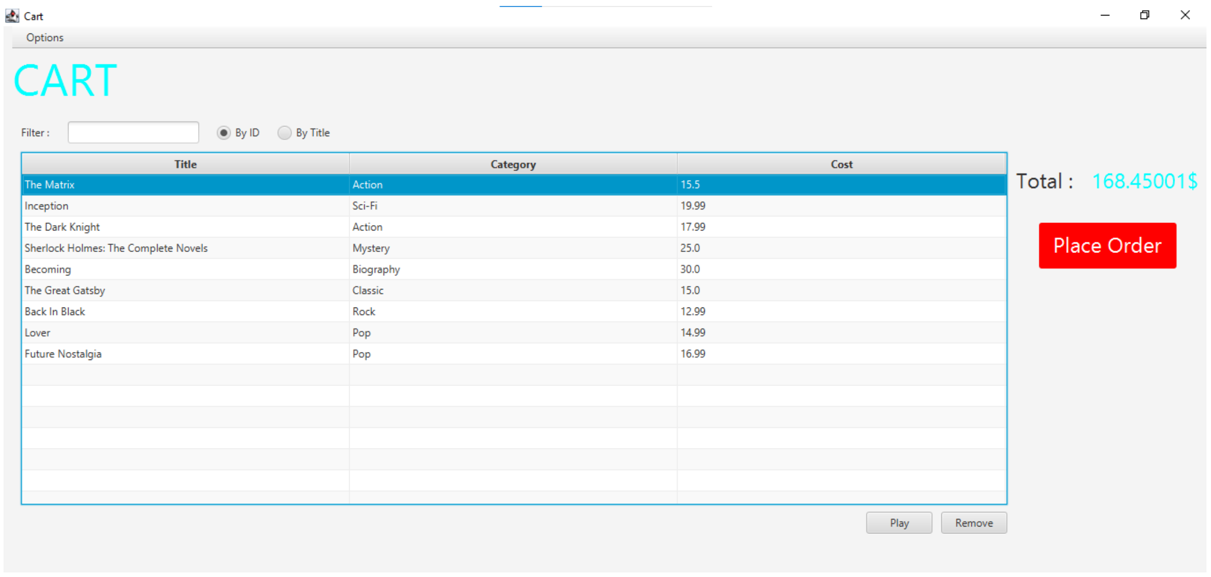
```
@FXML
private Button placeOrder;


@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() + "$");
```



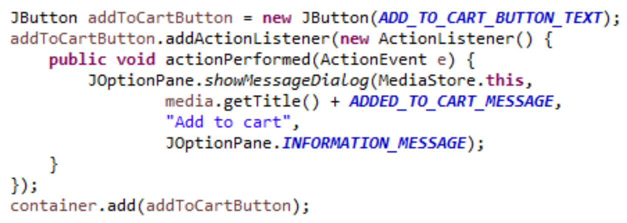
Title	Category	Cost
The Matrix	Action	15.5
Inception	Sci-Fi	19.99
The Dark Knight	Action	17.99
Sherlock Holmes: The Complete Novels	Mystery	25.0
Becoming	Biography	30.0
The Great Gatsby	Classic	15.0
Back In Black	Rock	12.99
Lover	Pop	14.99
Future Nostalgia	Pop	16.99

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**

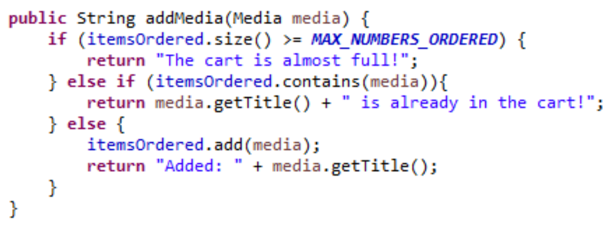
```
public class StoreScreen extends JFrame{
    private static Store store = new Store();
    private static CartLHY cart = new CartLHY();
```

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



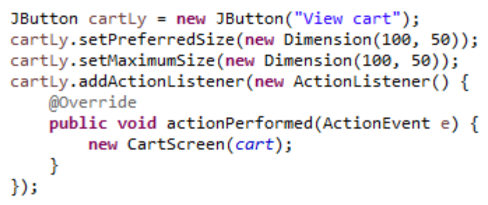
```java
JButton addToCartButton = new JButton(ADD_TO_CART_BUTTON_TEXT);
addToCartButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        JOptionPane.showMessageDialog(MediaStore.this,
                media.getTitle() + ADDED_TO_CART_MESSAGE,
         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

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



```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();
    }
}
```

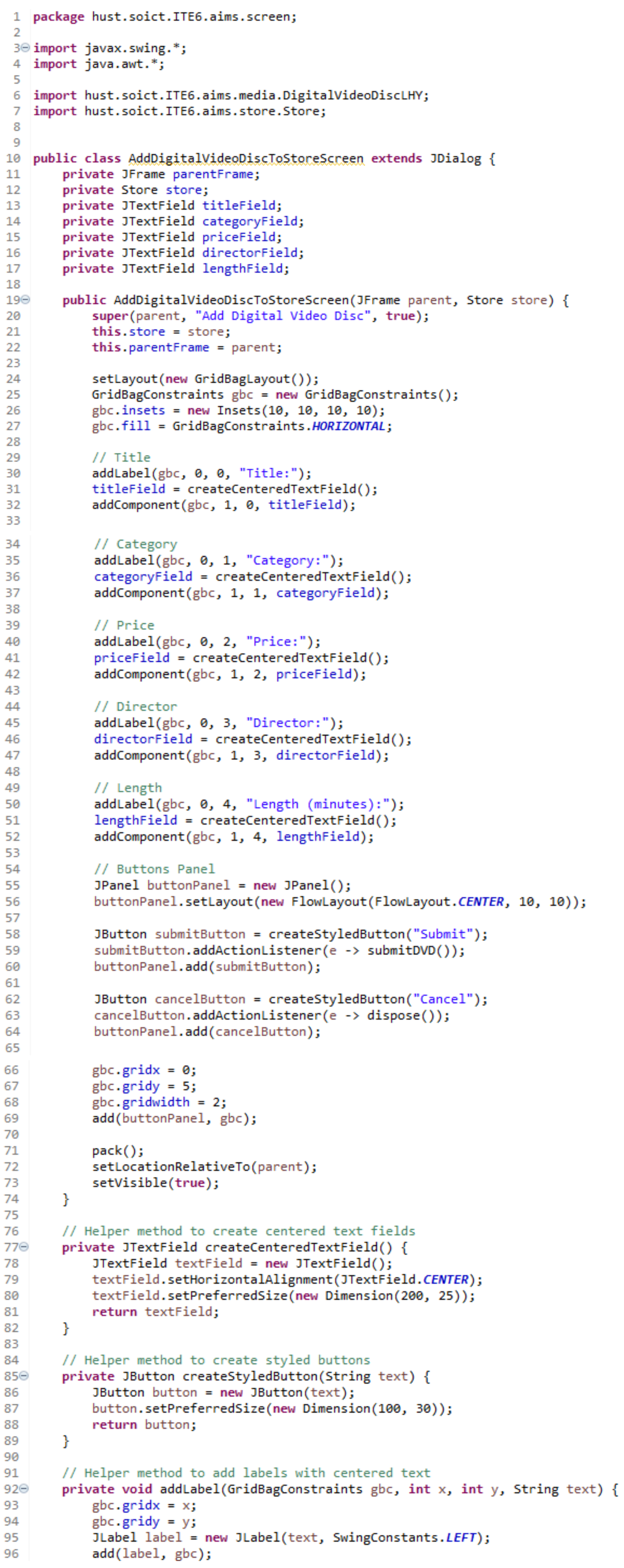
- Action **View Cart** hiện ra **CartScreen** với những cart đã được thêm



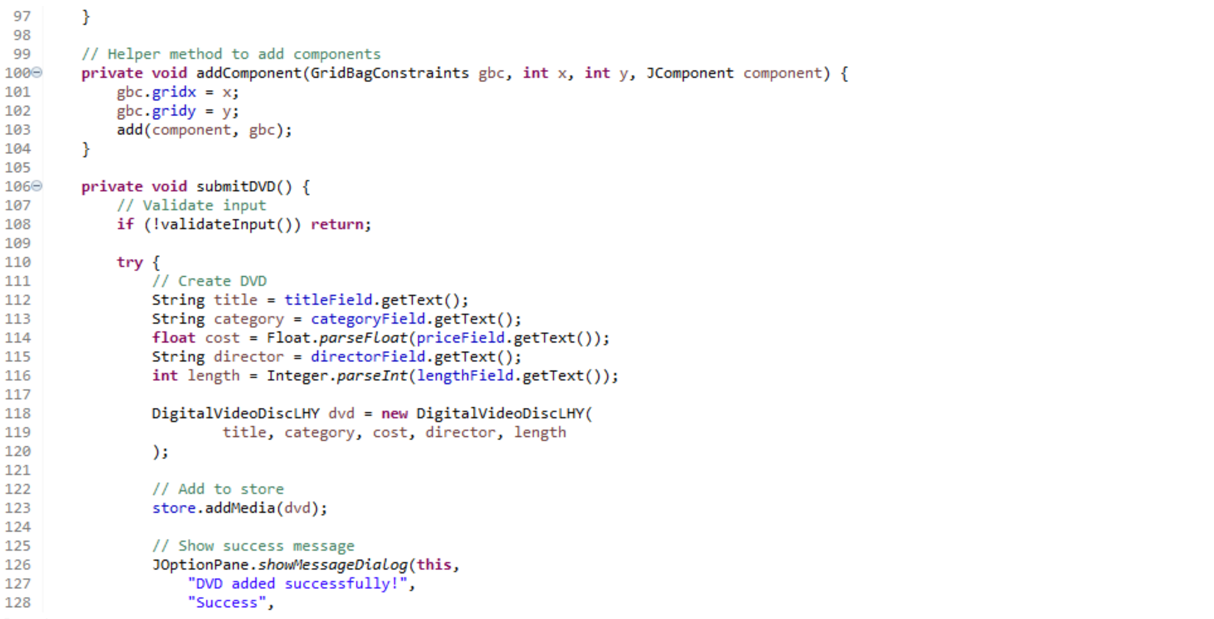
```java
JButton cartLy = new JButton("View cart");
cartLy.setPreferredSize(new Dimension(100, 50));
cartLy.setMaximumSize(new Dimension(100, 50));
cartLy.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        new CartScreen(cart);
    }
});
```

- Tạo 3 lớp sau:

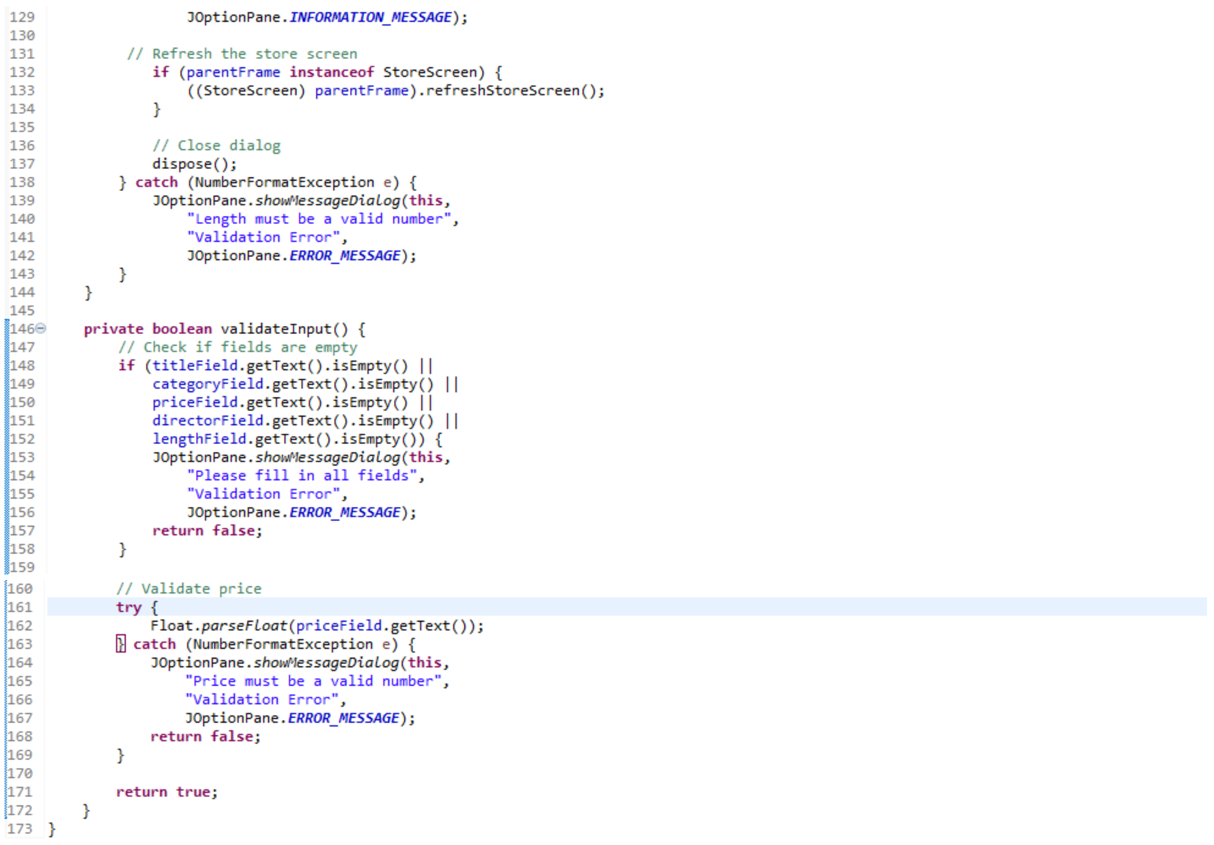
- AddDigitalVideoDiscToStoreScreen



```java
package hust.soict.ITE6.aims.screen;

import javax.swing.*;
import java.awt.*;

import hust.soict.ITE6.aims.media.DigitalVideoDiscLHY;
import hust.soict.ITE6.aims.store.Store;


public class AddDigitalVideoDiscToStoreScreen extends JDialog {
    private JFrame parentFrame;
    private Store store;
    private JTextField titleField;
    private JTextField categoryField;
    private JTextField priceField;
    private JTextField directorField;
    private JTextField lengthField;

    public AddDigitalVideoDiscToStoreScreen(JFrame parent, Store store) {
        super(parent, "Add Digital Video Disc", true);
        this.store = store;
        this.parentFrame = parent;

        setLayout(new GridBagLayout());
        GridBagConstraints gbc = new GridBagConstraints();
        gbc.insets = new Insets(10, 10, 10, 10);
        gbc.fill = GridBagConstraints.HORIZONTAL;

        // Title
        addLabel(gbc, 0, 0, "Title:");
        titleField = createCenteredTextField();
        addComponent(gbc, 1, 0, titleField);

        // Category
        addLabel(gbc, 0, 1, "Category:");
        categoryField = createCenteredTextField();
        addComponent(gbc, 1, 1, categoryField);

        // Price
        addLabel(gbc, 0, 2, "Price:");
        priceField = createCenteredTextField();
        addComponent(gbc, 1, 2, priceField);

        // Director
        addLabel(gbc, 0, 3, "Director:");
        directorField = createCenteredTextField();
        addComponent(gbc, 1, 3, directorField);

        // Length
        addLabel(gbc, 0, 4, "Length (minutes):");
        lengthField = createCenteredTextField();
        addComponent(gbc, 1, 4, lengthField);

        // Buttons Panel
        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new FlowLayout(FlowLayout.CENTER, 10, 10));

        JButton submitButton = createStyledButton("Submit");
        submitButton.addActionListener(e -> submitDVD());
        buttonPanel.add(submitButton);

        JButton cancelButton = createStyledButton("Cancel");
        cancelButton.addActionListener(e -> dispose());
        buttonPanel.add(cancelButton);

        gbc.gridx = 0;
        gbc.gridy = 5;
        gbc.gridwidth = 2;
        add(buttonPanel, gbc);

        pack();
        setLocationRelativeTo(parent);
        setVisible(true);
    }

    // Helper method to create centered text fields
    private JTextField createCenteredTextField() {
        JTextField textField = new JTextField();
        textField.setHorizontalAlignment(JTextField.CENTER);
        textField.setPreferredSize(new Dimension(200, 25));
        return textField;
    }

    // Helper method to create styled buttons
    private JButton createStyledButton(String text) {
        JButton button = new JButton(text);
        button.setPreferredSize(new Dimension(100, 30));
        return button;
    }

    // Helper method to add labels with centered text
    private void addLabel(GridBagConstraints gbc, int x, int y, String text) {
        gbc.gridx = x;
        gbc.gridy = y;
        JLabel label = new JLabel(text, SwingConstants.LEFT);
        add(label, gbc);
```

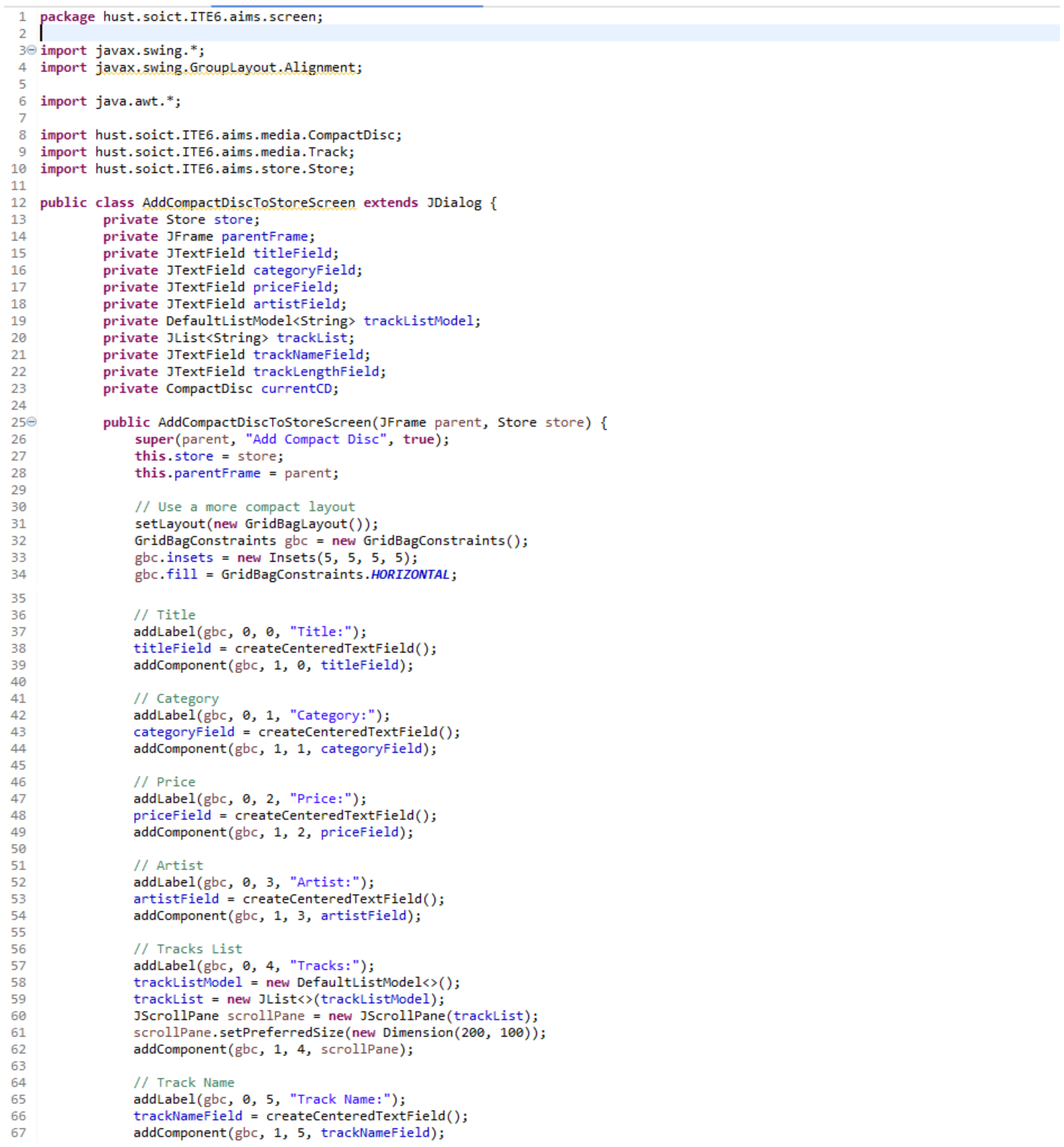


```java
    }

    // Helper method to add components
    private void addComponent(GridBagConstraints gbc, int x, int y, JComponent component) {
        gbc.gridx = x;
        gbc.gridy = y;
        add(component, gbc);
    }

    private void submitDVD() {
        // Validate input
        if (!validateInput()) return;

        try {
            // Create DVD
            String title = titleField.getText();
            String category = categoryField.getText();
            float cost = Float.parseFloat(priceField.getText());
            String director = directorField.getText();
            int length = Integer.parseInt(lengthField.getText());

            DigitalVideoDiscLHY dvd = new DigitalVideoDiscLHY(
                    title, category, cost, director, length
            );

            // Add to store
            store.addMedia(dvd);

            // Show success message
            JOptionPane.showMessageDialog(this,
                "DVD added successfully!",
                "Success",
```

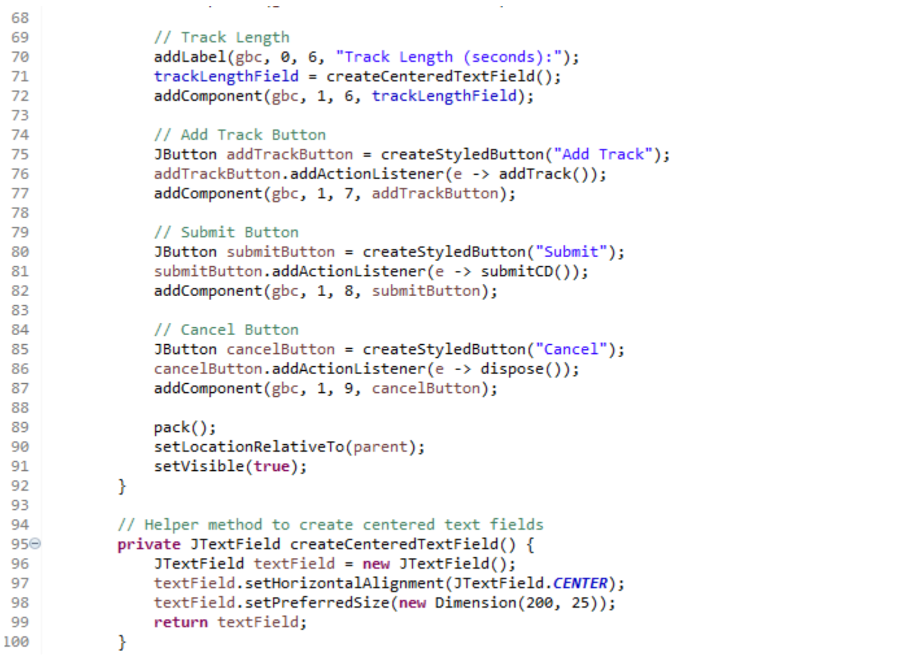


```java
                JOptionPane.INFORMATION_MESSAGE);

         // Refresh the store screen
            if (parentFrame instanceof StoreScreen) {
                ((StoreScreen) parentFrame).refreshStoreScreen();
            }

            // Close dialog
            dispose();
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(this,
                "Length must be a valid number",
                "Validation Error",
                JOptionPane.ERROR_MESSAGE);
        }
    }

    private boolean validateInput() {
        // Check if fields are empty
        if (titleField.getText().isEmpty() ||
            categoryField.getText().isEmpty() ||
            priceField.getText().isEmpty() ||
            directorField.getText().isEmpty() ||
            lengthField.getText().isEmpty()) {
            JOptionPane.showMessageDialog(this,
                "Please fill in all fields",
                "Validation Error",
                JOptionPane.ERROR_MESSAGE);
            return false;
        }

        // Validate price
        try {
            Float.parseFloat(priceField.getText());
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(this,
                "Price must be a valid number",
                "Validation Error",
                JOptionPane.ERROR_MESSAGE);
            return false;
        }

        return true;
    }
}
```

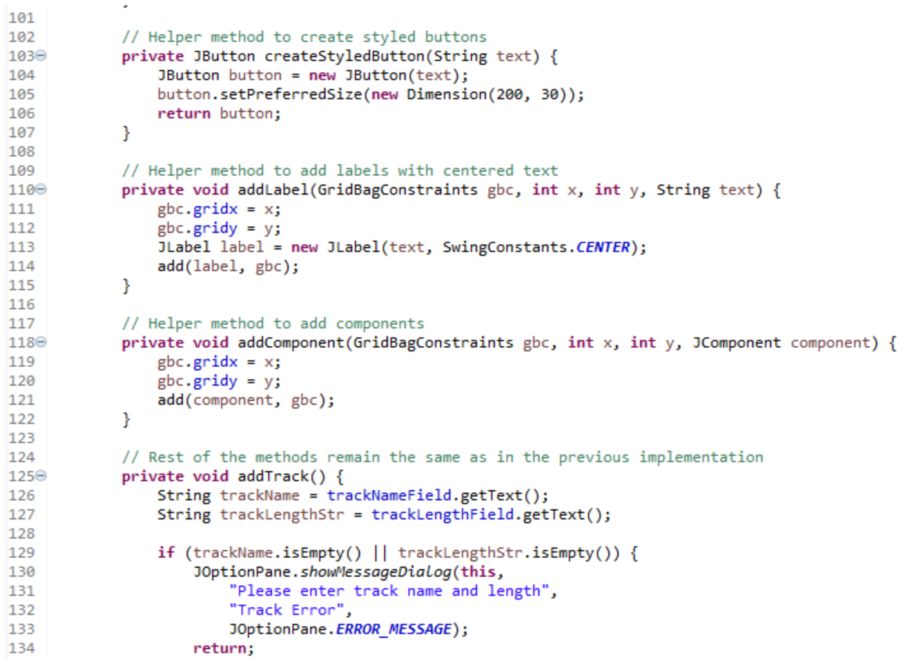
- AddCompactDiscToStoreScreen



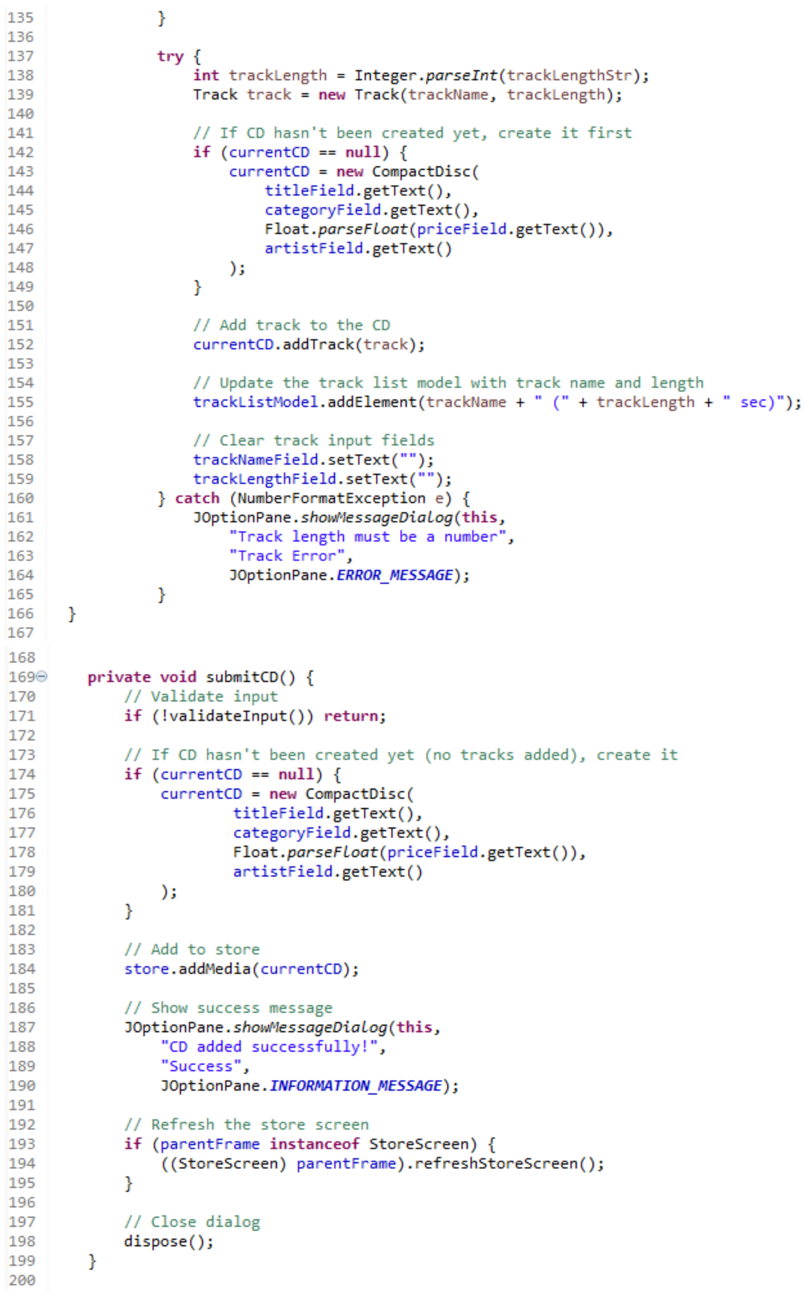
```java
package hust.soict.ITE6.aims.screen;

import javax.swing.*;
import javax.swing.GroupLayout.Alignment;

import java.awt.*;

import hust.soict.ITE6.aims.media.CompactDisc;
import hust.soict.ITE6.aims.media.Track;
import hust.soict.ITE6.aims.store.Store;

public class AddCompactDiscToStoreScreen extends JDialog {
        private Store store;
        private JFrame parentFrame;
        private JTextField titleField;
        private JTextField categoryField;
        private JTextField priceField;
        private JTextField artistField;
        private DefaultListModel<String> trackListModel;
        private JList<String> trackList;
        private JTextField trackNameField;
        private JTextField trackLengthField;
        private CompactDisc currentCD;

        public AddCompactDiscToStoreScreen(JFrame parent, Store store) {
            super(parent, "Add Compact Disc", true);
            this.store = store;
            this.parentFrame = parent;

            // Use a more compact layout
            setLayout(new GridBagLayout());
            GridBagConstraints gbc = new GridBagConstraints();
            gbc.insets = new Insets(5, 5, 5, 5);
            gbc.fill = GridBagConstraints.HORIZONTAL;


            // Title
            addLabel(gbc, 0, 0, "Title:");
            titleField = createCenteredTextField();
            addComponent(gbc, 1, 0, titleField);

            // Category
            addLabel(gbc, 0, 1, "Category:");
            categoryField = createCenteredTextField();
            addComponent(gbc, 1, 1, categoryField);

            // Price
            addLabel(gbc, 0, 2, "Price:");
            priceField = createCenteredTextField();
            addComponent(gbc, 1, 2, priceField);

            // Artist
            addLabel(gbc, 0, 3, "Artist:");
            artistField = createCenteredTextField();
            addComponent(gbc, 1, 3, artistField);

            // Tracks List
            addLabel(gbc, 0, 4, "Tracks:");
            trackListModel = new DefaultListModel<>();
            trackList = new JList<>(trackListModel);
            JScrollPane scrollPane = new JScrollPane(trackList);
            scrollPane.setPreferredSize(new Dimension(200, 100));
            addComponent(gbc, 1, 4, scrollPane);

            // Track Name
            addLabel(gbc, 0, 5, "Track Name:");
            trackNameField = createCenteredTextField();
            addComponent(gbc, 1, 5, trackNameField);
```



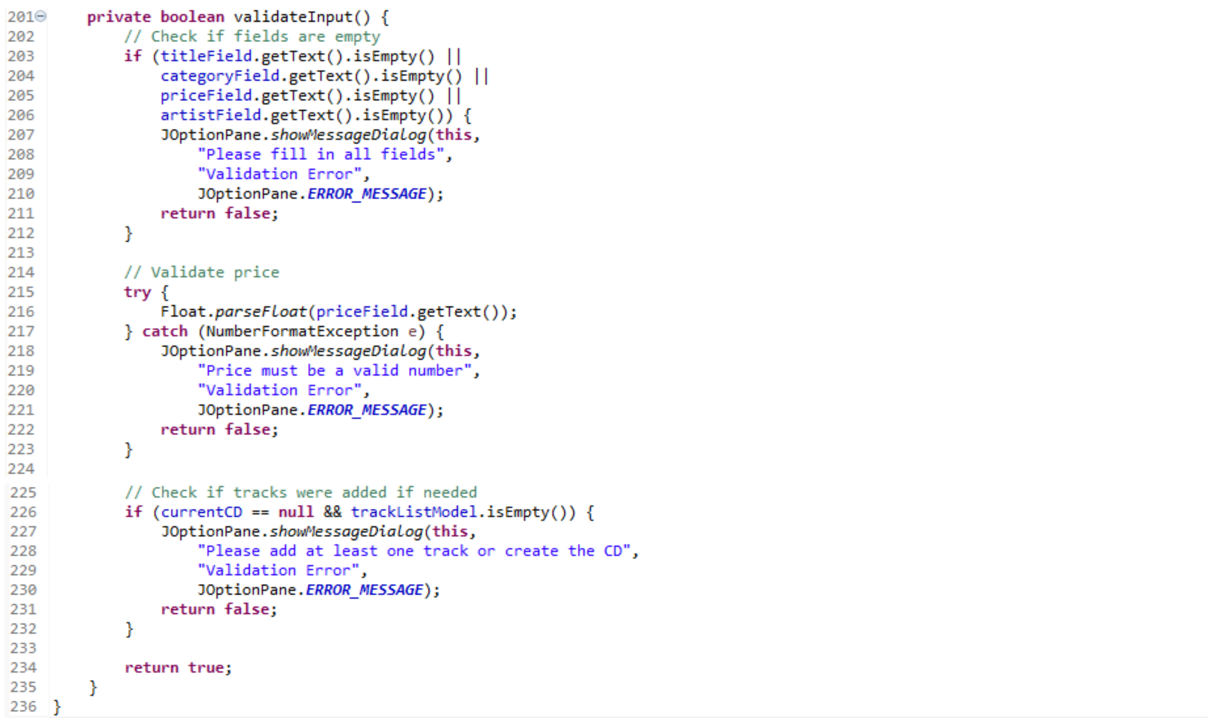
```java

        // Track Length
        addLabel(gbc, 0, 6, "Track Length (seconds):");
        trackLengthField = createCenteredTextField();
        addComponent(gbc, 1, 6, trackLengthField);

        // Add Track Button
        JButton addTrackButton = createStyledButton("Add Track");
        addTrackButton.addActionListener(e -> addTrack());
        addComponent(gbc, 1, 7, addTrackButton);

        // Submit Button
        JButton submitButton = createStyledButton("Submit");
        submitButton.addActionListener(e -> submitCD());
        addComponent(gbc, 1, 8, submitButton);

        // Cancel Button
        JButton cancelButton = createStyledButton("Cancel");
        cancelButton.addActionListener(e -> dispose());
        addComponent(gbc, 1, 9, cancelButton);

        pack();
        setLocationRelativeTo(parent);
        setVisible(true);
    }

    // Helper method to create centered text fields
    private JTextField createCenteredTextField() {
        JTextField textField = new JTextField();
        textField.setHorizontalAlignment(JTextField.CENTER);
        textField.setPreferredSize(new Dimension(200, 25));
        return textField;
    }
```



```java

    // Helper method to create styled buttons
    private JButton createStyledButton(String text) {
        JButton button = new JButton(text);
        button.setPreferredSize(new Dimension(200, 30));
        return button;
    }

    // Helper method to add labels with centered text
    private void addLabel(GridBagConstraints gbc, int x, int y, String text) {
        gbc.gridx = x;
        gbc.gridy = y;
        JLabel label = new JLabel(text, SwingConstants.CENTER);
        add(label, gbc);
    }

    // Helper method to add components
    private void addComponent(GridBagConstraints gbc, int x, int y, JComponent component) {
        gbc.gridx = x;
        gbc.gridy = y;
        add(component, gbc);
    }

    // Rest of the methods remain the same as in the previous implementation
    private void addTrack() {
        String trackName = trackNameField.getText();
        String trackLengthStr = trackLengthField.getText();

        if (trackName.isEmpty() || trackLengthStr.isEmpty()) {
            JOptionPane.showMessageDialog(this,
                "Please enter track name and length",
                "Track Error",
                JOptionPane.ERROR_MESSAGE);
            return;
```

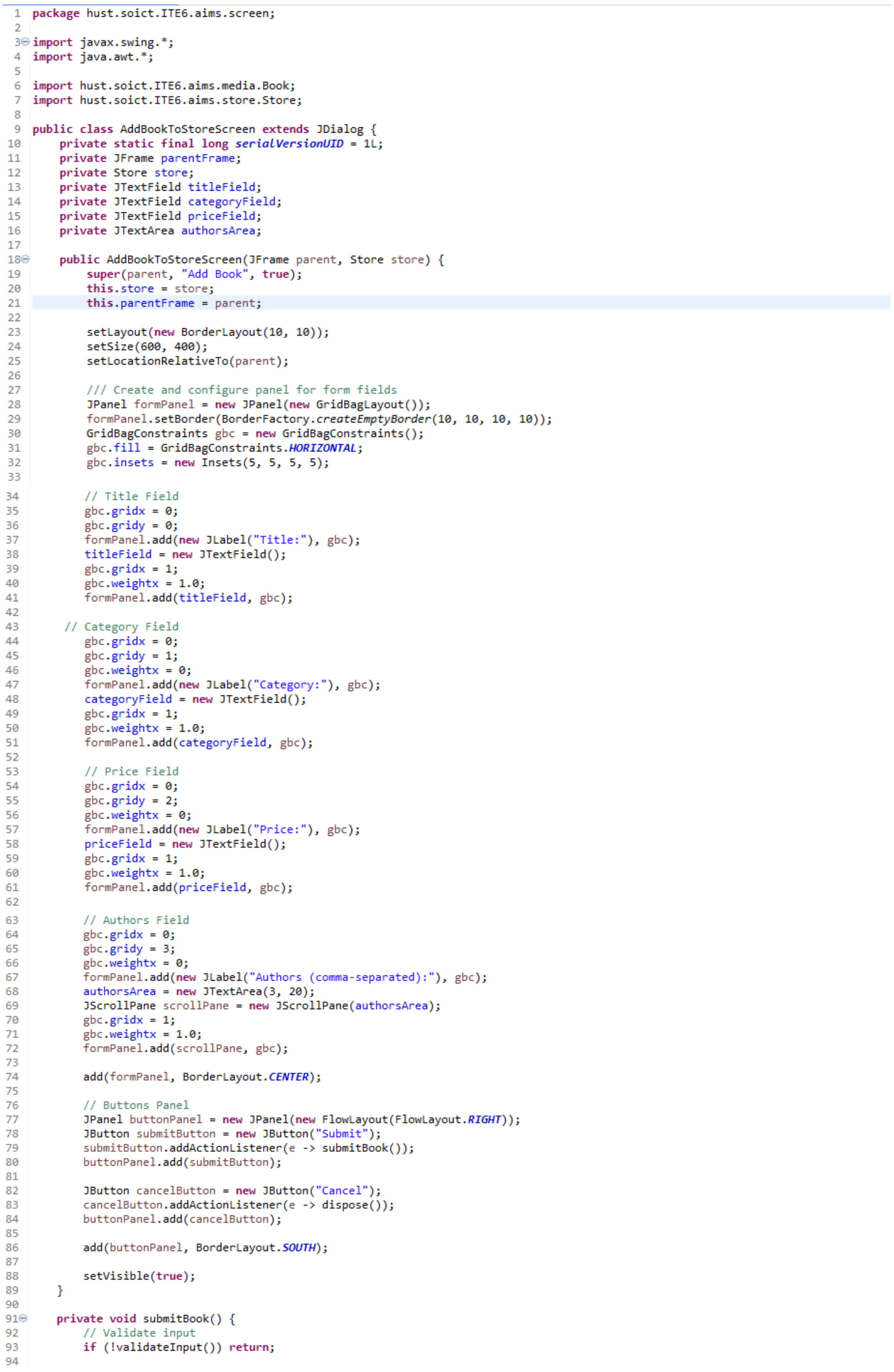


```java
            }

            try {
                int trackLength = Integer.parseInt(trackLengthStr);
                Track track = new Track(trackName, trackLength);

                // If CD hasn't been created yet, create it first
                if (currentCD == null) {
                    currentCD = new CompactDisc(
                        titleField.getText(),
                        categoryField.getText(),
                        Float.parseFloat(priceField.getText()),
                        artistField.getText()
                    );
                }

                // Add track to the CD
                currentCD.addTrack(track);

                // Update the track list model with track name and length
                trackListModel.addElement(trackName + " (" + trackLength + " sec)");

                // Clear track input fields
                trackNameField.setText("");
                trackLengthField.setText("");
            } catch (NumberFormatException e) {
                JOptionPane.showMessageDialog(this,
                    "Track length must be a number",
                    "Track Error",
                    JOptionPane.ERROR_MESSAGE);
            }
  }

    private void submitCD() {
        // Validate input
        if (!validateInput()) return;

        // If CD hasn't been created yet (no tracks added), create it
        if (currentCD == null) {
            currentCD = new CompactDisc(
                    titleField.getText(),
                    categoryField.getText(),
                    Float.parseFloat(priceField.getText()),
                    artistField.getText()
            );
        }

        // Add to store
        store.addMedia(currentCD);

        // Show success message
        JOptionPane.showMessageDialog(this,
            "CD added successfully!",
            "Success",
            JOptionPane.INFORMATION_MESSAGE);

        // Refresh the store screen
        if (parentFrame instanceof StoreScreen) {
            ((StoreScreen) parentFrame).refreshStoreScreen();
        }

        // Close dialog
        dispose();
    }
```

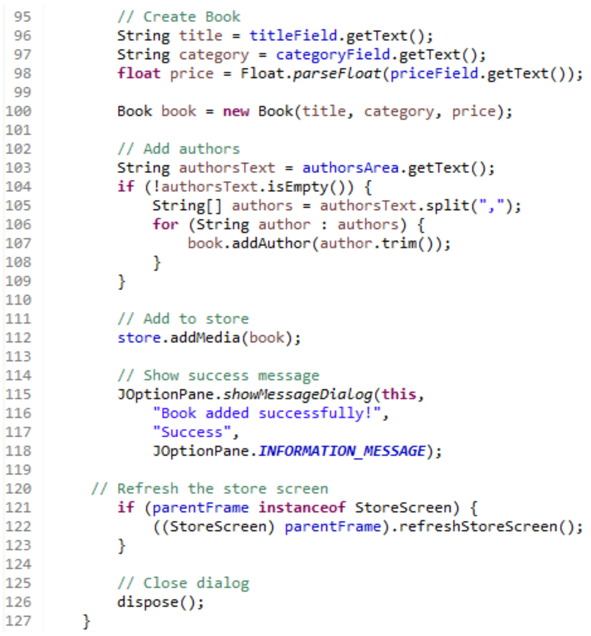


```java
    private boolean validateInput() {
        // Check if fields are empty
        if (titleField.getText().isEmpty() ||
            categoryField.getText().isEmpty() ||
            priceField.getText().isEmpty() ||
            artistField.getText().isEmpty()) {
            JOptionPane.showMessageDialog(this,
                "Please fill in all fields",
                "Validation Error",
                JOptionPane.ERROR_MESSAGE);
            return false;
        }

        // Validate price
        try {
            Float.parseFloat(priceField.getText());
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(this,
                "Price must be a valid number",
                "Validation Error",
                JOptionPane.ERROR_MESSAGE);
            return false;
        }

        // Check if tracks were added if needed
        if (currentCD == null && trackListModel.isEmpty()) {
            JOptionPane.showMessageDialog(this,
                "Please add at least one track or create the CD",
                "Validation Error",
                JOptionPane.ERROR_MESSAGE);
            return false;
        }

        return true;
    }
}
```

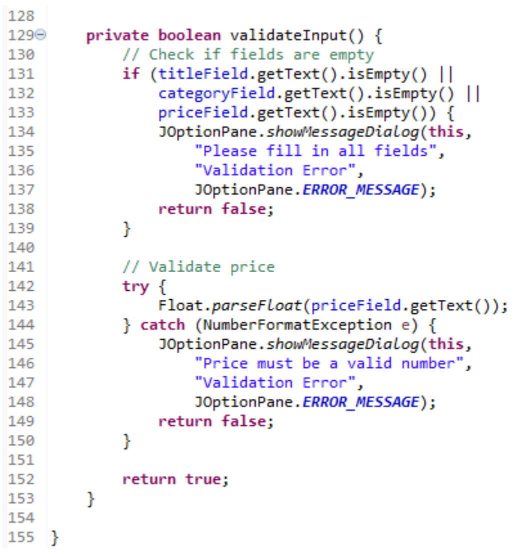
- AddBookToStoreScreen



```java
package hust.soict.ITE6.aims.screen;

import javax.swing.*;
import java.awt.*;

import hust.soict.ITE6.aims.media.Book;
import hust.soict.ITE6.aims.store.Store;

public class AddBookToStoreScreen extends JDialog {
    private static final long serialVersionUID = 1L;
    private JFrame parentFrame;
    private Store store;
    private JTextField titleField;
    private JTextField categoryField;
    private JTextField priceField;
    private JTextArea authorsArea;

    public AddBookToStoreScreen(JFrame parent, Store store) {
        super(parent, "Add Book", true);
        this.store = store;
        this.parentFrame = parent;

        setLayout(new BorderLayout(10, 10));
        setSize(600, 400);
        setLocationRelativeTo(parent);

        /// Create and configure panel for form fields
        JPanel formPanel = new JPanel(new GridBagLayout());
        formPanel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
        GridBagConstraints gbc = new GridBagConstraints();
        gbc.fill = GridBagConstraints.HORIZONTAL;
        gbc.insets = new Insets(5, 5, 5, 5);

        // Title Field
        gbc.gridx = 0;
        gbc.gridy = 0;
        formPanel.add(new JLabel("Title:"), gbc);
        titleField = new JTextField();
        gbc.gridx = 1;
        gbc.weightx = 1.0;
        formPanel.add(titleField, gbc);

     // Category Field
        gbc.gridx = 0;
        gbc.gridy = 1;
        gbc.weightx = 0;
        formPanel.add(new JLabel("Category:"), gbc);
        categoryField = new JTextField();
        gbc.gridx = 1;
        gbc.weightx = 1.0;
        formPanel.add(categoryField, gbc);

        // Price Field
        gbc.gridx = 0;
        gbc.gridy = 2;
        gbc.weightx = 0;
        formPanel.add(new JLabel("Price:"), gbc);
        priceField = new JTextField();
        gbc.gridx = 1;
        gbc.weightx = 1.0;
        formPanel.add(priceField, gbc);

        // Authors Field
        gbc.gridx = 0;
        gbc.gridy = 3;
        gbc.weightx = 0;
        formPanel.add(new JLabel("Authors (comma-separated):"), gbc);
        authorsArea = new JTextArea(3, 20);
        JScrollPane scrollPane = new JScrollPane(authorsArea);
        gbc.gridx = 1;
        gbc.weightx = 1.0;
        formPanel.add(scrollPane, gbc);

        add(formPanel, BorderLayout.CENTER);

        // Buttons Panel
        JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.RIGHT));
        JButton submitButton = new JButton("Submit");
        submitButton.addActionListener(e -> submitBook());
        buttonPanel.add(submitButton);

        JButton cancelButton = new JButton("Cancel");
        cancelButton.addActionListener(e -> dispose());
        buttonPanel.add(cancelButton);

        add(buttonPanel, BorderLayout.SOUTH);

        setVisible(true);
    }

    private void submitBook() {
        // Validate input
        if (!validateInput()) return;

```

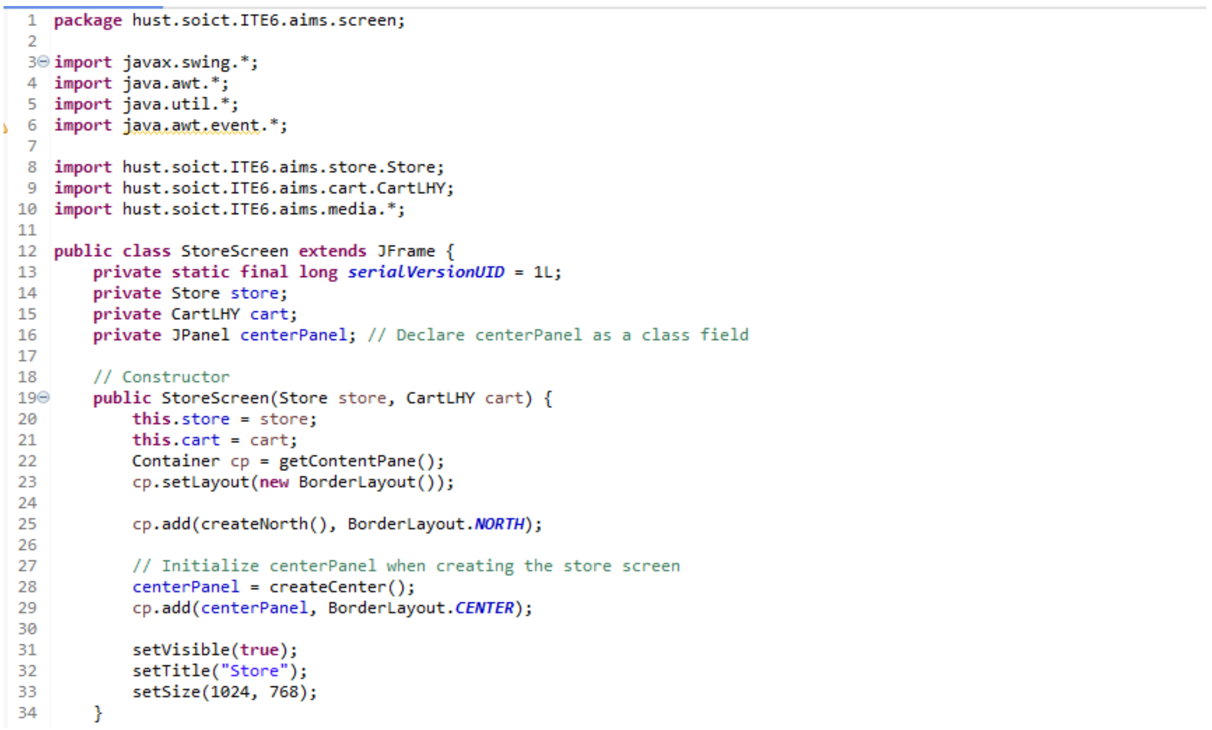


```java
        // Create Book
        String title = titleField.getText();
        String category = categoryField.getText();
        float price = Float.parseFloat(priceField.getText());

        Book book = new Book(title, category, price);

        // Add authors
        String authorsText = authorsArea.getText();
        if (!authorsText.isEmpty()) {
            String[] authors = authorsText.split(",");
            for (String author : authors) {
                book.addAuthor(author.trim());
            }
        }

        // Add to store
        store.addMedia(book);

        // Show success message
        JOptionPane.showMessageDialog(this,
            "Book added successfully!",
            "Success",
            JOptionPane.INFORMATION_MESSAGE);

       // Refresh the store screen
        if (parentFrame instanceof StoreScreen) {
            ((StoreScreen) parentFrame).refreshStoreScreen();
        }

        // Close dialog
        dispose();
    }
```

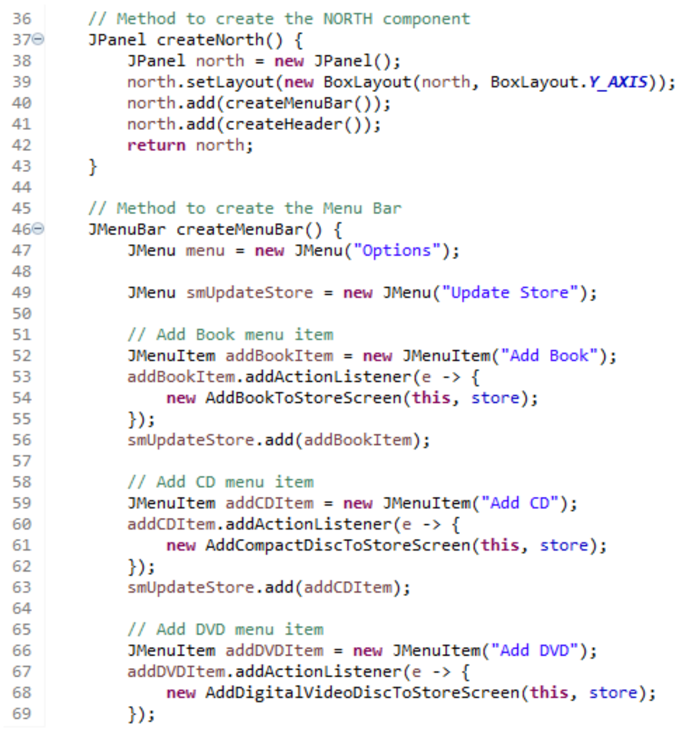


```java

    private boolean validateInput() {
        // Check if fields are empty
        if (titleField.getText().isEmpty() ||
            categoryField.getText().isEmpty() ||
            priceField.getText().isEmpty()) {
            JOptionPane.showMessageDialog(this,
                "Please fill in all fields",
                "Validation Error",
                JOptionPane.ERROR_MESSAGE);
            return false;
        }

        // Validate price
        try {
            Float.parseFloat(priceField.getText());
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(this,
                "Price must be a valid number",
                "Validation Error",
                JOptionPane.ERROR_MESSAGE);
            return false;
        }

        return true;
    }

}
```

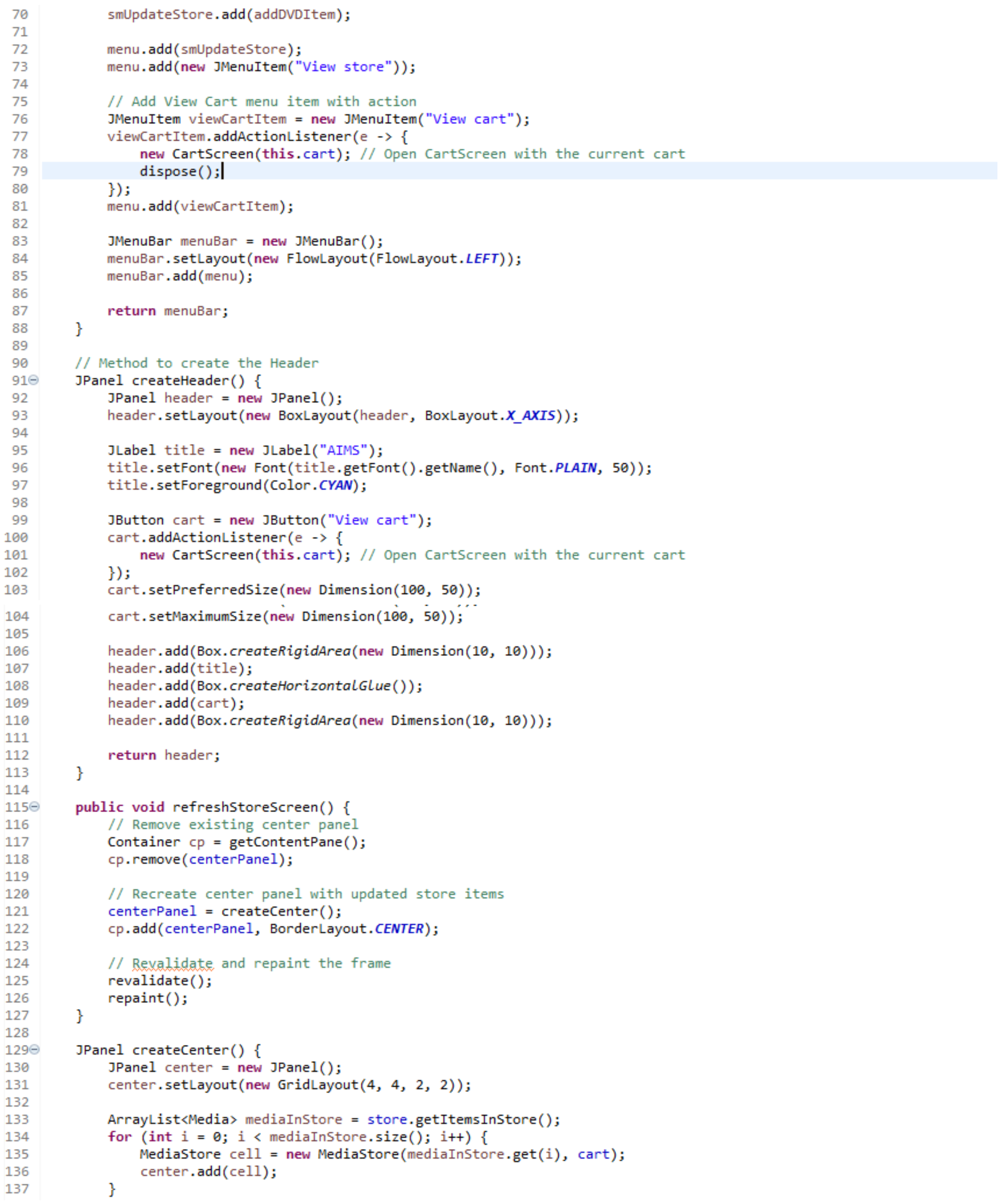
- Update **StoreScreen**



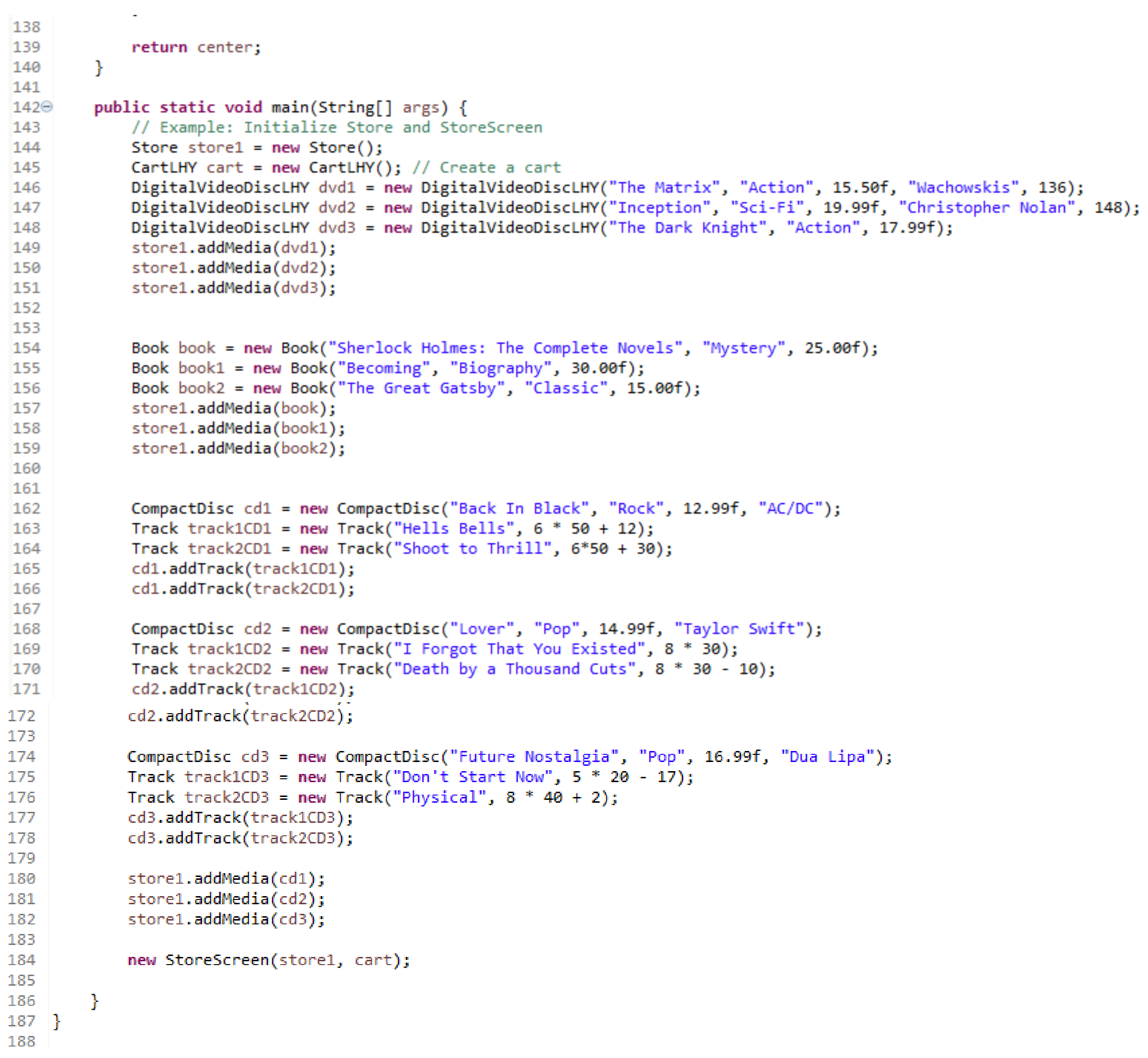
```java
package hust.soict.ITE6.aims.screen;

import javax.swing.*;
import java.awt.*;
import java.util.*;
import java.awt.event.*;

import hust.soict.ITE6.aims.store.Store;
import hust.soict.ITE6.aims.cart.CartLHY;
import hust.soict.ITE6.aims.media.*;

public class StoreScreen extends JFrame {
    private static final long serialVersionUID = 1L;
    private Store store;
    private CartLHY cart;
    private JPanel centerPanel; // Declare centerPanel as a class field

    // Constructor
    public StoreScreen(Store store, CartLHY cart) {
        this.store = store;
        this.cart = cart;
        Container cp = getContentPane();
        cp.setLayout(new BorderLayout());

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

        // Initialize centerPanel when creating the store screen
        centerPanel = createCenter();
        cp.add(centerPanel, BorderLayout.CENTER);

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



```java
    // Method to create the NORTH component
    JPanel createNorth() {
        JPanel north = new JPanel();
        north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
        north.add(createMenuBar());
        north.add(createHeader());
        return north;
    }

    // Method to create the Menu Bar
    JMenuBar createMenuBar() {
        JMenu menu = new JMenu("Options");

        JMenu smUpdateStore = new JMenu("Update Store");

        // Add Book menu item
        JMenuItem addBookItem = new JMenuItem("Add Book");
        addBookItem.addActionListener(e -> {
            new AddBookToStoreScreen(this, store);
        });
        smUpdateStore.add(addBookItem);

        // Add CD menu item
        JMenuItem addCDItem = new JMenuItem("Add CD");
        addCDItem.addActionListener(e -> {
            new AddCompactDiscToStoreScreen(this, store);
        });
        smUpdateStore.add(addCDItem);

        // Add DVD menu item
        JMenuItem addDVDItem = new JMenuItem("Add DVD");
        addDVDItem.addActionListener(e -> {
            new AddDigitalVideoDiscToStoreScreen(this, store);
        });
```

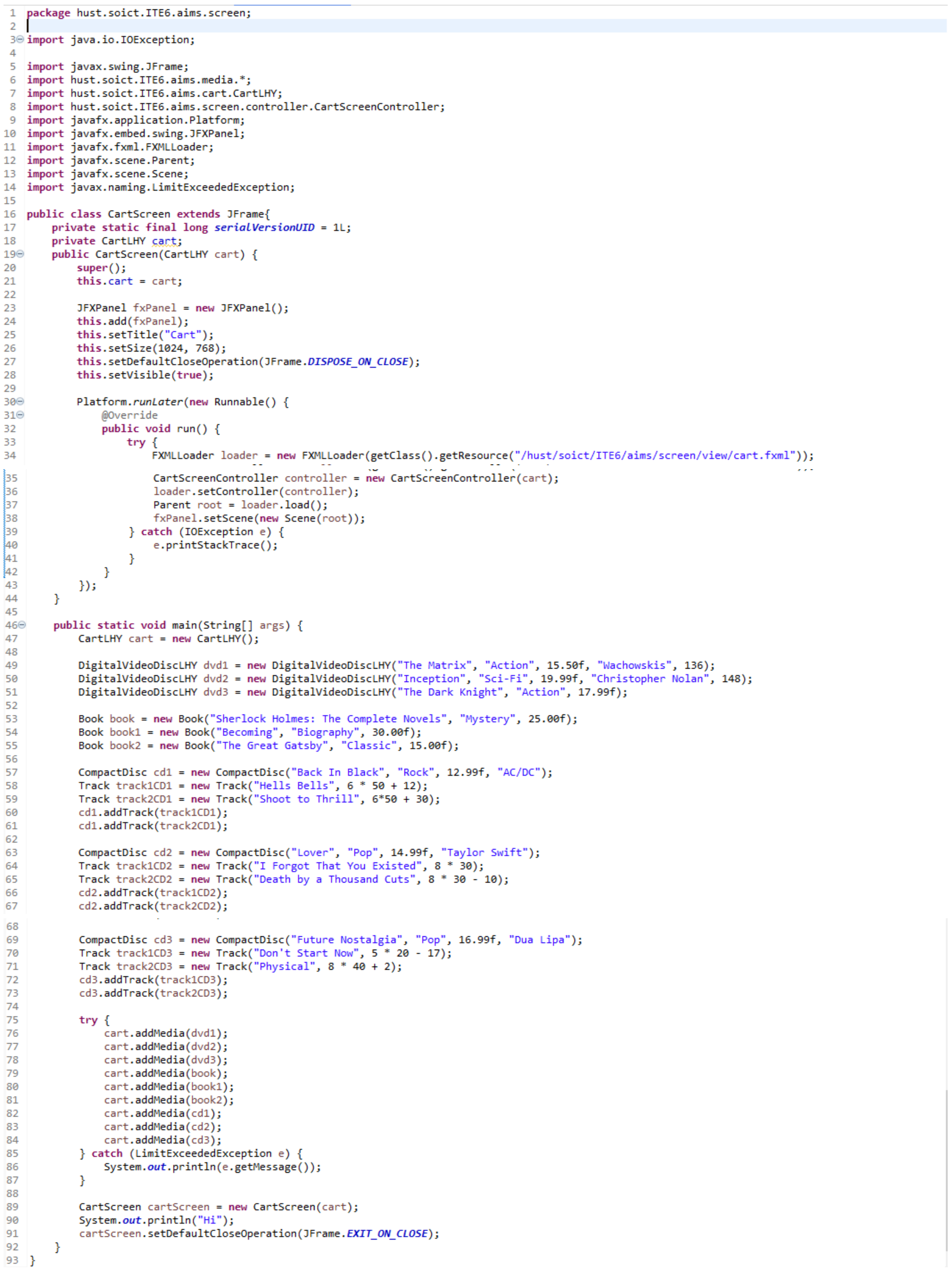


```java
        smUpdateStore.add(addDVDItem);

        menu.add(smUpdateStore);
        menu.add(new JMenuItem("View store"));

        // Add View Cart menu item with action
        JMenuItem viewCartItem = new JMenuItem("View cart");
        viewCartItem.addActionListener(e -> {
            new CartScreen(this.cart); // Open CartScreen with the current cart
            dispose();
        });
        menu.add(viewCartItem);

        JMenuBar menuBar = new JMenuBar();
        menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
        menuBar.add(menu);

        return menuBar;
    }

    // Method to create the Header
    JPanel createHeader() {
        JPanel header = new JPanel();
        header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));

        JLabel title = new JLabel("AIMS");
        title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
        title.setForeground(Color.CYAN);

        JButton cart = new JButton("View cart");
        cart.addActionListener(e -> {
            new CartScreen(this.cart); // Open CartScreen with the current cart
        });
        cart.setPreferredSize(new Dimension(100, 50));
        cart.setMaximumSize(new Dimension(100, 50));

        header.add(Box.createRigidArea(new Dimension(10, 10)));
        header.add(title);
        header.add(Box.createHorizontalGlue());
        header.add(cart);
        header.add(Box.createRigidArea(new Dimension(10, 10)));

        return header;
    }

    public void refreshStoreScreen() {
        // Remove existing center panel
        Container cp = getContentPane();
        cp.remove(centerPanel);

        // Recreate center panel with updated store items
        centerPanel = createCenter();
        cp.add(centerPanel, BorderLayout.CENTER);

        // Revalidate and repaint the frame
        revalidate();
        repaint();
    }

    JPanel createCenter() {
        JPanel center = new JPanel();
        center.setLayout(new GridLayout(4, 4, 2, 2));

        ArrayList<Media> mediaInStore = store.getItemsInStore();
        for (int i = 0; i < mediaInStore.size(); i++) {
            MediaStore cell = new MediaStore(mediaInStore.get(i), cart);
            center.add(cell);
        }
```



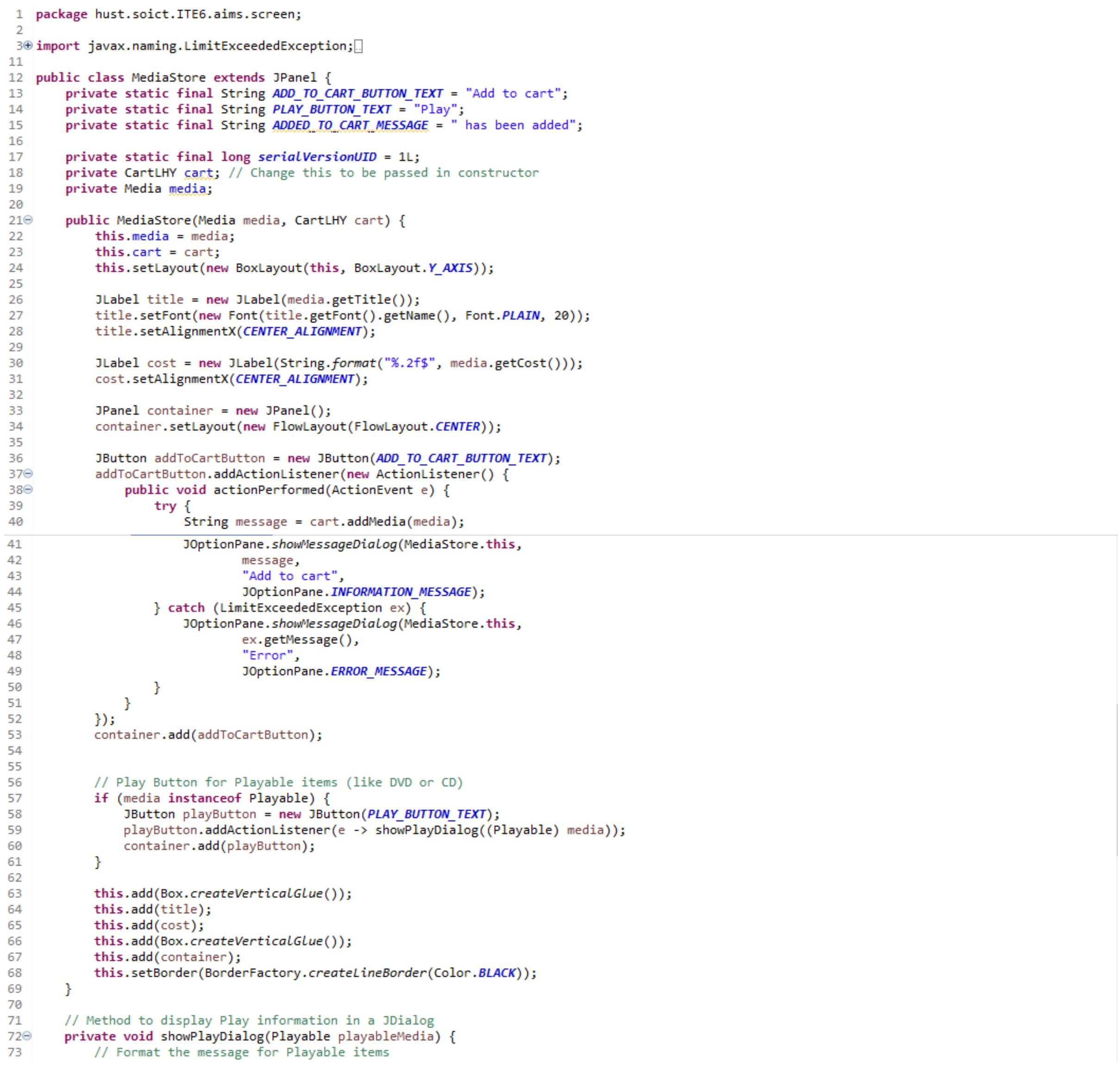
```java

        return center;
    }

    public static void main(String[] args) {
        // Example: Initialize Store and StoreScreen
        Store store1 = new Store();
        CartLHY cart = new CartLHY(); // Create a cart
        DigitalVideoDiscLHY dvd1 = new DigitalVideoDiscLHY("The Matrix", "Action", 15.50f, "Wachowskis", 136);
        DigitalVideoDiscLHY dvd2 = new DigitalVideoDiscLHY("Inception", "Sci-Fi", 19.99f, "Christopher Nolan", 148);
        DigitalVideoDiscLHY dvd3 = new DigitalVideoDiscLHY("The Dark Knight", "Action", 17.99f);
        store1.addMedia(dvd1);
        store1.addMedia(dvd2);
        store1.addMedia(dvd3);


        Book book = new Book("Sherlock Holmes: The Complete Novels", "Mystery", 25.00f);
        Book book1 = new Book("Becoming", "Biography", 30.00f);
        Book book2 = new Book("The Great Gatsby", "Classic", 15.00f);
        store1.addMedia(book);
        store1.addMedia(book1);
        store1.addMedia(book2);


        CompactDisc cd1 = new CompactDisc("Back In Black", "Rock", 12.99f, "AC/DC");
        Track track1CD1 = new Track("Hells Bells", 6 * 50 + 12);
        Track track2CD1 = new Track("Shoot to Thrill", 6*50 + 30);
        cd1.addTrack(track1CD1);
        cd1.addTrack(track2CD1);

        CompactDisc cd2 = new CompactDisc("Lover", "Pop", 14.99f, "Taylor Swift");
        Track track1CD2 = new Track("I Forgot That You Existed", 8 * 30);
        Track track2CD2 = new Track("Death by a Thousand Cuts", 8 * 30 - 10);
        cd2.addTrack(track1CD2);
        cd2.addTrack(track2CD2);

        CompactDisc cd3 = new CompactDisc("Future Nostalgia", "Pop", 16.99f, "Dua Lipa");
        Track track1CD3 = new Track("Don't Start Now", 5 * 20 - 17);
        Track track2CD3 = new Track("Physical", 8 * 40 + 2);
        cd3.addTrack(track1CD3);
        cd3.addTrack(track2CD3);

        store1.addMedia(cd1);
        store1.addMedia(cd2);
        store1.addMedia(cd3);

        new StoreScreen(store1, cart);

    }
}
```

- Update **CartScreen**

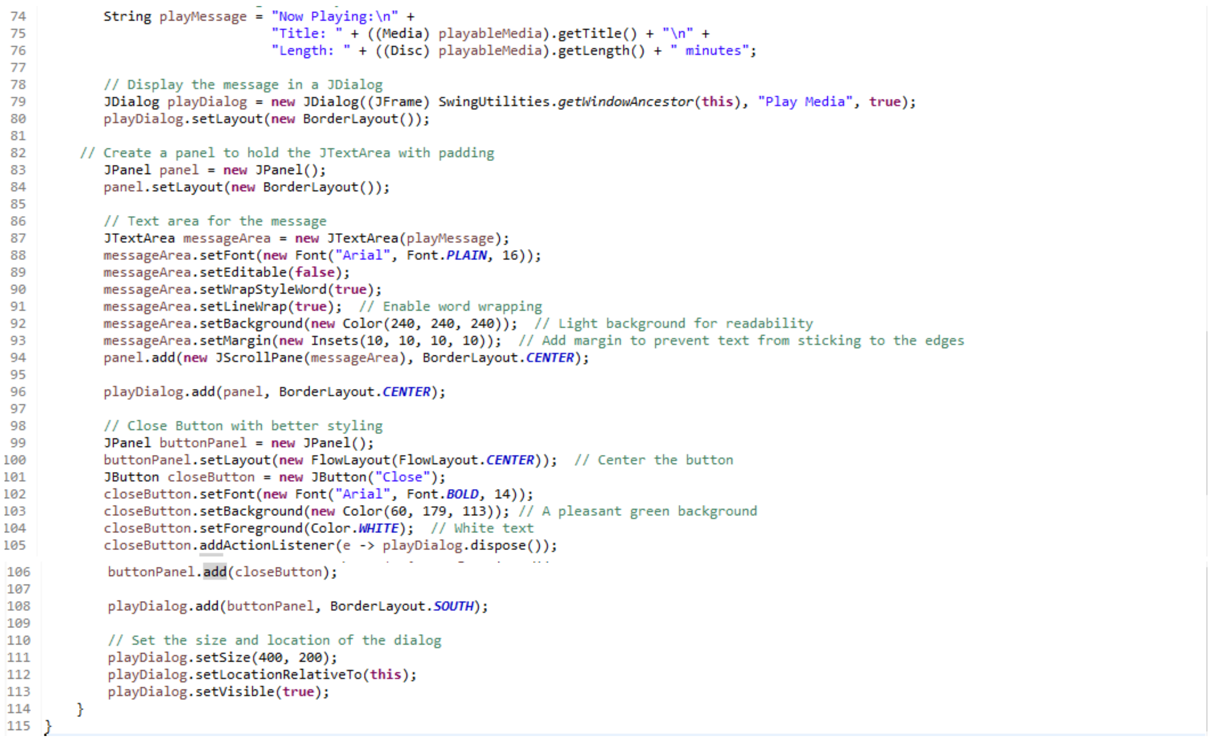


```java
package hust.soict.ITE6.aims.screen;

import java.io.IOException;

import javax.swing.JFrame;
import hust.soict.ITE6.aims.media.*;
import hust.soict.ITE6.aims.cart.CartLHY;
import hust.soict.ITE6.aims.screen.controller.CartScreenController;
import javafx.application.Platform;
import javafx.embed.swing.JFXPanel;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javax.naming.LimitExceededException;

public class CartScreen extends JFrame{
    private static final long serialVersionUID = 1L;
    private CartLHY cart;
    public CartScreen(CartLHY cart) {
        super();
        this.cart = cart;

        JFXPanel fxPanel = new JFXPanel();
        this.add(fxPanel);
        this.setTitle("Cart");
        this.setSize(1024, 768);
        this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        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();
                }
            }
        });
    }

    public static void main(String[] args) {
        CartLHY cart = new CartLHY();

        DigitalVideoDiscLHY dvd1 = new DigitalVideoDiscLHY("The Matrix", "Action", 15.50f, "Wachowskis", 136);
        DigitalVideoDiscLHY dvd2 = new DigitalVideoDiscLHY("Inception", "Sci-Fi", 19.99f, "Christopher Nolan", 148);
        DigitalVideoDiscLHY dvd3 = new DigitalVideoDiscLHY("The Dark Knight", "Action", 17.99f);

        Book book = new Book("Sherlock Holmes: The Complete Novels", "Mystery", 25.00f);
        Book book1 = new Book("Becoming", "Biography", 30.00f);
        Book book2 = new Book("The Great Gatsby", "Classic", 15.00f);

        CompactDisc cd1 = new CompactDisc("Back In Black", "Rock", 12.99f, "AC/DC");
        Track track1CD1 = new Track("Hells Bells", 6 * 50 + 12);
        Track track2CD1 = new Track("Shoot to Thrill", 6*50 + 30);
        cd1.addTrack(track1CD1);
        cd1.addTrack(track2CD1);

        CompactDisc cd2 = new CompactDisc("Lover", "Pop", 14.99f, "Taylor Swift");
        Track track1CD2 = new Track("I Forgot That You Existed", 8 * 30);
        Track track2CD2 = new Track("Death by a Thousand Cuts", 8 * 30 - 10);
        cd2.addTrack(track1CD2);
        cd2.addTrack(track2CD2);

        CompactDisc cd3 = new CompactDisc("Future Nostalgia", "Pop", 16.99f, "Dua Lipa");
        Track track1CD3 = new Track("Don't Start Now", 5 * 20 - 17);
        Track track2CD3 = new Track("Physical", 8 * 40 + 2);
        cd3.addTrack(track1CD3);
        cd3.addTrack(track2CD3);

        try {
            cart.addMedia(dvd1);
            cart.addMedia(dvd2);
            cart.addMedia(dvd3);
            cart.addMedia(book);
            cart.addMedia(book1);
            cart.addMedia(book2);
            cart.addMedia(cd1);
            cart.addMedia(cd2);
            cart.addMedia(cd3);
        } catch (LimitExceededException e) {
            System.out.println(e.getMessage());
        }

        CartScreen cartScreen = new CartScreen(cart);
        System.out.println("Hi");
        cartScreen.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

- Update **MediaStore**

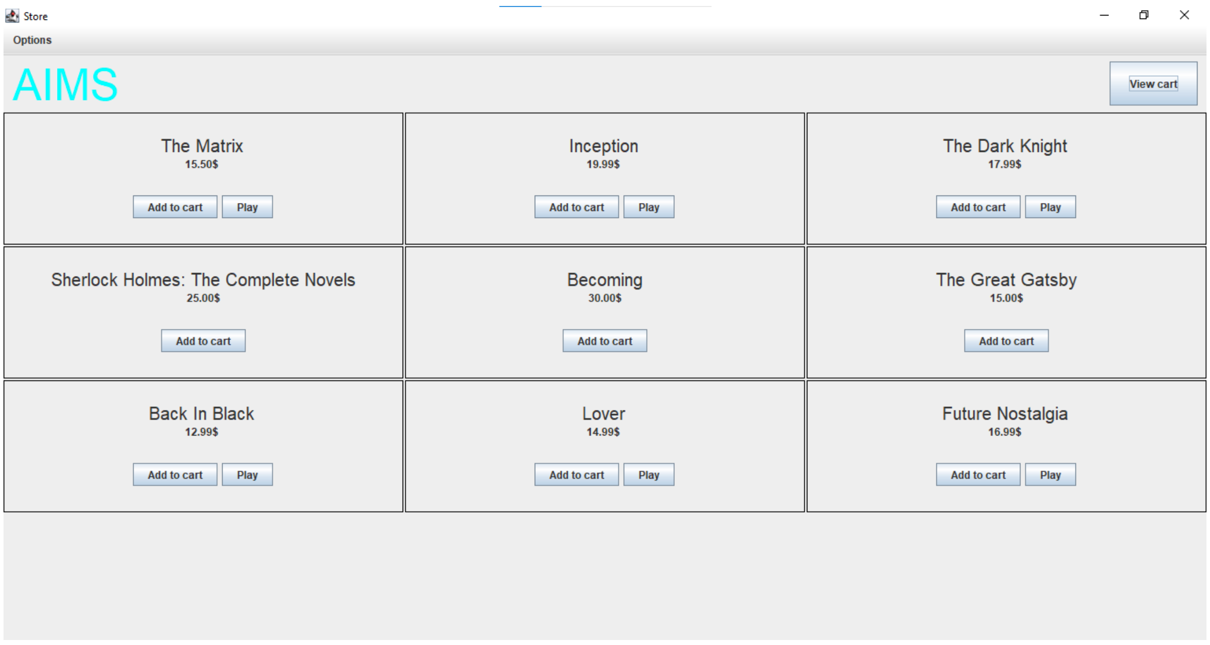


```java
package hust.soict.ITE6.aims.screen;

import javax.naming.LimitExceededException;

public class MediaStore extends JPanel {
    private static final String ADD_TO_CART_BUTTON_TEXT = "Add to cart";
    private static final String PLAY_BUTTON_TEXT = "Play";
    private static final String ADDED_TO_CART_MESSAGE = " has been added";

    private static final long serialVersionUID = 1L;
    private CartLHY cart; // Change this to be passed in constructor
    private Media media;

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

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

        JLabel cost = new JLabel(String.format("%.2f$", media.getCost()));
        cost.setAlignmentX(CENTER_ALIGNMENT);

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

        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);


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

        this.add(Box.createVerticalGlue());
        this.add(title);
        this.add(cost);
        this.add(Box.createVerticalGlue());
        this.add(container);
        this.setBorder(BorderFactory.createLineBorder(Color.BLACK));
    }

    // Method to display Play information in a JDialog
    private void showPlayDialog(Playable playableMedia) {
        // Format the message for Playable items
```



```java
        String playMessage = "Now Playing:\n" +
                             "Title: " + ((Media) playableMedia).getTitle() + "\n" +
                             "Length: " + ((Disc) playableMedia).getLength() + " minutes";

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

      // Create a panel to hold the JTextArea with padding
        JPanel panel = new JPanel();
        panel.setLayout(new BorderLayout());

        // Text area for the message
        JTextArea messageArea = new JTextArea(playMessage);
        messageArea.setFont(new Font("Arial", Font.PLAIN, 16));
        messageArea.setEditable(false);
        messageArea.setWrapStyleWord(true);
        messageArea.setLineWrap(true);  // Enable word wrapping
        messageArea.setBackground(new Color(240, 240, 240));  // Light background for readability
        messageArea.setMargin(new Insets(10, 10, 10, 10));  // Add margin to prevent text from sticking to the edges
        panel.add(new JScrollPane(messageArea), BorderLayout.CENTER);

        playDialog.add(panel, BorderLayout.CENTER);

        // Close Button with better styling
        JPanel buttonPanel = new JPanel();
        buttonPanel.setLayout(new FlowLayout(FlowLayout.CENTER));  // Center the button
        JButton closeButton = new JButton("Close");
        closeButton.setFont(new Font("Arial", Font.BOLD, 14));
        closeButton.setBackground(new Color(60, 179, 113)); // A pleasant green background
        closeButton.setForeground(Color.WHITE);  // White text
        closeButton.addActionListener(e -> playDialog.dispose());
        buttonPanel.add(closeButton);

        playDialog.add(buttonPanel, BorderLayout.SOUTH);

        // Set the size and location of the dialog
        playDialog.setSize(400, 200);
        playDialog.setLocationRelativeTo(this);
        playDialog.setVisible(true);
    }
}
```

- Complete the Aims GUI application



The Matrix
15.50$

Inception
19.99$

The Dark Knight
17.99$

Sherlock Holmes: The Complete Novels
25.00$

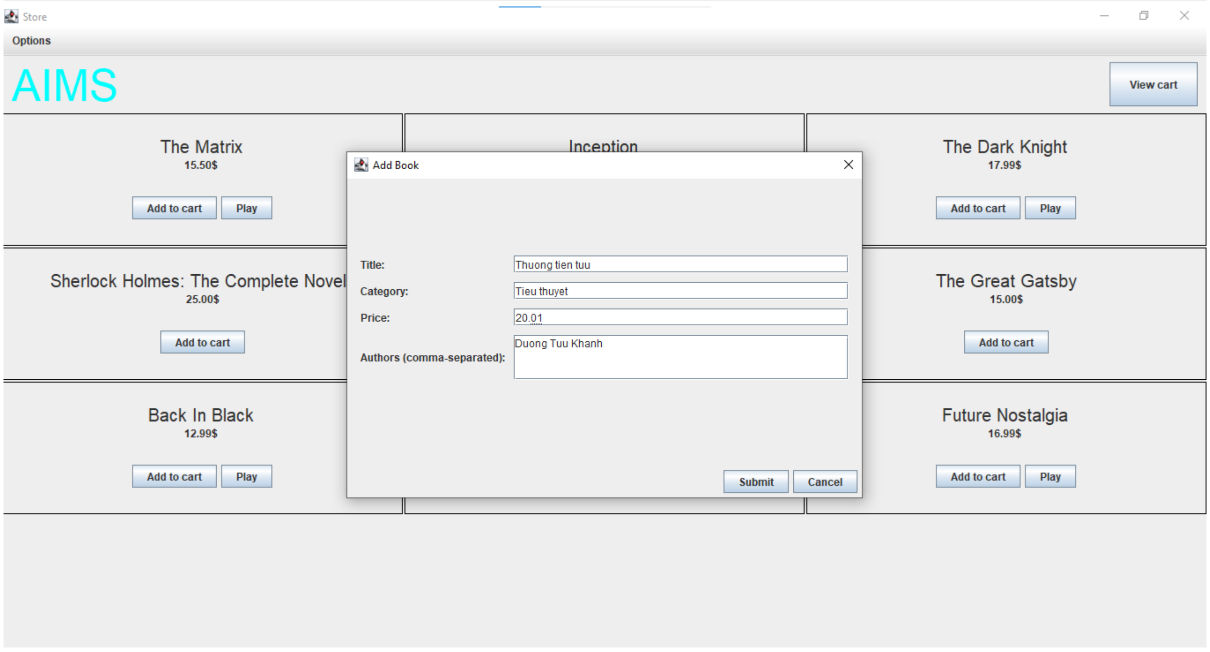
Becoming
30.00$

The Great Gatsby
15.00$

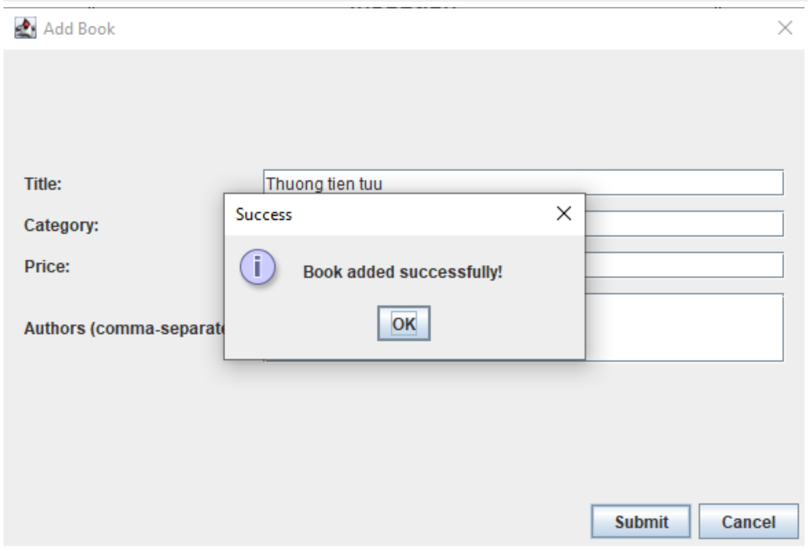
Back In Black
12.99$

Lover
14.99$

Future Nostalgia
16.99$

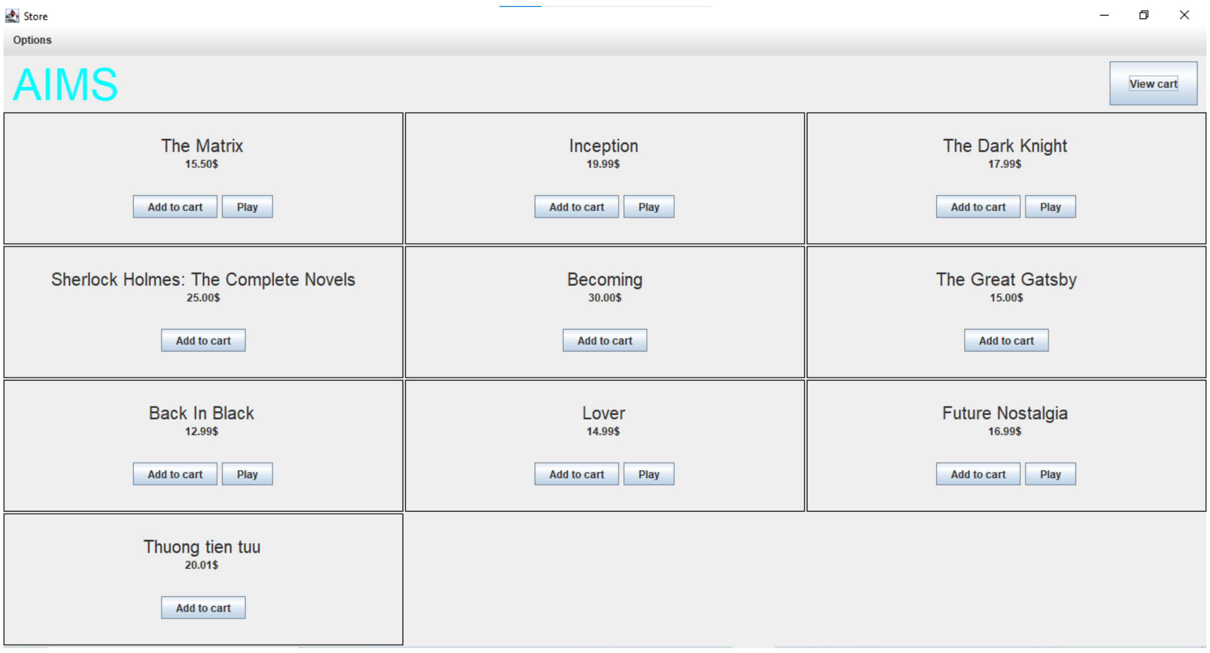


Title: Thuong tien tuu
Category: Tieu thuyet
Price: 20.01
Authors (comma-separated): Duong Tuu Khanh



Title: Thuong tien tuu
Category:
Price:
Authors (comma-separat

Book added successfully!



The Matrix
15.50$

Inception
19.99$

The Dark Knight
17.99$

Sherlock Holmes: The Complete Novels
25.00$

Becoming
30.00$

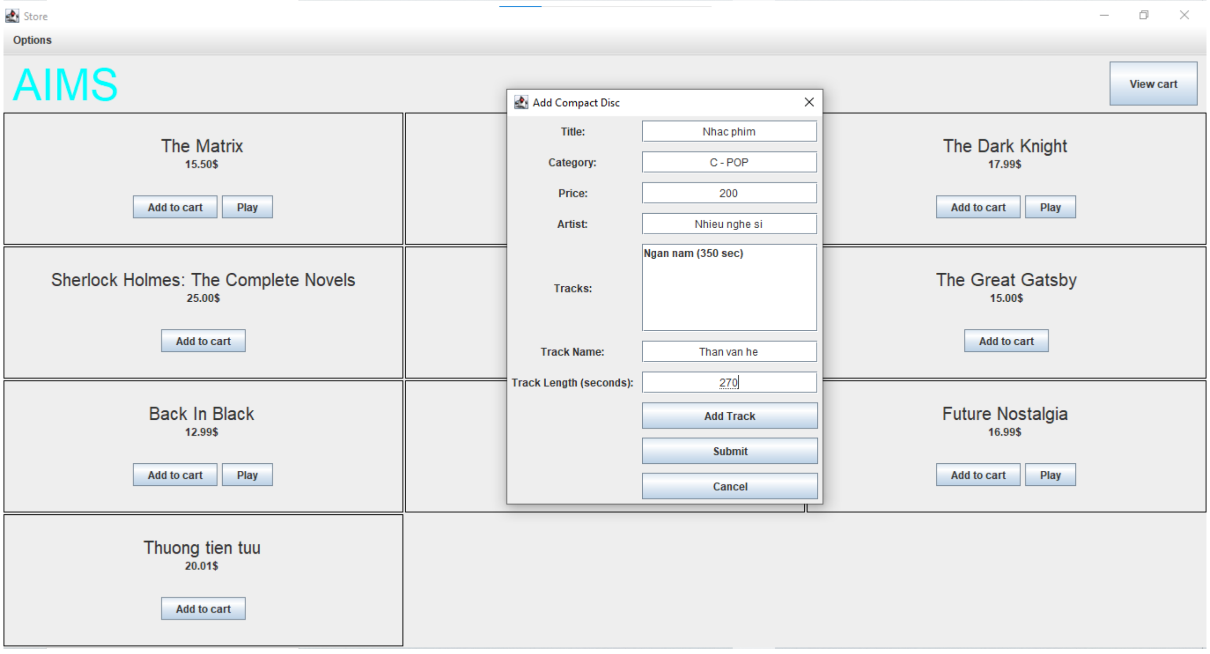
The Great Gatsby
15.00$

Back In Black
12.99$

Lover
14.99$

Future Nostalgia
16.99$

Thuong tien tuu
20.01$



Title:	Nhac phim
Category:	C - POP
Price:	200
Artist:	Nhieu nghe si
Tracks:	Ngan nam (350 sec)
Track Name:	Than van he
Track Length (seconds):	270

The Matrix
15.50$

The Dark Knight
17.99$

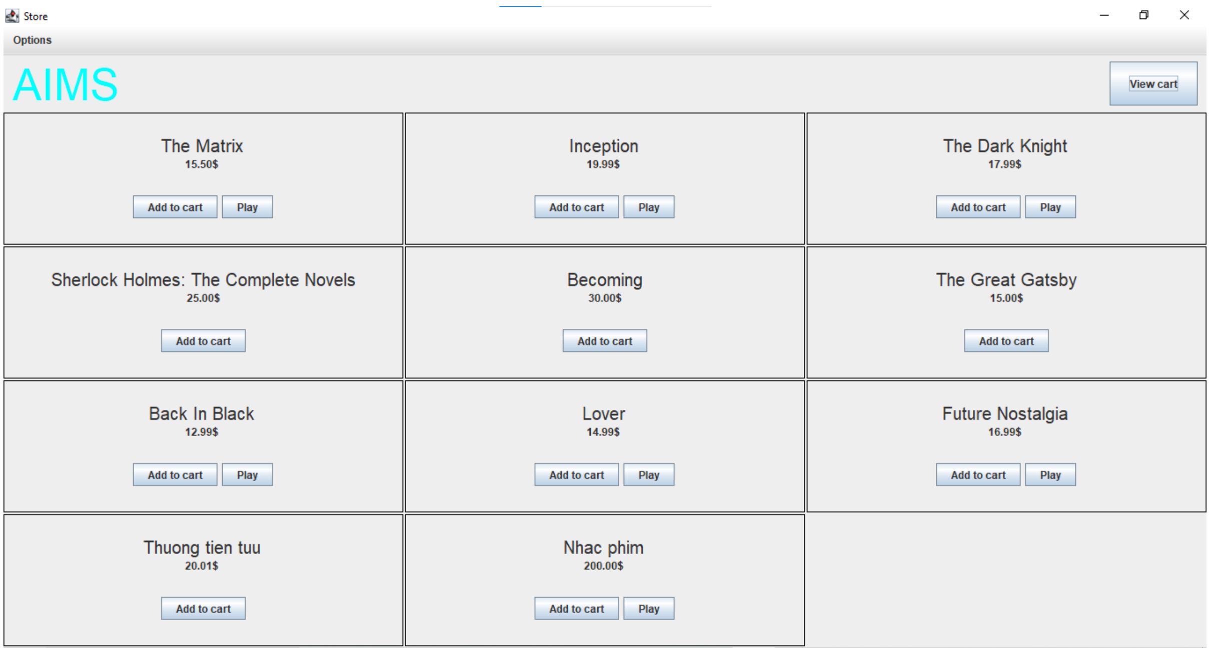
Sherlock Holmes: The Complete Novels
25.00$

The Great Gatsby
15.00$

Back In Black
12.99$

Future Nostalgia
16.99$

Thuong tien tuu
20.01$



The Matrix
15.50$

Inception
19.99$

The Dark Knight
17.99$

Sherlock Holmes: The Complete Novels
25.00$

Becoming
30.00$

The Great Gatsby
15.00$

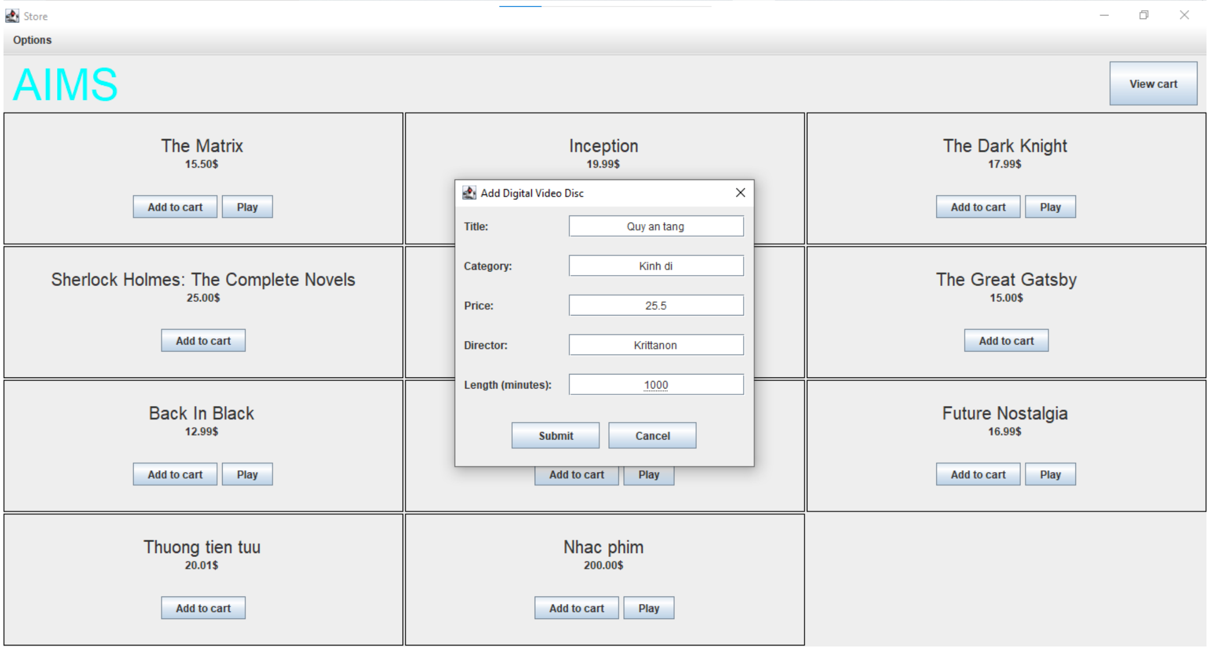
Back In Black
12.99$

Lover
14.99$

Future Nostalgia
16.99$

Thuong tien tuu
20.01$

Nhac phim
200.00$



The Matrix
15.50$

Inception
19.99$

The Dark Knight
17.99$

Sherlock Holmes: The Complete Novels
25.00$

The Great Gatsby
15.00$

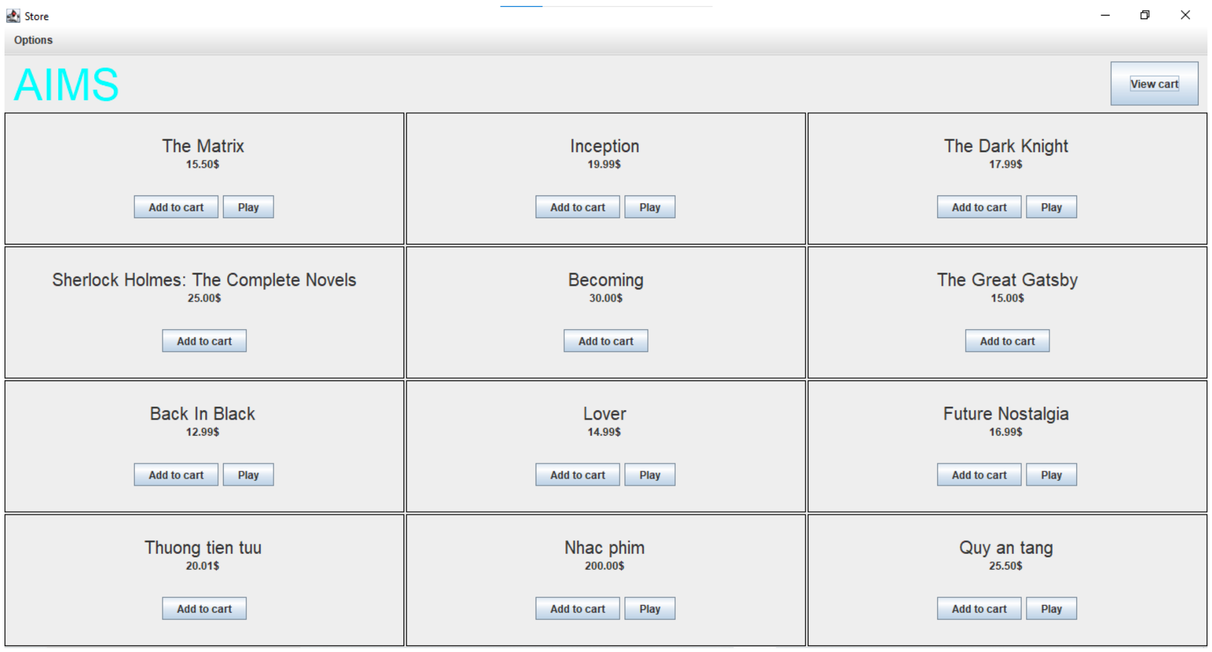
Back In Black
12.99$

Future Nostalgia
16.99$

Thuong tien tuu
20.01$

Nhac phim
200.00$

Title:	Quy an tang
Category:	Kinh di
Price:	25.5
Director:	Krittanon
Length (minutes):	1000



AIMS

The Matrix
15.50$

Inception
19.99$

The Dark Knight
17.99$

Sherlock Holmes: The Complete Novels
25.00$

Becoming
30.00$

The Great Gatsby
15.00$

Back In Black
12.99$

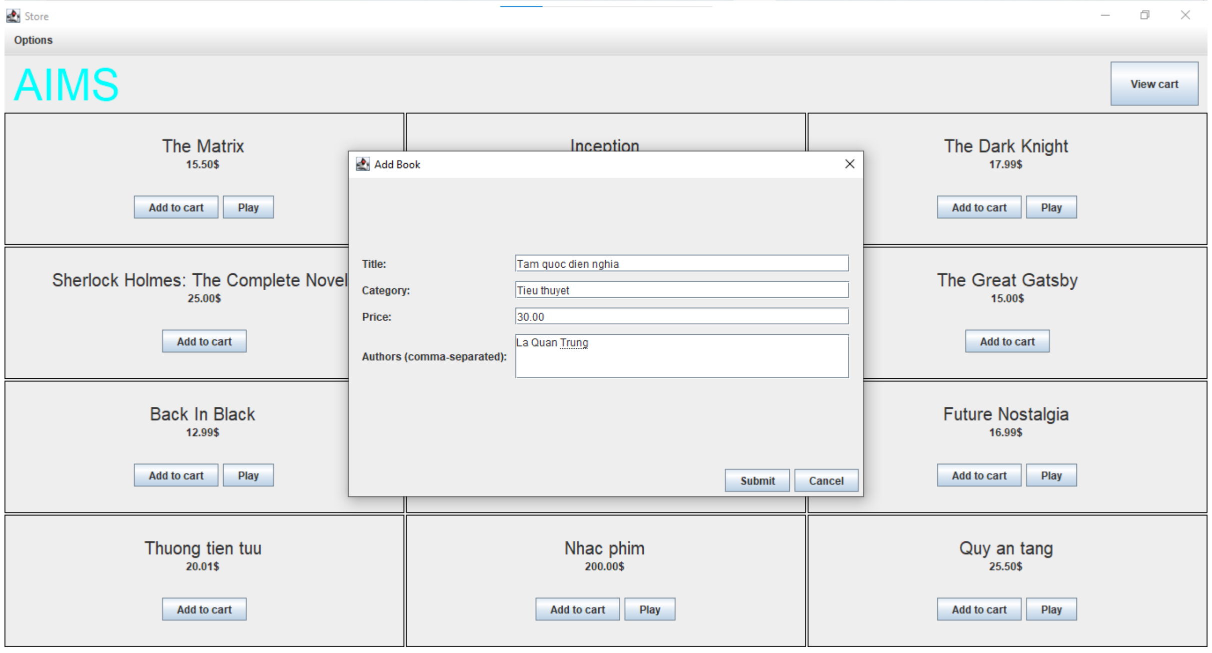
Lover
14.99$

Future Nostalgia
16.99$

Thuong tien tuu
20.01$

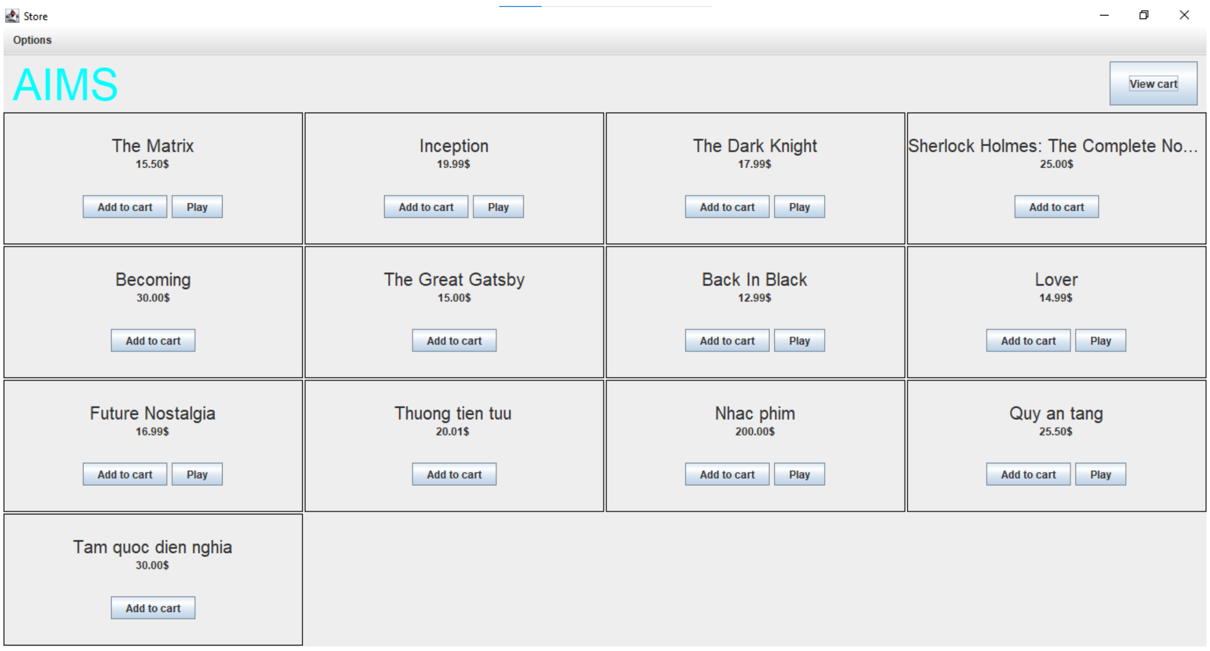
Nhac phim
200.00$

Quy an tang
25.50$

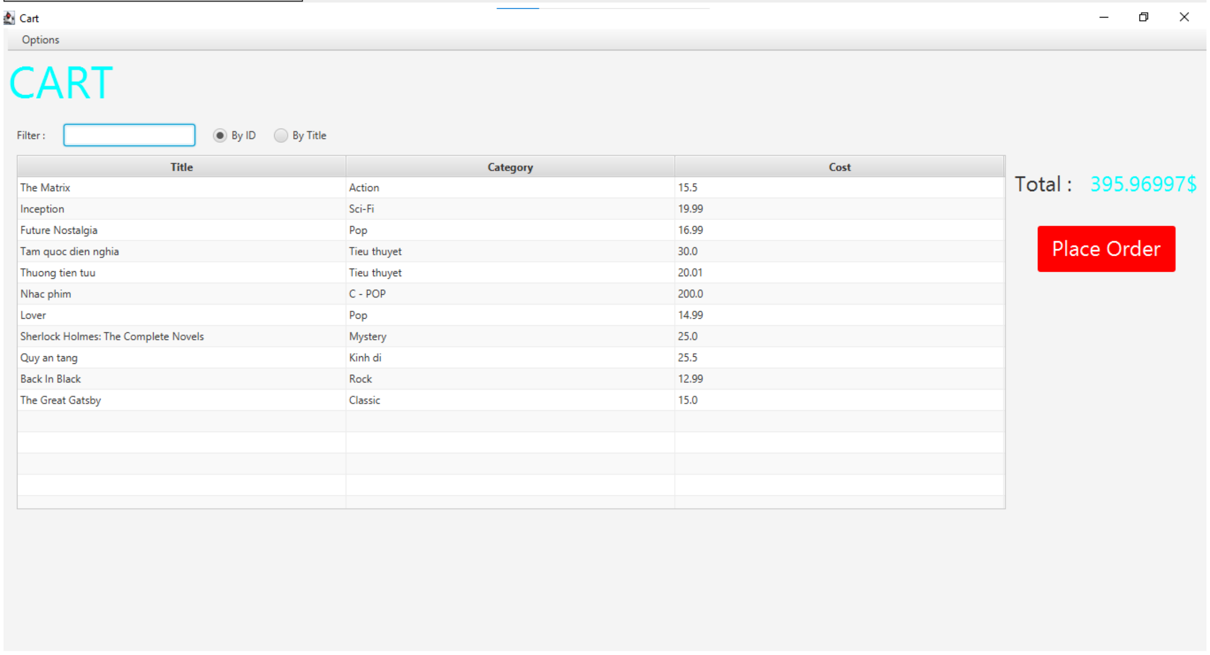


AIMS

Title: Tam quoc dien nghia
Category: Tieu thuyet
Price: 30.00
Authors (comma-separated): La Quan Trung

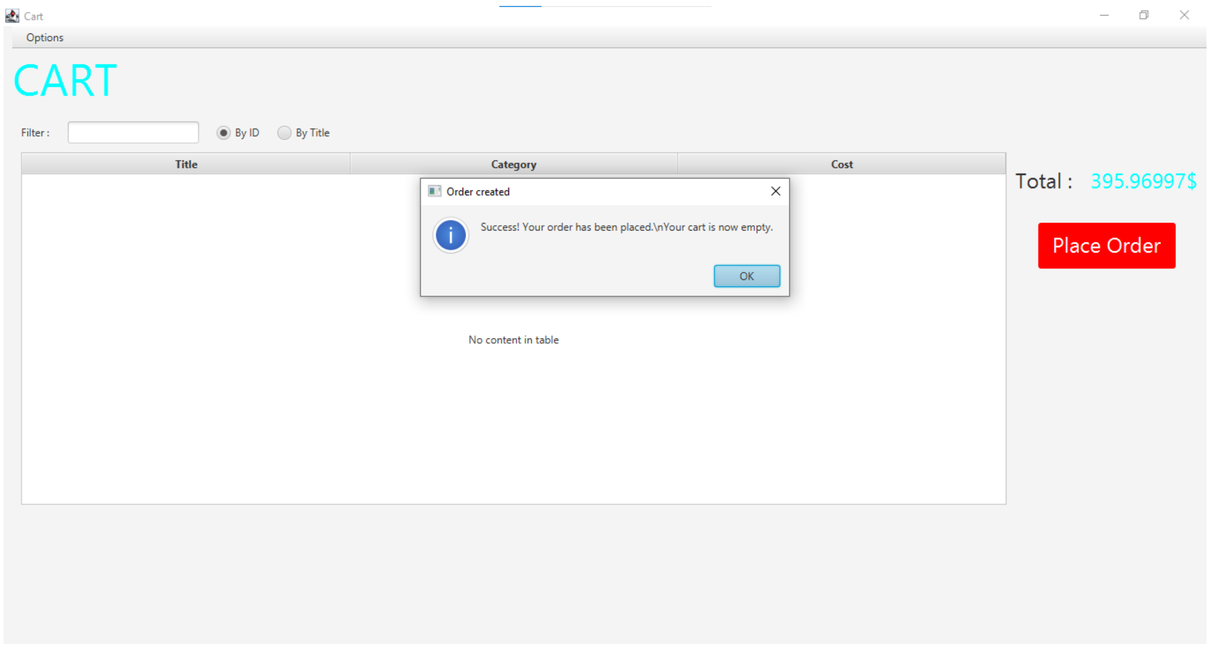


The Matrix 15.50$ Add to cart · Play	Inception 19.99$ Add to cart · Play	The Dark Knight 17.99$ Add to cart · Play	Sherlock Holmes: The Complete No… 25.00$ Add to cart
Becoming 30.00$ Add to cart	The Great Gatsby 15.00$ Add to cart	Back In Black 12.99$ Add to cart · Play	Lover 14.99$ Add to cart · Play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



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$


Cart

Options

CART

Filter : By ID By Title

Title	Category	Cost

No content in table

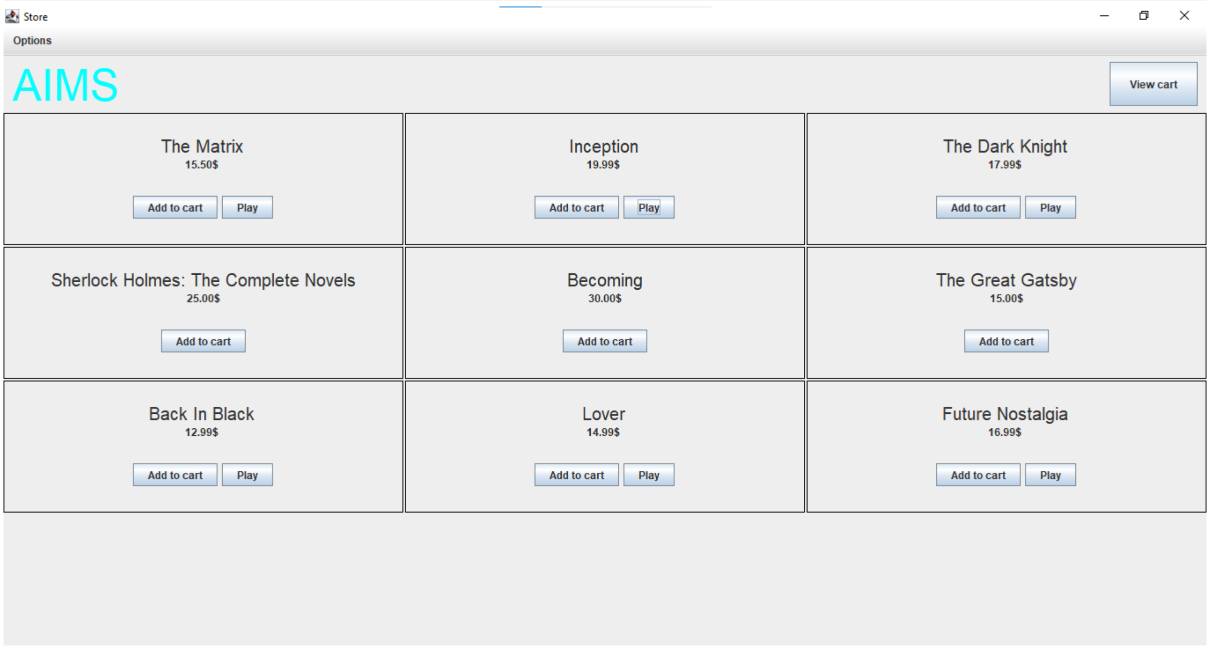
Total : 395.96997$

Place Order

Order created

Success! Your order has been placed.\nYour cart is now empty.

OK

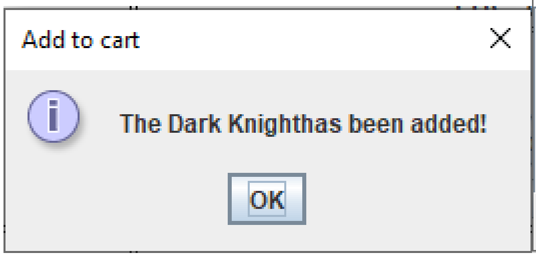

Store

Options

AIMS

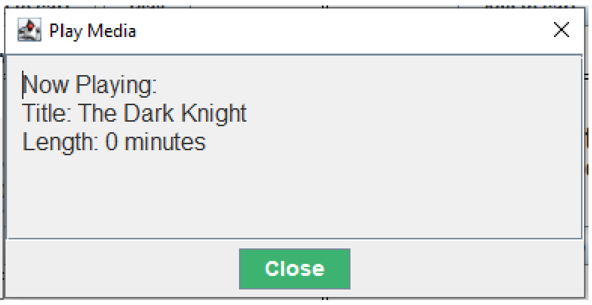
View cart

The Matrix 15.50$ Add to cart Play	Inception 19.99$ Add to cart Play	The Dark Knight 17.99$ Add to cart Play
Sherlock Holmes: The Complete Novels 25.00$ Add to cart	Becoming 30.00$ Add to cart	The Great Gatsby 15.00$ Add to cart
Back In Black 12.99$ Add to cart Play	Lover 14.99$ Add to cart Play	Future Nostalgia 16.99$ Add to cart Play


Add to cart

The Dark Knighthas been added!

OK


Play Media

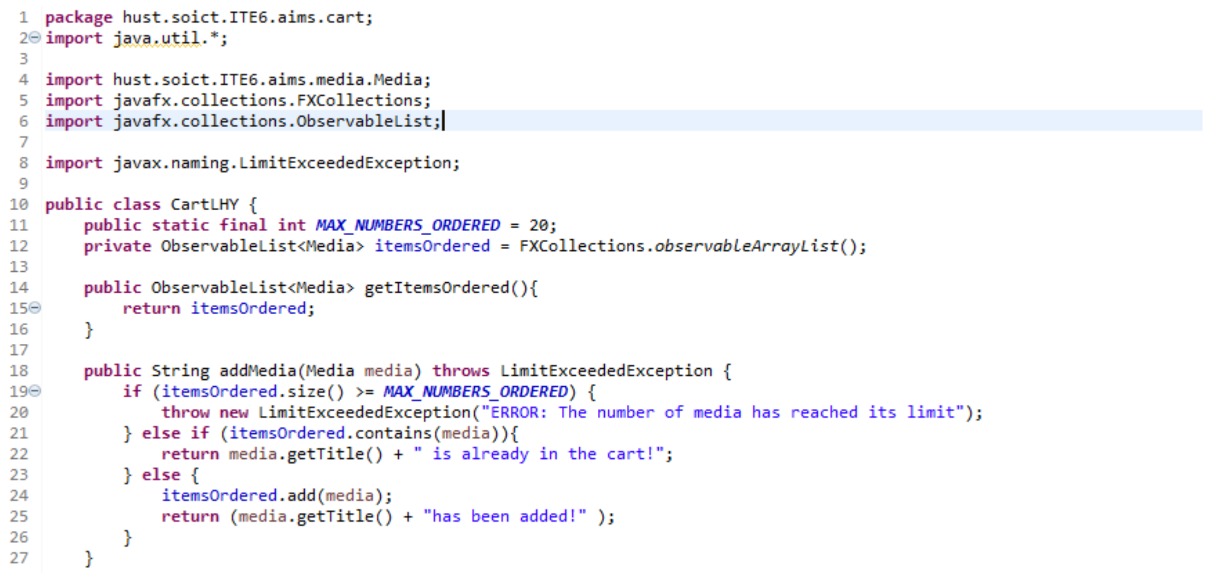
Now Playing:
Title: The Dark Knight
Length: 0 minutes

Close

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

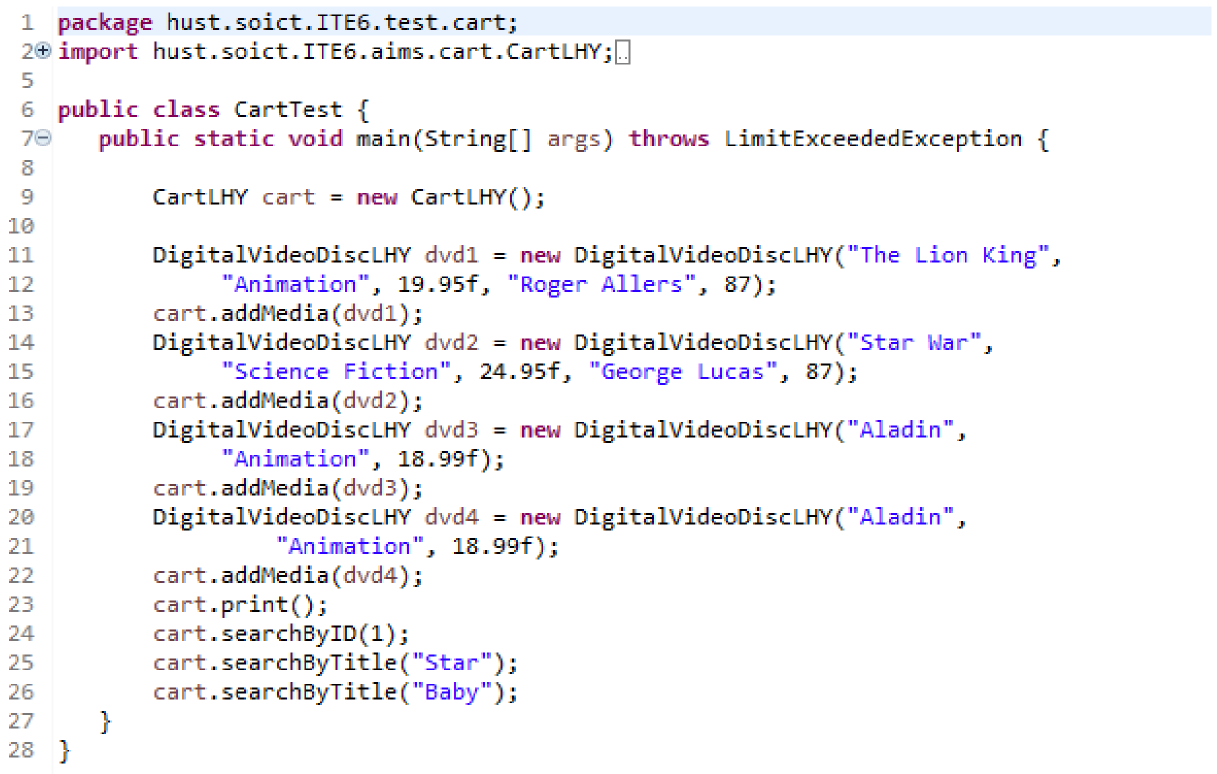
- Branch: topic/aims-project/exceptions

- Cart Class



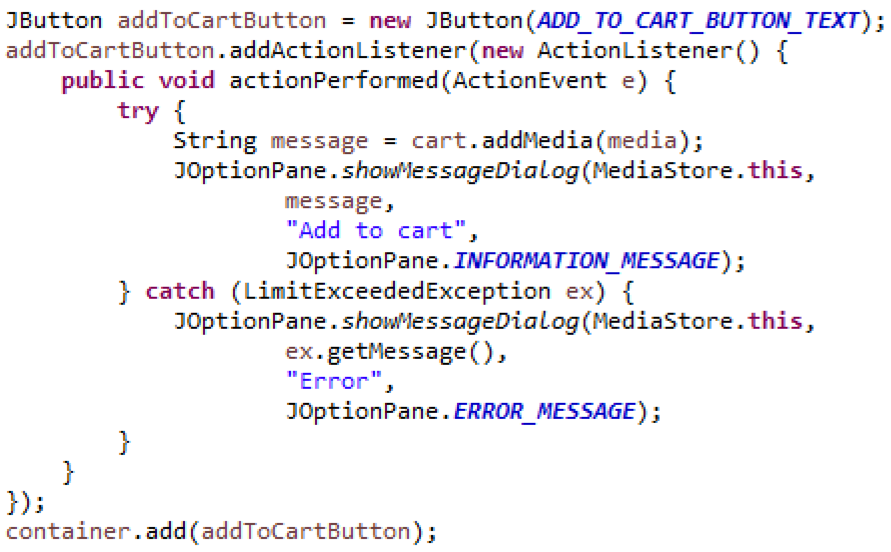
```java
package hust.soict.ITE6.aims.cart;
import java.util.*;

import hust.soict.ITE6.aims.media.Media;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;

import javax.naming.LimitExceededException;

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

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

    public String addMedia(Media media) throws LimitExceededException {
        if (itemsOrdered.size() >= MAX_NUMBERS_ORDERED) {
            throw new LimitExceededException("ERROR: The number of media has reached its limit");
        } else if (itemsOrdered.contains(media)){
            return media.getTitle() + " is already in the cart!";
        } else {
            itemsOrdered.add(media);
            return (media.getTitle() + "has been added!" );
        }
    }
```

- **CartTest**

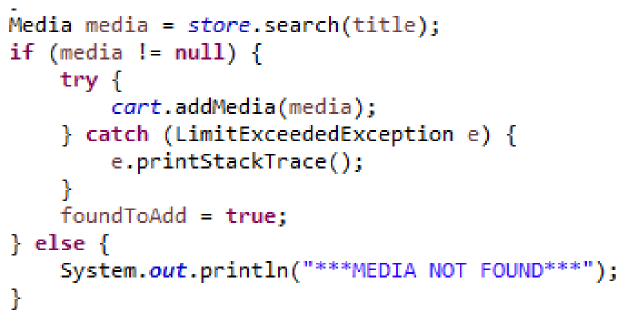


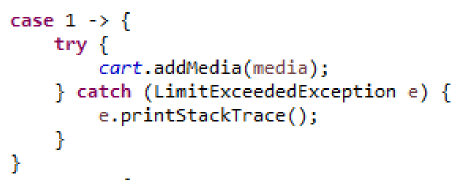
```java
package hust.soict.ITE6.test.cart;
import hust.soict.ITE6.aims.cart.CartLHY;

public class CartTest {
    public static void main(String[] args) throws LimitExceededException {

        CartLHY cart = new CartLHY();

        DigitalVideoDiscLHY dvd1 = new DigitalVideoDiscLHY("The Lion King",
                "Animation", 19.95f, "Roger Allers", 87);
        cart.addMedia(dvd1);
        DigitalVideoDiscLHY dvd2 = new DigitalVideoDiscLHY("Star War",
                "Science Fiction", 24.95f, "George Lucas", 87);
        cart.addMedia(dvd2);
        DigitalVideoDiscLHY dvd3 = new DigitalVideoDiscLHY("Aladin",
                "Animation", 18.99f);
        cart.addMedia(dvd3);
        DigitalVideoDiscLHY dvd4 = new DigitalVideoDiscLHY("Aladin",
                    "Animation", 18.99f);
        cart.addMedia(dvd4);
        cart.print();
        cart.searchByID(1);
        cart.searchByTitle("Star");
        cart.searchByTitle("Baby");
    }
}
```

- **MediaStore**


```java
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**

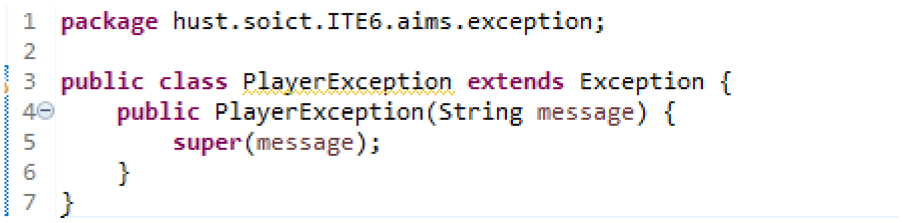

```java
Media media = store.search(title);
if (media != null) {
    try {
        cart.addMedia(media);
    } catch (LimitExceededException e) {
        e.printStackTrace();
    }
    foundToAdd = true;
} else {
    System.out.println("***MEDIA NOT FOUND***");
}
```


```java
case 1 -> {
    try {
        cart.addMedia(media);
    } catch (LimitExceededException e) {
        e.printStackTrace();
    }
}
```

13. Create a class which inherits from Exception

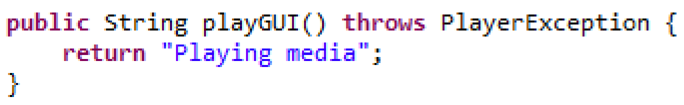
13.1. Create new class named PlayerException

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


```java
package hust.soict.ITE6.aims.exception;

public class PlayerException extends Exception {
    public PlayerException(String message) {
        super(message);
    }
}
```

13.2. Raise the PlayerException in the play() method

- **Media:**


```java
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!");
    }
}
```

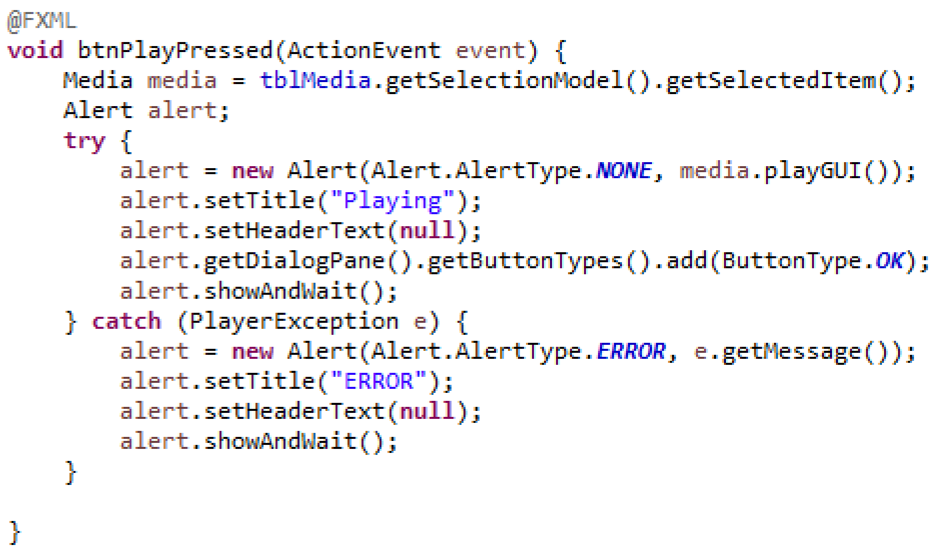
13.3. Update play() in the Playable interface

```
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!");
        }
}
```

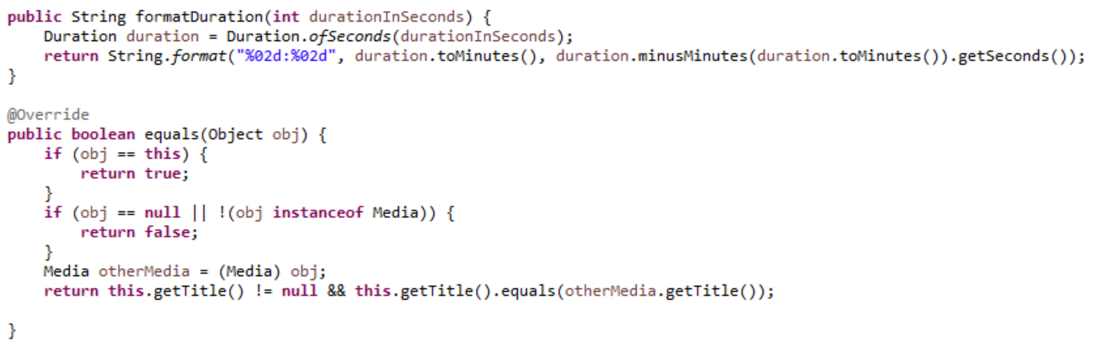
- Update **CartScreenController**



```java
@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.**

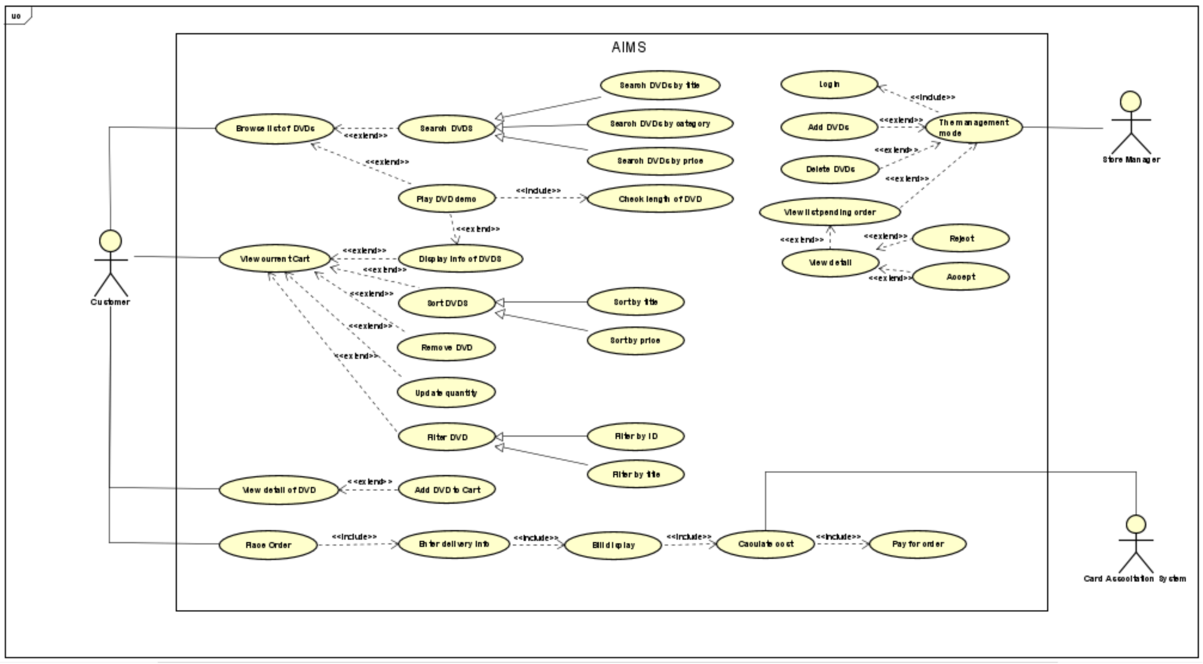
15. Modify the equals() method of Media class
Tránh gặp **NullPointerException** và **ClassCastException**



```java
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)