

Report OOP – Lab 05

Họ và tên: Nguyễn Kiều Duyên

MSSV: 20225618

Mã lớp: 744520 **Kỳ** 2024-1

Table of Contents

I.	New written code	2
1.	Swing Component	2
2.	Tổ chức Swing components with Layout Managers	4
3.	Tạo GUI cho AIMS với Swing	5
4.	API JavaFX	9
5.	View Cart Screen với Scene Builder	12
6.	Tích hợp JavaFX vào Swing – Lớp JFXPanel	12
7.	Hiển thị các mục trong Store – JavaFX's data-driven UI	13
8.	Cập nhật các nút trong TableView – ChangeListener	14
9.	Xóa 1 Media – nút btnRemove	15
10.	Lọc các mục trong giỏ hàng	15
11.	Hoàn thành ứng dụng GUI của Aims	16
12.	Kiểm tra source code trước đó để xử lý Exception	24
13.	Tạo một lớp kế thừa từ Exception	26
14.	Cập nhật lớp Aims	28
15.	Thay đổi method equals() của lớp Media	28
II.	UML Diagram	29
1.	Use-Case Diagram	29
2.	Class Diagram (Final Lab05)	30
3.	Exception Hierarchical	31

I. New written code

1. Swing Component

Branch: `topic/guiproject/swingcomponent_awt`

Tạo thư mục **GUIProject** đặt toàn bộ mã trong package `hust.soict.hedsapi.swing`.

1.1. AWTAccumulator

Tạo **AWTAccumulator** tạo GUI để nhập vào số và tính tổng các số đã nhập.

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

1.2. SwingAccumulator

Tạo lớp **SwingAccumulator** với chức năng tương tự **AWTAccumulator**

```
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     }
```

1.3. So sánh các thành phần Swing và AWT

Lập trình với AWT và Swing khá giống nhau (bao gồm các thành phần/container, xử lý sự kiện).

Tuy nhiên, có một số khác biệt cần lưu ý:

- Container cấp cao nhất trong Swing và AWT.
- Tên lớp của các thành phần trong AWT và tên lớp tương ứng trong Swing.

2. Tổ chức Swing components with Layout Managers

Branch: `topic/guiproject/swingcomponent_layoutmanagers`

Sử dụng **Jpanel** như secondary-level container để tổ chức lại components.

2.1. Tạo lớp `NumberGrid`

```
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);
    }
}
```

2.2. Thêm buttons

```
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);
}
```

2.3. Hoàn thiện lớp bên trong ButtonListener

```
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 delString = tfDisplay.getText();
            if (delString.length() > 0) {
                delString = delString.substring(0, delString.length() - 1);
            }
            tfDisplay.setText(delString);
        }
        else if (button.equals("C")) {
            tfDisplay.setText("");
        }
    }
}
```

3. Tạo GUI cho AIMS với Swing

Đối với ứng dụng AIMS, triển khai ba màn hình:

- Màn hình “**View Store**” sử dụng **Swing**.
- Màn hình “**View Cart**” sử dụng **JavaFX**.
- Màn hình “**Update Store**” sử dụng **JavaFX**.

Branch: topic/aims-project/view-store-screen

- Với **View Store Screen** mã nguồn sẽ được đặt trong package **hust.soict.hedspi.aims.screen**
- Đối với màn hình View Store, chúng ta sẽ sử dụng **BorderLayout**:
- **Component NORTH** sẽ chứa **menu bar** và **header**.
- **Component CENTER** sẽ chứa một **panel** sử dụng **GridLayout**, mỗi ô trong đó là một mặt hàng của cửa hàng.

3.1. Tạo lớp StoreScreen

Đây là **View Store Screen**

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

3.2. Component NORTH

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

- Tạo createMenuBar()

```
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;  
}
```

- Tạo createHeader()

```
JPanel createHeader() {  
    JPanel header = new JPanel();  
    header.setLayout(new BorderLayout(header, BorderLayout.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;  
}
```

3.3. Componet CENTER

```
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`

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

3.4. Lớp `MediaStore`

```
public class MediaStore extends JPanel {  
    private Media media;  
    public MediaStore(Media media) {  
  
        this.media = media;  
        this.setLayout(new BorderLayout(this, BorderLayout.Y_AXIS));  
  
        JLabel title = new JLabel(media.getTitle());  
        title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 20));  
        title.setAlignmentX(CENTER_ALIGNMENT);  
  
        JLabel cost = new JLabel(""+media.getCost()+"$");  
        cost.setAlignmentX(CENTER_ALIGNMENT);  
  
        JPanel container = new JPanel();  
        container.setLayout(new FlowLayout(FlowLayout.CENTER));  
  
        container.add(new JButton("Add to cart"));  
        if(media instanceof Playable) {  
            container.add(new JButton("Play"));  
        }  
  
        this.add(Box.createVerticalGlue());  
        this.add(title);  
        this.add(cost);  
        this.add(Box.createVerticalGlue());  
        this.add(container);  
  
        this.setBorder(BorderFactory.createLineBorder(Color.BLACK));  
    }  
}
```

3.5. Kết hợp tất cả

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

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

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

3.6. Thêm tương tác người dùng vào các nút trên Media

```
// Thêm tương tác cho nút Add to Cart
JButton addToCartButton = new JButton("Add to cart");
addToCartButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        JOptionPane.showMessageDialog(null, media.getTitle() + "added to cart");
    }
});
container.add(addToCartButton);

// Thêm tương tác cho nút Play
if(media instanceof Playable) {
    JButton playButton = new JButton("Play");
    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);
}
```

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

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


4. API JavaFX

Branch: `topic/guiproject/api-javafx`

4.1. Tạo GUI bằng SceneBuilder

- Tại folder **GUIProject**
- Tạo file **Painter.fxml** trong `hust.soict.hedspi.javafx` và mở bằng SceneBuilder



4.2. Tạo lớp PainterController

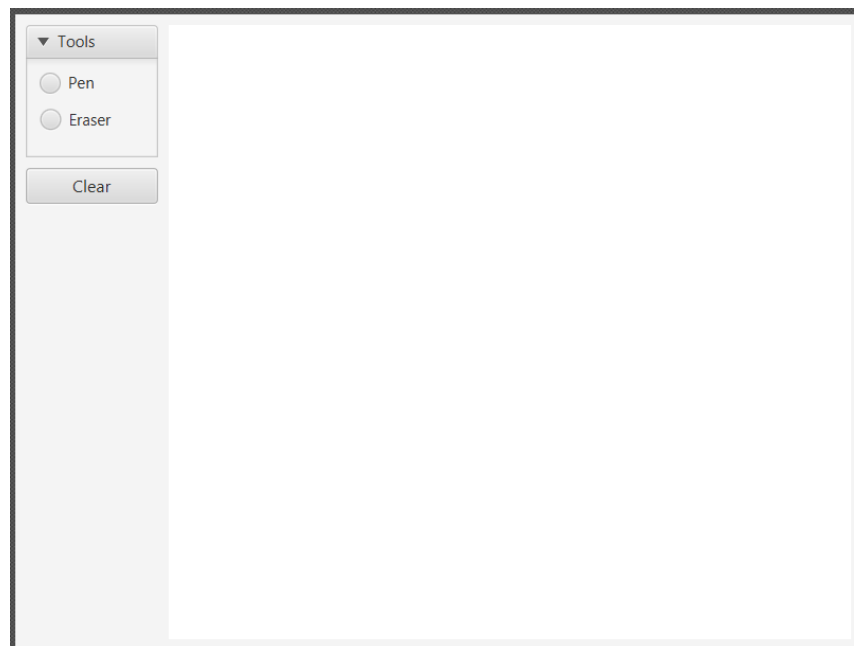
```
public class PainterController {  
  
    @FXML  
    private Pane drawingAreaPane;  
  
    @FXML  
    void clearButtonPressed(ActionEvent event) {  
        drawingAreaPane.getChildren().clear();  
    }  
  
    @FXML  
    void drawingAreaMouseDragged(MouseEvent event) {  
        Circle newCircle = new Circle(event.getX(),  
                                       event.getY(), 4, Color.BLACK);  
        drawingAreaPane.getChildren().add(newCircle);  
    }  
}
```

4.3. Tạo lớp Painter để chạy Application

```
public class Painter extends Application {  
  
    @Override  
    public void start(Stage stage) throws Exception {  
        Parent root = FXMLLoader.load(getClass()  
            .getResource("/hust/soict/hedspi/javafx/Painter.fxml"));  
  
        Scene scene = new Scene(root);  
        stage.setTitle("Painter");  
        stage.setScene(scene);  
        stage.show();  
    }  
    public static void main(String[] args) {  
        launch(args);  
    }  
}
```

4.4. Thêm chức năng Eraser

- Thêm **TitledPane** và 2 **RadioButtons** tương ứng với pen và eraser.
- Đặt thuộc tính **Toggle Group** cho cả 2 **RadioButtons** giống nhau để chỉ 1 trong số chúng được chọn tại 1 thời điểm
- Màn hình **Painter** sau khi thêm, file **Painter.fxml**



- Chỉnh sửa lại **PainterController**

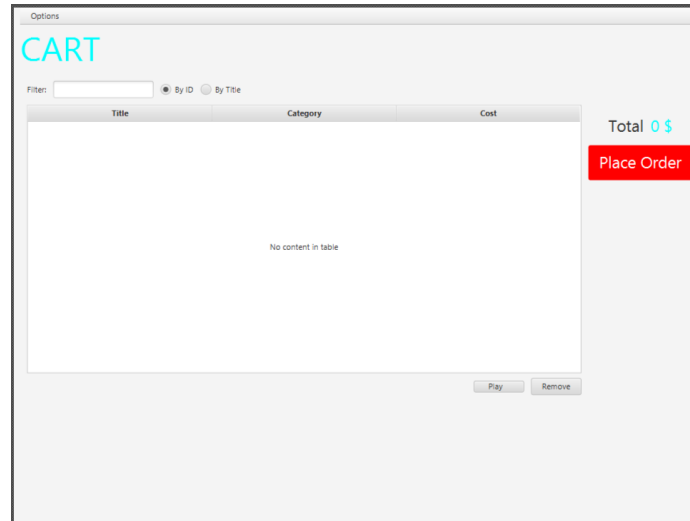
```
13 public class PainterController {
14
15     @FXML
16     private RadioButton eraser;
17
18     @FXML
19     private RadioButton pen;
20
21     @FXML
22     private Pane drawingAreaPane;
23
24     // add a ToggleGroup to group RadioButtons
25     private ToggleGroup toggleGroup;
26
27     @FXML
28     void clearButtonPressed(ActionEvent event) {
29         drawingAreaPane.getChildren().clear();
30     }
31
32     @FXML
33     void drawingAreaMouseDragged(MouseEvent event) {
34         Rectangle clipArea = new Rectangle(0, 0, drawingAreaPane.getWidth(), drawingAreaPane.getHeight());
35         drawingAreaPane.setClip(clipArea);
36         Color inkColor = Color.BLACK;
37         if (eraser.isSelected()) {
38             inkColor = Color.WHITE;
39         }
40         Circle newCircle = new Circle(event.getX(), event.getY(), 4, inkColor);
41         drawingAreaPane.getChildren().add(newCircle);
42     }
43
44     @FXML
45     void initialize() {
46         toggleGroup = new ToggleGroup();
47         pen.setToggleGroup(toggleGroup);
48         eraser.setToggleGroup(toggleGroup);
49
50         pen.setSelected(true);
51     }
52 }
```

Branch: topic/aims-project/view-cart-screen (Từ 5-11)

5. View Cart Screen với Scene Builder

Thực hành trong package `hust.soict.hedspi.aims.screen` với các bài tiếp theo.

- Tạo `cart.fxml` trong `hust.soict.hedspi.aims.screen.view`



6. Tích hợp JavaFX vào Swing – Lớp JFXPanel

- Tạo lớp `CartScreen`

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

    public CartScreen(Cart cart) {
        super();

        this.cart = cart;

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

        this.setTitle("Cart");
        this.setSize(1024, 768);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setVisible(true);
        Platform.runLater(new Runnable() {
            @Override
            public void run() {
                try {
                    FXMLLoader loader = new FXMLLoader(getClass()
                        .getResource("/hust/soict/hedspi/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. Hiển thị các mục trong Store – JavaFX's data-driven UI

- Tạo **CartScreenController** trong **hust.soict.hedspi.aims.screen.controller**

```
public class CartScreenController {
    private Cart cart;

    @FXML
    private TableColumn<Media, Float> colMediaCost;

    @FXML
    private TableColumn<Media, String> colMediaTitle;

    @FXML
    private TableColumn<Media, String> colMediaCategory;

    @FXML
    private TableView<Media> tblMedia;

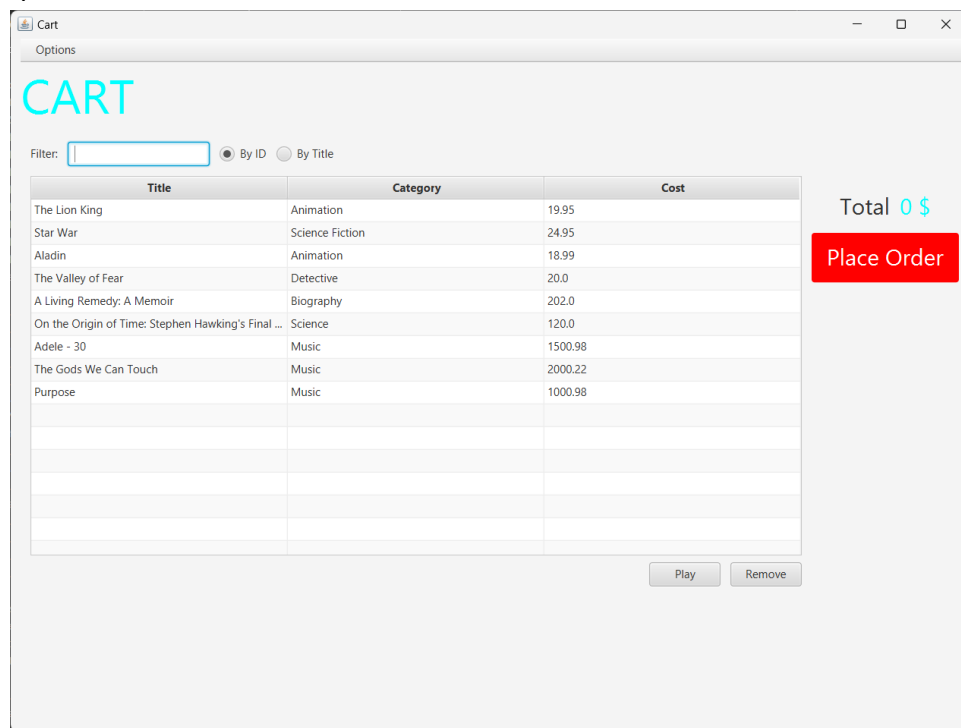
    public CartScreenController(Cart cart) {
        super();
        this.cart = cart;
    }

    @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());
    }
}
```

- Thay đổi thuộc tính **itemsOrdered** từ **List<Media>** thành **ObservableList<Media>** và thêm method **getItemsOrdered()** trong lớp **Cart**

```
private ObservableList<Media> itemsOrdered = FXCollections.observableArrayList();
public ObservableList<Media> getItemsOrdered() {
    return itemsOrdered;
}
```

- Kết quả màn hình **CartScreen**



8. Cập nhật các nút trong TableView – ChangeListener

- Cập nhật nút **btnPlay** và **btnRemove** trong **CartScreenController**

```
@FXML
private Button btnPlay;

@FXML
private Button btnRemove;
```

```
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. Xóa 1 Media – nút btnRemove

- Triển khai sự kiện cho nút Remove

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

- Đồng thời thêm Action trong **Cart.fxml**

10. Lọc các mục trong giỏ hàng

TextField nơi người dùng nhập chuỗi lọc: **tfFilter**

Hai nút radio để xác định tiêu chí lọc:

- Nút “By ID”: **radioBtnFilterId**
- Nút “By Title”: **radioBtnFilterTitle**

```
@FXML
private TextField tfFilter;

@FXML
private RadioButton radioBtnFilterId;

@FXML
private RadioButton radioBtnFilterTitle;
```

Thêm **ChangeListener** cho **tfFilter** trong **initialize()**

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

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

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

            if(!keyword.isEmpty() && radioBtnFilterId.isSelected()) {
                filteredList.setPredicate(media -> {
                    String idString = String.valueOf(media.getId());
                    return idString.equals(keyword);
                });
            } else if(!keyword.isEmpty() && radioBtnFilterTitle.isSelected()) {
                filteredList.setPredicate(media -> {
                    String title = media.getTitle().toLowerCase();
                    return title.contains(keyword.toLowerCase());
                });
            } else {
                filteredList.setPredicate(null);
            }
            tblMedia.setItems(filteredList);
        }
    });
```

11. Hoàn thành ứng dụng GUI của Aims

11.1. Hoàn thành Cart Screen

- Nút Place order `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();
}
```

- Thêm method **placeOrder** trong lớp **Cart**

```
public String placeOrder() {
    if(itemsOrdered.size()==0) {
        return "Your cart is empty!";
    } else {
        qtyOrdered = 0;
        itemsOrdered.clear();
        return "Order created!\n" + "Now your cart will be empty!";
    }
}
```

- Nút 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();
}
```

- Cập nhật Cost vào thao tác xóa Media trong cart

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

Branch: topic/aims-project/update-store-screen

- Kết nối **StoreScreen** với **CartScreen**

- Thêm new Cart vào lớp **StoreScreen**
- Xóa **Cart**, **initsetup** ở lớp **CartScreen.java**

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


- Mỗi lần ấn nút **Add to cart** thì add Media vào cart vừa tạo (**MediaStore.java**)

```
JButton addToCartButton = new JButton("Add to cart");
addToCartButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        JOptionPane.showMessageDialog(null, cart.addMedia(media));
    }
});
container.add(addToCartButton);
```

- Sửa lại dạng trả về của method **addMedia()** trong **Cart.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 (media.getTitle() + "has been added!");
    }
}
```

- Thêm action **View Cart** thì hiện ra **CartScreen** với những cart đã thêm

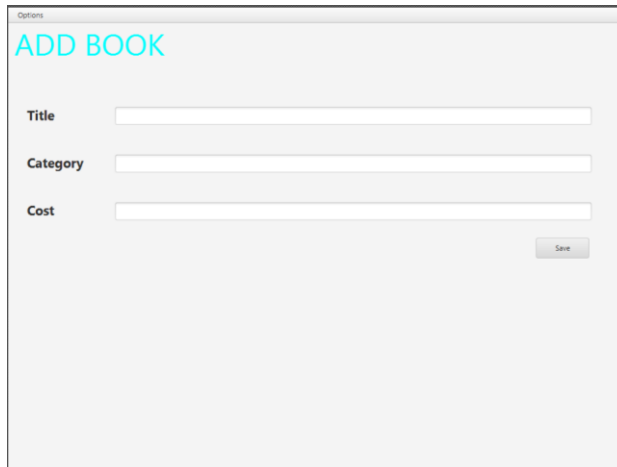
```
JButton cartBtn = new JButton("View cart");
cartBtn.setPreferredSize(new Dimension(100, 50));
cartBtn.setMaximumSize(new Dimension(100, 50));
cartBtn.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        new CartScreen(cart);
    }
});
```

11.2. Update Store Screen

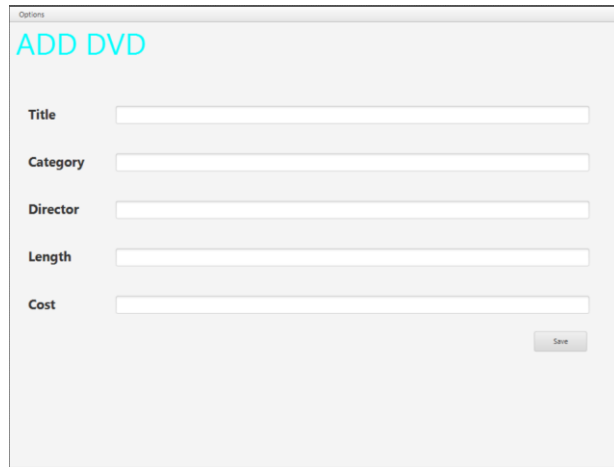
Tạo 3 lớp

- **AddDigitalVideoDiscToStoreScreen**
- **AddBookToStoreScreen**
- **AddCompactDiscToStoreScreen**
- **AddTrackScreen**

- Trong `hust.soict.hedspi.aims.screen.view`



addBook.fxml



addDVD.fxml



addCD.fxml



addTracks.fxml

- Trong `hust.soict.hedspi.aims.screen.controller`

- **AddBookScreenController.java**

```
public class AddBookScreenController {  
    private Store store;  
  
    @FXML  
    private ResourceBundle resources;  
  
    @FXML  
    private URL location;  
  
    @FXML  
    private Button btnSave;  
  
    @FXML  
    private TextField tfCategory;  
  
    @FXML  
    private TextField tfCost;  
  
    @FXML  
    private TextField tfTitle;
```

```
    private boolean allFieldsFilled = false;  
  
    public AddBookScreenController(Store store) {  
        super();  
        this.store = store;  
    }
```

```

@FXML
void btnSavePressed(ActionEvent event) {
    String title = tfTitle.getText();
    String category = tfCategory.getText();
    float cost = 0.0f;
    try {
        cost = Float.parseFloat(tfCost.getText());
    } catch (NumberFormatException e) {
        Alert alert = new Alert(Alert.AlertType.ERROR, "Failed to parse cost!");
        alert.setTitle("Wrong type");
        alert.setHeaderText(null);
        alert.showAndWait();
        return;
    }
    Book book = new Book(title, category, cost);
    store.addMedia(book);
    tfTitle.clear();
    tfCategory.clear();
    tfCost.clear();
    Alert alert = new Alert(Alert.AlertType.INFORMATION, "Book has been added to the store!");
    alert.setTitle("Success");
    alert.setHeaderText(null);
    alert.showAndWait();
}

@FXML
void initialize() {
    btnSave.setDisable(true);

    tfTitle.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
    tfCategory.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
    tfCost.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
}

private void checkFieldsFilled() {
    if (!tfTitle.getText().isEmpty() && !tfCategory.getText().isEmpty() && !tfCost.getText().isEmpty()) {
        allFieldsFilled = true;
    } else {
        allFieldsFilled = false;
    }
    btnSave.setDisable(!allFieldsFilled);
}

```

- AddDVDScreenController.java

```

public class AddDVDScreenController {

    private Store store;

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;

    @FXML
    private Button btnSave;

    @FXML
    private TextField tfCategory;

    @FXML
    private TextField tfCost;

    @FXML
    private TextField tfDirector;

```

```

@FXML
private TextField tfLength;

@FXML
private TextField tfTitle;

private boolean allFieldsFilled = false;

public AddDVDScreenController(Store store) {
    super();
    this.store = store;
}

```

```

@FXML
void btnSavePressed(ActionEvent event) {
    String title = tfTitle.getText();
    String category = tfCategory.getText();
    String director = tfDirector.getText();
    int length = 0;
    try {
        length = Integer.parseInt(tfLength.getText());
    } catch (Exception e) {
        Alert alert = new Alert(Alert.AlertType.ERROR, "Failed to parse length!");
        alert.setTitle("Wrong type");
        alert.setHeaderText(null);
        alert.showAndWait();
        return;
    }
    float cost = 0.0f;
    try {
        cost = Float.parseFloat(tfCost.getText());
    } catch (NumberFormatException e) {
        Alert alert = new Alert(Alert.AlertType.ERROR, "Failed to parse cost!");
        alert.setTitle("Wrong type");
        alert.setHeaderText(null);
        alert.showAndWait();
        return;
    }
    DigitalVideoDisc DVD = new DigitalVideoDisc(title, category, director, length, cost);
    store.addMedia(DVD);
    tfTitle.clear();
    tfCategory.clear();
    tfDirector.clear();
    tfLength.clear();
    tfCost.clear();
    Alert alert = new Alert(Alert.AlertType.INFORMATION, "DVD has been added to the store!");
    alert.setTitle("Success");
    alert.setHeaderText(null);
    alert.showAndWait();
}

```

```

@FXML
void initialize() {
    btnSave.setDisable(true);

    tfTitle.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
    tfCategory.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
    tfDirector.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
    tfLength.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
    tfCost.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
}

private void checkFieldsFilled() {
    if (!tfTitle.getText().isEmpty() && !tfCategory.getText().isEmpty() && !tfDirector.getText().isEmpty()
        && !tfLength.getText().isEmpty() && !tfCost.getText().isEmpty()) {
        allFieldsFilled = true;
    } else {
        allFieldsFilled = false;
    }
    btnSave.setDisable(!allFieldsFilled);
}

```

- AddCDScreenController.java

```
public class AddCDScreenController {

    private Store store;

    private CompactDisc CD;

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;

    @FXML
    private Button btnAddTrack;

    @FXML
    private Button btnSave;

    @FXML
    private Button btnAddCD;

    @FXML
    private TextField tfCategory;
```

```
@FXML
private TextField tfCost;

@FXML
private TextField tfArtist;

@FXML
private TextField tfTitle;

private boolean allFieldsFilled = false;

public AddCDScreenController(Store store) {
    super();
    this.store = store;
}
```

```
@FXML
void btnAddCDPressed(ActionEvent event) {
    store.addMedia(CD);
    tfTitle.clear();
    tfCategory.clear();
    tfArtist.clear();
    tfCost.clear();
    btnSave.setDisable(true);
    btnAddCD.setDisable(true);
    btnAddTrack.setDisable(true);
    Alert alert = new Alert(Alert.AlertType.INFORMATION, "CD has been added to the store!");
    alert.setTitle("Success");
    alert.setHeaderText(null);
    alert.showAndWait();
}

@FXML
void btnAddTrackPressed(ActionEvent event) {
    new AddTrack(CD);
}

@FXML
void btnSavePressed(ActionEvent event) {
    String title = tfTitle.getText();
    String category = tfCategory.getText();
    String artist = tfArtist.getText();
    float cost = 0.0f;
    try {
        cost = Float.parseFloat(tfCost.getText());
    } catch (NumberFormatException e) {
        Alert alert = new Alert(Alert.AlertType.ERROR, "Failed to parse cost!");
        alert.setTitle("Wrong type");
        alert.setHeaderText(null);
        alert.showAndWait();
        return;
    }
    CD = new CompactDisc(title, category, artist, cost);
    btnAddCD.setDisable(false);
    btnAddTrack.setDisable(false);
    btnSave.setDisable(true);
}
```

```

@FXML
void initialize() {
    btnSave.setDisable(true);
    btnAddCD.setDisable(true);
    btnAddTrack.setDisable(true);

    tfTitle.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
    tfCategory.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
    tfArtist.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
    tfCost.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
}

private void checkFieldsFilled() {
    if (!tfTitle.getText().isEmpty() && !tfCategory.getText().isEmpty()
        && !tfArtist.getText().isEmpty() && !tfCost.getText().isEmpty()) {
        allFieldsFilled = true;
    } else {
        allFieldsFilled = false;
    }
    btnSave.setDisable(!allFieldsFilled);
}

```

- AddTrackScreenController.java

```

public class AddTrackScreenController {
    private CompactDisc CD;

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;

    @FXML
    private Button btnSaveTrack;

    @FXML
    private TextField tfLength;

    @FXML
    private TextField tfTitle;

    private boolean allFieldsFilled = false;

    public AddTrackScreenController(CompactDisc CD) {
        super();
        this.CD = CD;
    }
}

```

```

@FXML
void btnSaveTrackPressed(ActionEvent event) {
    String title = tfTitle.getText();
    int length = 0;
    try {
        length = Integer.parseInt(tfLength.getText());
    } catch (Exception e) {
        Alert alert = new Alert(Alert.AlertType.ERROR, "Failed to parse length!");
        alert.setTitle("Wrong type");
        alert.setHeaderText(null);
        alert.showAndWait();
        return;
    }
    Track track = new Track(title, length);
    CD.addTrack(track);
    tfTitle.clear();
    tfLength.clear();
    Alert alert = new Alert(Alert.AlertType.INFORMATION, "Track has been added!");
    alert.setTitle("Success");
    alert.setHeaderText(null);
    alert.showAndWait();
}

@FXML
void initialize() {
    btnSaveTrack.setDisable(true);

    tfTitle.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
    tfLength.textProperty().addListener((observable, oldValue, newValue) -> checkFieldsFilled());
}

private void checkFieldsFilled() {
    if (!tfTitle.getText().isEmpty() && !tfLength.getText().isEmpty()) {
        allFieldsFilled = true;
    } else {
        allFieldsFilled = false;
    }
    btnSaveTrack.setDisable(!allFieldsFilled);
}

```

- Trong `hust.soict.hedspi.aims.screen`

Tạo ra 4 class

- **AddBookToStoreScreen.java**
- **AddDigitalVideoDiscToStoreScreen.java**
- **AddCompactDiscToStoreScreen.java**
- **AddTrack.java**

4 Class này để có code giống nhau nên dưới đây chỉ minh họa 1 lớp là

AddDigitalVideoDiscToStoreScreen.java

```

public class AddDigitalVideoDiscToStoreScreen extends JFrame{
    private static Store store;

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

    public AddDigitalVideoDiscToStoreScreen(Store store) {
        super();

        AddDigitalVideoDiscToStoreScreen.store = store;

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

        this.setTitle("Add DVD");
        this.setSize(1024, 768);
        this.setVisible(true);
        Platform.runLater(new Runnable() {
            @Override
            public void run() {
                try {
                    FXMLLoader loader = new FXMLLoader(getClass().getResource("/hust/soict/hedspi/aims/screen/view/addDVD.fxml"));

                    AddDVDScreenController controller = new AddDVDScreenController(store);
                    loader.setController(controller);
                    Parent root = loader.load();
                    fxPanel.setScene(new Scene(root));
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

Branch: topic/aims-project/update-store-screen (12,13,14,15)

12. Kiểm tra source code trước đó để xử lý Exception

- Xem xét tất cả các method, class trong lớp **AimsProject** để catch/handle tất cả những exception cần thiết.
- Throw Exception cho method **addMedia()** trong **Cart.java** và **testCart.java**

```

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!");
    }
}

```



```

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

        Cart cart = new Cart();

        DigitalVideoDisc dvd1 = new DigitalVideoDisc("The Lion King",
            "Animation", "Roger Allers", 87, 19.95f);
        cart.addMedia(dvd1);

        DigitalVideoDisc dvd2 = new DigitalVideoDisc("Star War",
            "Science Fiction", "George Lucas", 87, 24.95f);
        cart.addMedia(dvd2);

        DigitalVideoDisc dvd3 = new DigitalVideoDisc("Aladin",
            "Animation", 18.99f);
        cart.addMedia(dvd3);

        DigitalVideoDisc dvd4 = new DigitalVideoDisc("Aladin",
            "Animation", 18.99f);
        cart.addMedia(dvd4);

        cart.print();

        cart.searchByID(3);
        cart.searchByTitle("Lion");
    }
}

```

- Đồng thời catch Exception trong **MediaStore.java** và **Aims.java**

```

// Thêm tương tác cho nút Add to cart
JButton addToCartButton = new JButton("Add to cart");
addToCartButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        try {
            String message = cart.addMedia(media);
            JOptionPane.showMessageDialog(null, message);
        } catch (LimitExceededException ex) {
            JOptionPane.showMessageDialog(null, ex.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
        }
    }
});
container.add(addToCartButton);

```

```

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***");
}

```

```

case 1:
    try {
        cart.addMedia(media);
    } catch (LimitExceededException e) {
        e.printStackTrace();
    }
    break;

```

13. Tạo một lớp kế thừa từ Exception

13.1. Tạo 1 lớp mới có tên là PlayerException

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

```
3 public class PlayerException extends Exception {
4     public PlayerException(String message) {
5         super(message);
6     }
7 }
```

13.2. Raise ngoại lệ PlayerException vào method play() (Bài này dùng method playGUI())

- Thêm vào **Media**

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

- Thêm vào **DigitalVideoDisc**

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

- Thêm vào **Track**

```
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. Cập nhật play() trong interface Playable

- Không cập nhật vì ở đây dùng exception cho **playGUI()**

13.4. Cập nhật play() trong lớp 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!");
    }
}
```

- Cập nhật các lớp khác

- **MediaStore**

```
// Thêm tương tác cho nút Play
if (media instanceof Playable) {
    JButton playButton = new JButton("Play");
    playButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {

            JDialog dialog = new JDialog();
            dialog.setTitle(media.getTitle());
            dialog.setSize(400, 300);

            String mediaInfo = "";
            try {
                mediaInfo = "<html>" + media.playGUI().replace("\n", "<br/>") + "</html>";
                JLabel mediaLabel = new JLabel(mediaInfo);
                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);
            } catch (PlayerException e1) {
                JOptionPane.showMessageDialog(null, e1.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
            }

        }
    });
    container.add(playButton);
}
```

- 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. Cập nhật lớp Aims

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

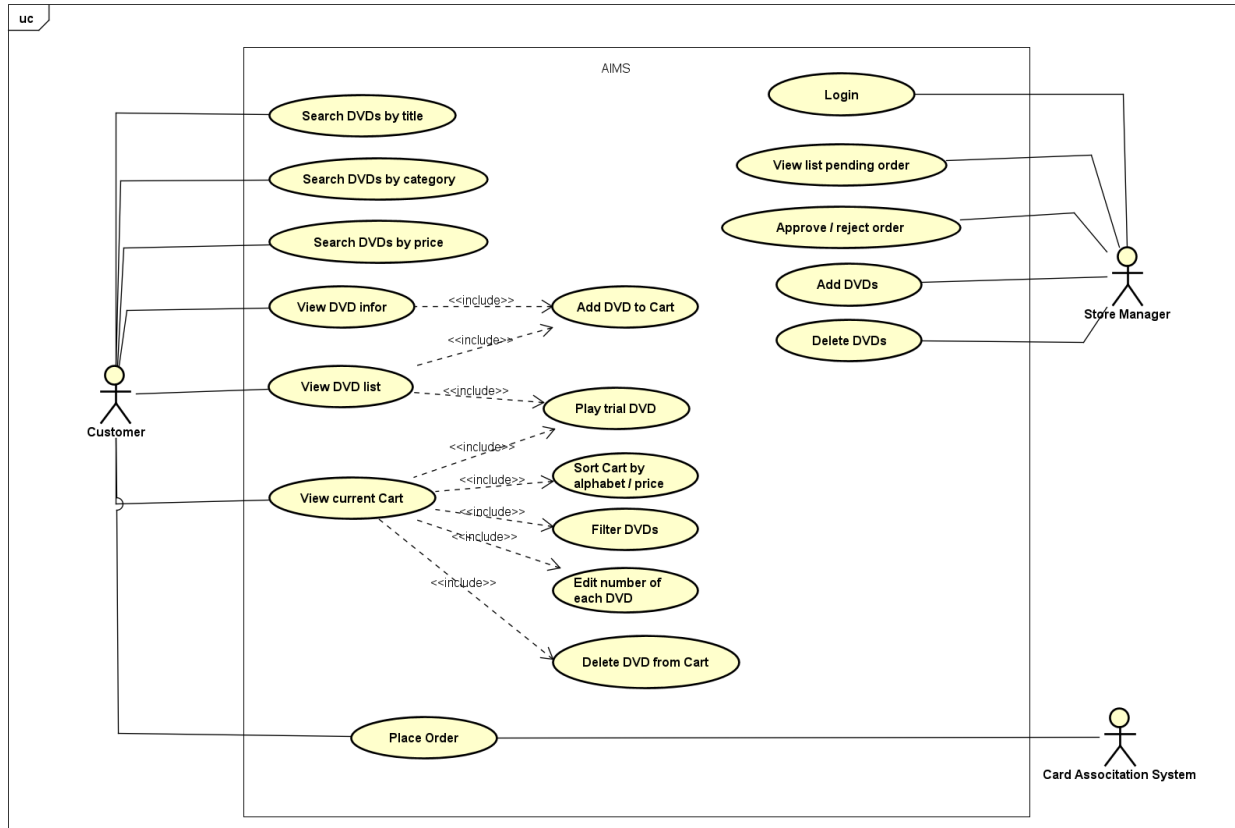
15. Thay đổi method equals() của lớp Media

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

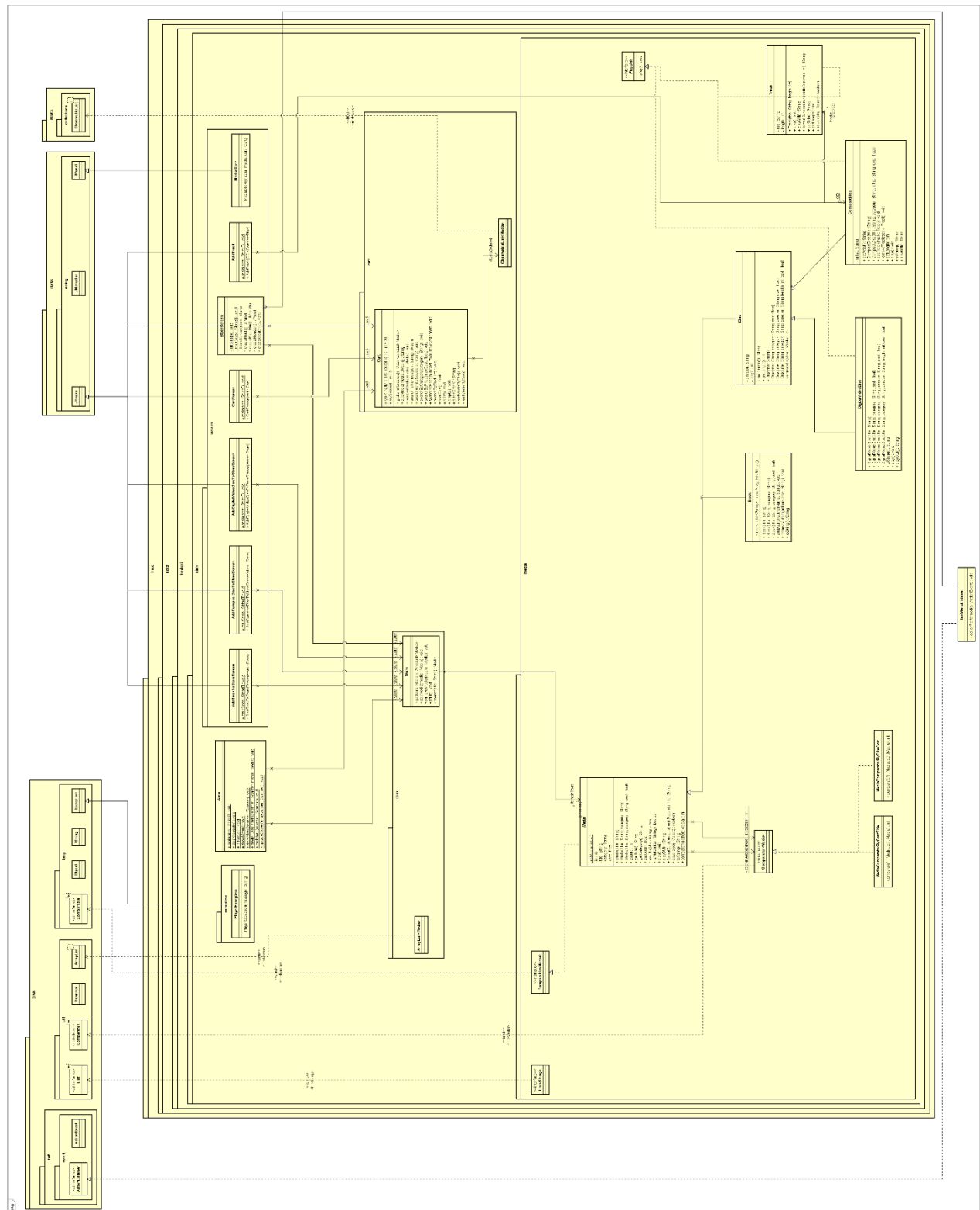
```
@Override
public boolean equals(Object obj) {
    if (obj == this) {
        return true;
    }
    if (obj == null || !(obj instanceof Media)) {
        return false;
    }
    Media otherMedia = (Media) obj;
    return this.getTitle() != null && this.getTitle().equals(otherMedia.getTitle());
}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

II. UML Diagram

1. Use-Case Diagram



2. Class Diagram (Final Lab05)



3. Exception Hierarchical

