

Student name: Lê Ngọc Anh Vũ

Student ID: 20236014

REPORT LAB 04

1 Swing components

Note: For the exercises in this lab (excluding the AIMS exercises), you will create a new Java project named `GUIProject`, and put all your source code in a package called “`hust.soict.dsai.swing`” (for DS & AI).

Note: From this section onwards, it is assumed that you are a DS-AI student, so your folder structure will contain the “`dsai`” package. If you are an HEDSPI or ICT student, you should replace the “`dsai`” string with “`hedspi`” or “`globalict`”.

In this exercise, we revisit the elements of the Swing API and compare them with those of AWT by implementing the same mini-application using the two libraries. The application is an accumulator which accumulates the values entered by the user and displays the sum.

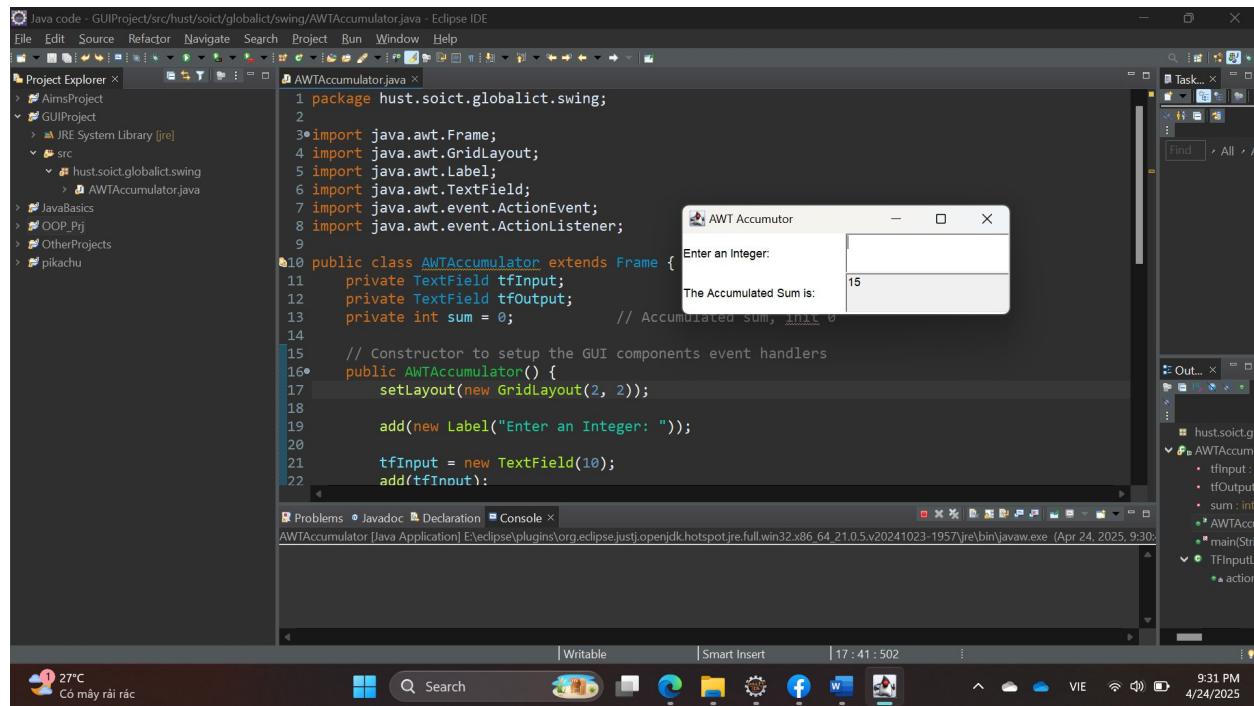
1.1 AWTAccumulator

1.1.1 Create class `AWTAccumulator` with the source code as below

```
1 package hust.soict.globalict.swing;
2
3 import java.awt.Frame;
4 import java.awt.GridLayout;
5 import java.awt.Label;
6 import java.awt.TextField;
7 import java.awt.event.ActionEvent;
8 import java.awt.event.ActionListener;
9
10 public class AWTAccumulator extends Frame {
11     private TextField tfInput;
12     private TextField tfOutput;
13     private int sum = 0;           // Accumulated sum, init 0
14
15     // Constructor to setup the GUI components event handlers
16     public AWTAccumulator() {
17         setLayout(new GridLayout(2, 2));
18
19         add(new Label("Enter an Integer: "));
20
21         tfInput = new TextField(10);
22         add(tfInput);
23         tfInput.addActionListener(new TFIputListener());
24
25         add(new Label("The Accumulated Sum is: "));
26
27         tfOutput = new TextField(10);
28         tfOutput.setEditable(false);
29         add(tfOutput);
```

```

30         setTitle("AWT Accumulator");
31         setSize(350, 120);
32         setVisible(true);
33     }
34 }
35 public static void main(String[] args) {
36     new AWTAccumulator();
37 }
38
39 public class TFInputListener implements ActionListener{
40
41     @Override
42     public void actionPerformed(ActionEvent e) {
43         int numberIn = Integer.parseInt(tfInput.getText());
44         sum += numberIn;
45         tfInput.setText("");
46         tfOutput.setText(sum + "");
47     }
48 }
49 }
```



1.2 Swing Accumulator

1.2.1 Create class `SwingAccumulator` with the source code as below:

```
1 package hust.soict.globalict.swing;
2
3 import java.awt.Container;
4 import java.awt.GridLayout;
5 import java.awt.event.ActionEvent;
6 import java.awt.event.ActionListener;
7
8 import javax.swing.JFrame;
9 import javax.swing.JLabel;
10 import javax.swing.JTextField;
11
12 public class SwingAccumulator extends JFrame {
13     private JTextField tfInput;
14     private JTextField tfOutput;
15     private int sum = 0;
16
17     public SwingAccumulator() {
18         Container cp = getContentPane();
19         cp.setLayout(new GridLayout(2, 2));
20
21         cp.add(new JLabel("Enter an Integer: "));
22
23         tfInput = new JTextField(10);
24         cp.add(tfInput);
25         tfInput.addActionListener(new TFIputListener());
26
27         cp.add(new JLabel("The Accumulated Sum is: "));
28
29         tfOutput = new JTextField(10);
30
31         tfOutput.setEditable(false);
32         cp.add(tfOutput);
33
34         setTitle("Swing Accumulator");
35         setSize(350, 120);
36         setVisible(true);
37     }
38
39     public static void main(String[] args) {
40         new SwingAccumulator();
41     }
42
43     private class TFIputListener implements ActionListener{
44         @Override
45         public void actionPerformed(ActionEvent e) {
46             int numberIn = Integer.parseInt(tfInput.getText());
47             sum += numberIn;
48             tfInput.setText("");
49             tfOutput.setText(sum + "");
50         }
51     }
52 }
```

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the project structure with files like AimsProject, GUIProject, JavaBasics, OOP.Pjt, OtherProjects, and pikachu.
- Code Editor:** Displays the `SwingAccumulator.java` file containing Java code for a Swing application. The code creates a JTextField for input, adds it to a JPanel, and sets up an ActionListener to update a JLabel with the accumulated sum.
- Run Output:** Shows the application's output window with the message "The Accumulated Sum is: 33".
- Task View:** Shows a task named "Swing Accumulator".
- Console:** Shows the command-line output of the application.
- System Tray:** Shows the weather (27°C) and system status icons.

2 Organizing Swing components with Layout Managers

2.2 Using JPanel as a secondary-level container to organize components:

2.2.1 Create class NumberGrid:

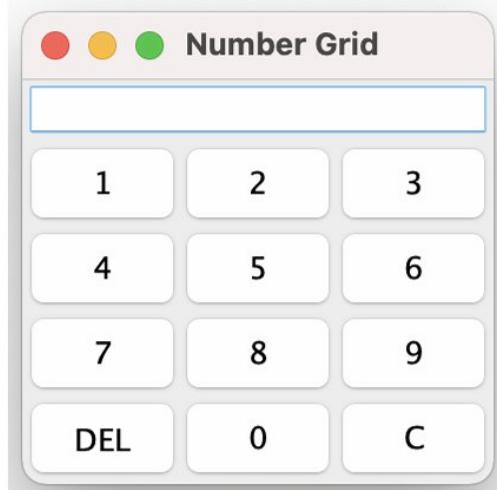


Figure 8. NumberGrid

```

1 package hust.soict.globalict.swing;
2
3 import java.awt.BorderLayout;
4
5 public class NumberGrid extends JFrame {
6     private JButton[] btnNumbers = new JButton[10];
7     private JButton btnDelete, btnReset;
8     private JTextField tfDisplay;
9
10    public NumberGrid() {
11
12        tfDisplay = new JTextField();
13        tfDisplay.setComponentOrientation(ComponentOrientation.RIGHT_TO_LEFT);
14
15        JPanel panelButtons = new JPanel(new GridLayout(4, 3));
16        addButtons(panelButtons);
17
18        Container cp = getContentPane();
19        cp.setLayout(new BorderLayout());
20        cp.add(tfDisplay, BorderLayout.NORTH);
21        cp.add(panelButtons, BorderLayout.CENTER);
22
23        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
24        setTitle("Number Grid");
25        setSize(200, 200);
26        setVisible(true);
27    }

```

2.2.2 Adding buttons

```

39    public void addButtons(JPanel panelButtons) {
40        ButtonListener btnListener = new ButtonListener();
41        for(int i = 1; i <= 9; i++) {
42            btnNumbers[i] = new JButton(""+i);
43            panelButtons.add(btnNumbers[i]);
44            btnNumbers[i].addActionListener(btnListener);
45        }
46
47        btnDelete = new JButton("DEL");
48        panelButtons.add(btnDelete);
49        btnDelete.addActionListener(btnListener);
50
51        btnNumbers[0] = new JButton("0");
52        panelButtons.add(btnNumbers[0]);
53        btnNumbers[0].addActionListener(btnListener);
54
55        btnReset = new JButton("C");
56        panelButtons.add(btnReset);
57        btnReset.addActionListener(btnListener);
58    }

```

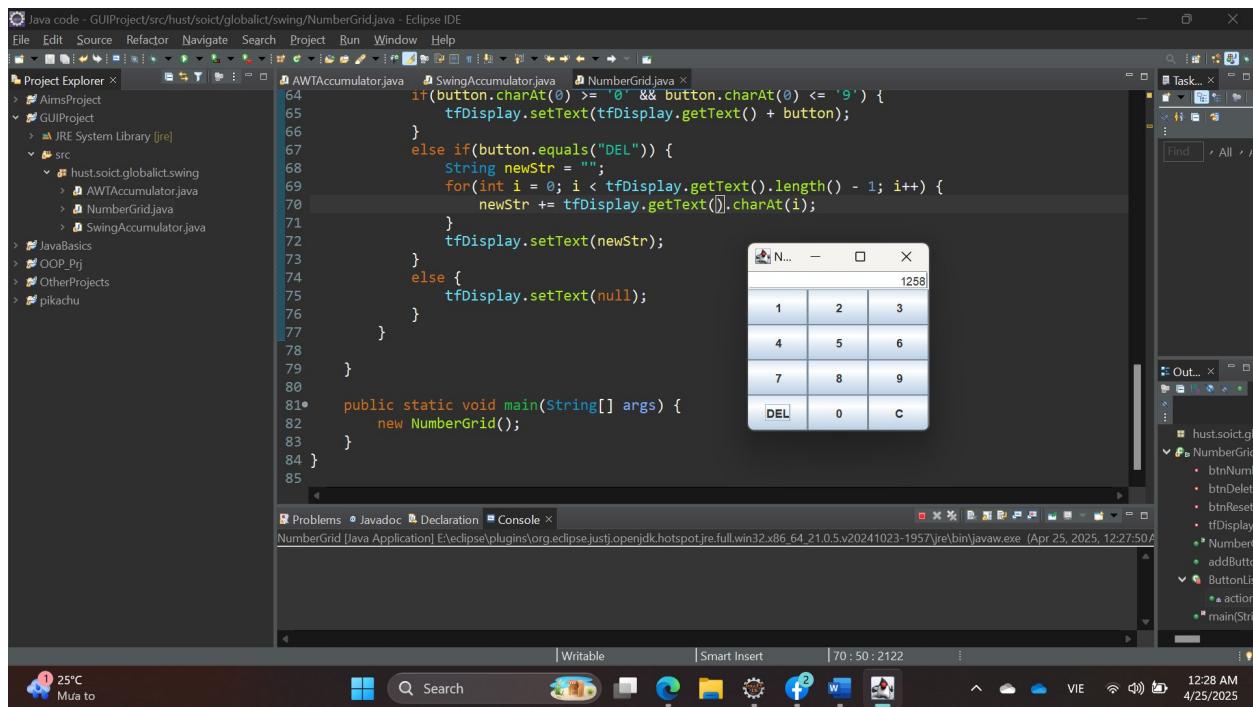
2.2.3 Complete inner class ButtonListener

```

60    private class ButtonListener implements ActionListener{
61        @Override
62        public void actionPerformed(ActionEvent e) {
63            String button = e.getActionCommand();
64            if(button.charAt(0) >= '0' && button.charAt(0) <= '9') {
65                tfDisplay.setText(tfDisplay.getText() + button);
66            }
67            else if(button.equals("DEL")) {
68                String newStr = "";
69                for(int i = 0; i < tfDisplay.getText().length() - 1; i++) {
70                    newStr += tfDisplay.getText().charAt(i);
71                }
72                tfDisplay.setText(newStr);
73            }
74            else {
75                tfDisplay.setText(null);
76            }
77        }
78    }

```

Testing:



3 Create a graphical user interface for AIMS with Swing

Additional Requirement: From this lab, we will split the AIMS system into two applications according to the role of the user: Customer and Store Manager

- Store Manager can view and update the store (i.e: Add Book, Add CD, Add DVD).
- Customers can view the store as the store manager, but they can choose to add media to the cart. Customers can also perform some functions related to the cart.

3.1 View Store Screen

For the view store screen, we will use the BorderLayout:

In the NORTH component, there will be the menu bar and the header

In the CENTER component, there will be a panel that uses the GridLayout, each cell is an item in the store.

3.1.1 Create the **StoreManagerScreen** class:

```
public class StoreManagerScreen{  
    private Store store;
```

Figure 17. Declaration of *StoreManagerScreen* class

This will be our view store screen while logging in with the role Store Manager.

Declare one attribute in the *StoreManagerScreen* class: *Store store*. This is because we need information on the items in the store to display them.

```
1 package hust.soict.globalict.aims.screen.manager;
2
3 import hust.soict.globalict.aims.store.Store;
4
5 public class StoreManagerScreen {
6     private Store store;
7 }
```

3.1.2 The NORTH component:

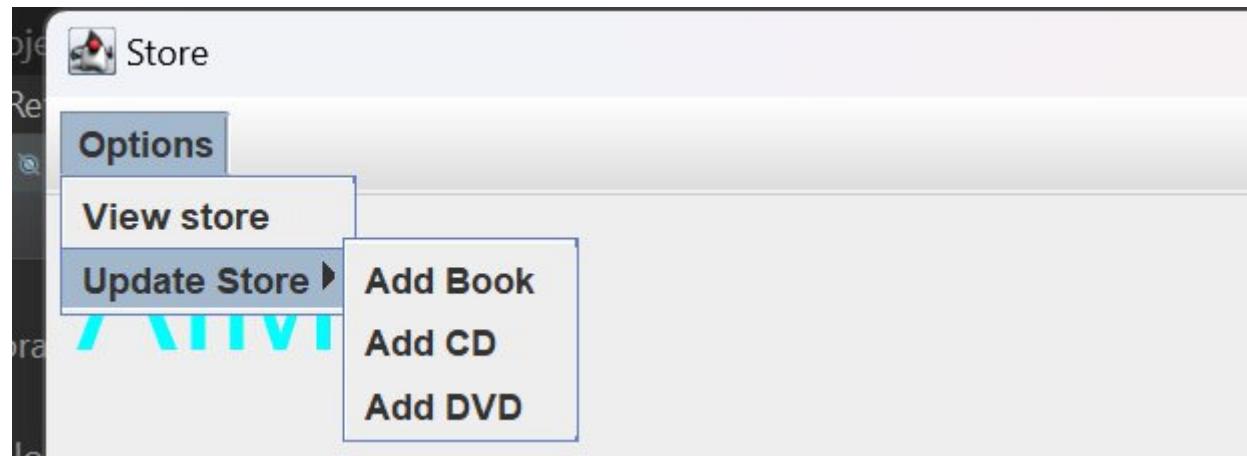
Create the method `createNorth()`, which will create our NORTH component:

```
41•     JPanel createNorth() {
42         JPanel north = new JPanel();
43         north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
44         north.add(createMenuBar());
45         north.add(createHeader());
46         return north;
47     }
```

Create the method `createMenuBar()`:

```
49•     JMenuBar createMenuBar() {
50         JMenu menu = new JMenu("Options");
51
52         menu.add(new JMenuItem("View store"));
53
54         JMenu smUpdateStore = new JMenu("Update Store");
55         smUpdateStore.add(new JMenuItem("Add Book"));
56         smUpdateStore.add(new JMenuItem("Add CD"));
57         smUpdateStore.add(new JMenuItem("Add DVD"));
58         menu.add(smUpdateStore);
59
60         JMenuBar menuBar = new JMenuBar();
61         menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
62         menuBar.add(menu);
63
64         return menuBar;
65     }
```

The resulting menu bar will look something like this:



Create the method `createHeader()`:

```
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);

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

    return header;
}
```

3.1.3 The CENTER component:

```
83• JPanel createCenter() {
84     JPanel center = new JPanel();
85     center.setLayout(new GridLayout(3, 3, 2, 2));
86
87     ArrayList<Media> mediaInStore = store.getItemsInStore();
88     for(int i = 0; i < 9; i++) {
89         MediaStore cell = new MediaStore(mediaInStore.get(i));
90         center.add(cell);
91     }
92
93     return center;
94 }
```

3.1.4 The `MediaStore` class:

Here, since the `MediaStore` is a GUI element, it extends the `JPanel` class. It has one attribute: `Media media`.

```
1 package hust.soict.globalict.aims.screen.manager;
2
3• import java.awt.Color;
4
5 public class MediaStore extends JPanel {
6     private Media media;
7
8     public MediaStore(Media media) {
9         this.media = media;
10        this.setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));
11
12        JLabel title = new JLabel(media.getTitle());
13        title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 15));
14        title.setAlignmentX(CENTER_ALIGNMENT);
15
16        JLabel cost = new JLabel(" " + media.getCost() + " $");
17        cost.setAlignmentX(CENTER_ALIGNMENT);
18
19        JPanel container = new JPanel();
20        container.setLayout(new FlowLayout(FlowLayout.CENTER));
21
22        if(media instanceof Playable) {
23            JButton playButton = new JButton("Play");
24            container.add(playButton);
25        }
26    }
27}
```

```

38     this.add(Box.createVerticalGlue());
39     this.add(title);
40     this.add(cost);
41     this.add(Box.createHorizontalGlue());
42     this.add(container);
43
44     this.setBorder(BorderFactory.createLineBorder(color.black));
45 }
46 }
```

3.1.5 Putting it all together:

Finally, we have all the component methods to use in the constructor of `StoreScreen`:

```

public StoreManagerScreen(Store store) {
    this.store = store;

    Container cp = getContentPane();
    cp.setLayout(new BorderLayout());
    cp.add(createNorth(), BorderLayout.NORTH);
    cp.add(createCenter(), BorderLayout.CENTER);

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

3.2 Update Store Screen

We have successfully set up all the components for our store, but they are just static – buttons and menu items don't respond when clicked. Now, it's your task to implement the handling of the event when the user interacts with:

- The menu bar:
 - When a user clicks on one of the items of the “Update Store” menu on the menu bar (such as “Add Book”, “Add CD”, or “Add DVD”), the application should switch to an appropriate new screen for the user to add the new item. This screen should have the same menu bar as the View Store Screen, so the user can go back to view the store.
 - When the user clicks on “View Store”, the application should switch to the View Store Screen.
- The buttons on MediaHome:
 - When the user clicks on the “Play” button, the Media should be played in a dialog window.
You can use JDialog here.

Note:

- For simplicity:
 - You only need to do the “Add item to store” screen, the “Remove item from store” is omitted. As shown above, there is no option of “Remove item from store” in the “Update Store” menu.
 - In the “Add item to store” Screen, you don't need to do data type validation yet.

I create three more classes: `AddDigitalVideoDiscToStoreScreen`, `AddCompactDiscToStoreScreen` and `AddBookToStoreScreen`

All of them extend from class `AddItemToStoreScreen`

For `AddItemToStoreScreen` class:

```
1 package hust.soict.globalict.aims.screen.manager;
2
3 import java.awt.BorderLayout;
4
5 public abstract class AddItemToStoreScreen extends JFrame {
6     private JLabel titleLabel, categoryLabel, costLabel;
7     private JTextField titleInput, categoryInput, costInput;
8     private JPanel main;
9     private JButton buttonFinish = new JButton();
10
11    public AddItemToStoreScreen() {
12        main = new JPanel();
13        main.setLayout(new GridLayout(0, 1, 5, 5));
14
15        titleLabel = new JLabel("Title*:");
16        titleInput = new JTextField(30);
17        titleInput.addActionListener(new ActionListener() {
18            @Override
19            public void actionPerformed(ActionEvent e) {
20                if(titleInput.getText().equals("") || costInput.getText().equals("")) {
21                    buttonFinish.setEnabled(false);
22                }
23                else {
24                    buttonFinish.setEnabled(true);
25                }
26            }
27        });
28        JPanel title = new JPanel(new GridLayout(1, 2, 10, 0));
29        title.add(titleLabel);
30
31        title.add(titleInput);
32        main.add(title);
33
34        categoryLabel = new JLabel("Category: ");
35        categoryInput = new JTextField(20);
36        JPanel category = new JPanel(new GridLayout(1, 2, 10, 0));
37        category.add(categoryLabel);
38        category.add(categoryInput);
39        main.add(category);
40
41        costLabel = new JLabel("Cost*:");
42        costInput = new JTextField(20);
43        costInput.addActionListener(new ActionListener() {
44            @Override
45            public void actionPerformed(ActionEvent e) {
46                if(titleInput.getText().equals("") || costInput.getText().equals("")) {
47                    buttonFinish.setEnabled(false);
48                }
49                else {
50                    buttonFinish.setEnabled(true);
51                }
52            }
53        });
54        JPanel cost = new JPanel(new GridLayout(1, 2, 10, 0));
55        cost.add(costLabel);
56        cost.add(costInput);
57        main.add(cost);
58    }
59}
```

```

74*     public void add(JPanel panel) {
75         main.add(panel);
76     }
77
78*     public JPanel getMainPanel() {
79         return main;
80     }
81
82*     public JPanel request() {
83         JPanel panel = new JPanel();
84         panel.setLayout(new BoxLayout(panel, BoxLayout.X_AXIS));
85
86         JLabel rqt = new JLabel("You have to fill all (*) fields");
87         rqt.setFont(new Font(rqt.getFont().getName(), Font.PLAIN, 15));
88         rqt.setForeground(Color.red);
89
90         panel.add(rqt);
91         return panel;
92     }
93
94*     public JPanel createNorth(String str) {
95         JPanel panelNorth = new JPanel(new BorderLayout());
96         panelNorth.add(createMenuBar(), BorderLayout.NORTH);
97         panelNorth.add(createHeader(str), BorderLayout.CENTER);
98         return panelNorth;
99     }
100
101*    public JPanel createHeader(String str) {
102        JPanel header = new JPanel();

104        JLabel title = new JLabel(str);
105        title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
106        title.setForeground(Color.CYAN);
107
108        header.add(Box.createRigidArea(new Dimension(10, 10)));
109        header.add(title, BorderLayout.CENTER);
110        header.add(Box.createHorizontalGlue());
111        header.add(Box.createRigidArea(new Dimension(10, 10)));
112
113        return header;
114    }
115
116*    public JPanel createSouth() {
117        JPanel panel = new JPanel(new BorderLayout());
118        panel.add(request(), BorderLayout.WEST);
119        buttonFinish = createButton();
120        buttonFinish.setEnabled(false);
121        panel.add(buttonFinish, BorderLayout.EAST);
122        panel.setBorder(new EmptyBorder(0, 5, 5, 5));
123        return panel;
124    }
125
126*    public String getTitle() {
127        return titleInput.getText();
128    }
129
130*    public String getCategory() {
131        return categoryInput.getText();
132    }

134*    public String getCost() {
135        return costInput.getText();
136    }
137
138*    public void setTitle(String str) {
139        titleInput.setText(str);
140    }
141
142*    public void setCategory(String str) {
143        categoryInput.setText(str);
144    }
145
146*    public void setCost(String str) {
147        costInput.setText(str);
148    }
149
150    public abstract JButton createButton();
151
152    public abstract JMenuBar createMenuBar();
153 }
```

For AddDigitalVideoDiscToStoreScreen class:

```
1 package hust.soict.globalict.aims.screen.manager;
2
3 import java.awt.BorderLayout;
4
5 public class AddDigitalVideoDiscToStoreScreen extends AddItemToStoreScreen {
6     private JLabel lengthLabel, directorLabel;
7     private JTextField lengthInput, directorInput;
8     private JFrame frame = new JFrame();
9
10    public AddDigitalVideoDiscToStoreScreen() {
11        super();
12        lengthLabel = new JLabel("Length: ");
13        lengthInput = new JTextField(10);
14        JPanel length = new JPanel(new GridLayout(1, 2, 10, 0));
15        length.add(lengthLabel);
16        length.add(lengthInput);
17        add(length);
18
19        directorLabel = new JLabel("Director: ");
20        directorInput = new JTextField(20);
21        JPanel director = new JPanel(new GridLayout(1, 2, 10, 0));
22        director.add(directorLabel);
23        director.add(directorInput);
24        add(director);
25
26        JPanel panel = new JPanel(new BorderLayout());
27
28        panel.add(createNorth(), BorderLayout.NORTH);
29        getMainPanel().setBorder(new EmptyBorder(10, 250, 250, 250));
30
31        panel.add(getMainPanel(), BorderLayout.CENTER);
32        panel.add(createSouth(), BorderLayout.SOUTH);
33
34        frame.add(panel);
35        frame.setTitle("Adding DVD");
36        frame.setSize(800, 600);
37        frame.setLocationRelativeTo(null);
38        frame.setVisible(true);
39    }
40
41    public JPanel createNorth() {
42        JPanel panelNorth = new JPanel(new BorderLayout());
43        panelNorth.add(createMenuBar(), BorderLayout.NORTH);
44        panelNorth.add(createHeader("Digital Video Disc"), BorderLayout.CENTER);
45        return panelNorth;
46    }
47
48    @Override
49    public JMenuBar createMenuBar() {
50        JMenu menu = new JMenu("Options");
51
52        JMenuItem viewStore = new JMenuItem("View store");
53        viewStore.addActionListener(new ActionListener() {
54            @Override
55            public void actionPerformed(ActionEvent e) {
56                public void actionPerformed(ActionEvent e) {
57                    frame.dispose();
58                    new StoreManagerScreen(Main.store);
59                }
60            });
61        });
62    }
63
64
65    @Override
66    public JMenuBar createMenuBar() {
67        JMenu menu = new JMenu("Options");
68
69        JMenuItem viewStore = new JMenuItem("View store");
70        viewStore.addActionListener(new ActionListener() {
71            @Override
72            public void actionPerformed(ActionEvent e) {
73                frame.dispose();
74                new StoreManagerScreen(Main.store);
75            }
76        });
77    }
78}
```

```

77     menu.add(viewStore);
78
79     JMenu smUpdateStore = new JMenu("Update Store");
80     JMenuItem addBook = new JMenuItem("Add Book");
81     addBook.addActionListener(new ActionListener() {
82         @Override
83         public void actionPerformed(ActionEvent e) {
84             frame.dispose();
85             new AddBookToStoreScreen();
86         }
87     });
88     smUpdateStore.add(addBook);
89
90     JMenuItem addCD = new JMenuItem("Add CD");
91     addCD.addActionListener(new ActionListener() {
92         @Override
93         public void actionPerformed(ActionEvent e) {
94             frame.dispose();
95             new AddCompactDiscToStoreScreen();
96         }
97     });
98     smUpdateStore.add(addCD);
99
100    JMenuItem addDVD = new JMenuItem("Add DVD");
101    addDVD.setEnabled(false);
102    smUpdateStore.add(addDVD);
103    menu.add(smUpdateStore);
104
105    JMenuBar menuBar = new JMenuBar();

```

```

106        menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
107        menuBar.add(menu);
108
109        return menuBar;
110    }
111
112    @Override
113    public JButton createButton() {
114        JButton btnFinish = new JButton("Finish");
115        btnFinish.addActionListener(new ActionListener() {
116            @Override
117            public void actionPerformed(ActionEvent e) {
118                String length = lengthInput.getText();
119                if(length.equals("")) {
120                    length = "0";
121                }
122                Main.store.addMedia(new DigitalVideoDisc(0, getTitle(),
123                                                getCategory(), directorInput.getText(),
124                                                Integer.parseInt(length),
125                                                Float.parseFloat(getCost())));
126                btnFinish.setEnabled(false);
127                setTitle("");
128                setCategory("");
129                directorInput.setText("");
130                lengthInput.setText("");
131                setCost("");
132            }
133        });
134        return btnFinish;

```

For AddCompactDiscToStoreScreen class:

```

1 package hust.soict.globalict.aims.screen.manager;
2
3 import java.awt.BorderLayout;
4
5 public class AddCompactDiscToStoreScreen extends AddItemToStoreScreen {
6     private JLabel artistLabel, lengthLabel, directorLabel;
7     private JTextField artistInput, lengthInput, directorInput;
8     private JFrame frame = new JFrame();
9
10    public AddCompactDiscToStoreScreen() {
11        super();
12        artistLabel = new JLabel("Artist: ");
13        artistInput = new JTextField(20);
14        JPanel artist = new JPanel(new GridLayout(1, 2, 10, 0));
15        artist.add(artistLabel);
16        artist.add(artistInput);
17        add(artist);
18    }
19
20    public void actionPerformed(ActionEvent e) {
21        String artistName = artistInput.getText();
22        String lengthString = lengthInput.getText();
23        float cost = Float.parseFloat(lengthString);
24        String directorName = directorInput.getText();
25        int id = Main.store.addMedia(new DigitalVideoDisc(0, artistName,
26                                                directorName, "", cost));
27        if(id != -1) {
28            JOptionPane.showMessageDialog(frame, "Success!");
29        } else {
30            JOptionPane.showMessageDialog(frame, "Error!");
31        }
32    }
33}

```

```

36     lengthLabel = new JLabel("Length: ");
37     lengthInput = new JTextField(10);
38     JPanel length = new JPanel(new GridLayout(1, 2, 10, 0));
39     length.add(lengthLabel);
40     length.add(lengthInput);
41     add(length);
42
43     directorLabel = new JLabel("Director: ");
44     directorInput = new JTextField(20);
45     JPanel director = new JPanel(new GridLayout(1, 2, 10, 0));
46     director.add(directorLabel);
47     director.add(directorInput);
48     add(director);
49
50     JPanel panel = new JPanel(new BorderLayout());
51
52     panel.add(createNorth(), BorderLayout.NORTH);
53     getMainPanel().setBorder(new EmptyBorder(10, 250, 200, 250));
54     panel.add(getMainPanel(), BorderLayout.CENTER);
55     panel.add(createSouth(), BorderLayout.SOUTH);
56
57     frame.add(panel);
58     frame.setTitle("Adding CD");
59     frame.setSize(800, 600);
60     frame.setLocationRelativeTo(null);
61     frame.setVisible(true);
62 }
63
64* public JPanel createNorth() {
65
66     JPanel panelNorth = new JPanel(new BorderLayout());
67     panelNorth.add(createMenuBar(), BorderLayout.NORTH);
68     panelNorth.add(createHeader("Compact Disc"), BorderLayout.CENTER);
69     return panelNorth;
70 }
71
72* @Override
73     public JMenuBar createMenuBar() {
74         JMenu menu = new JMenu("Options");
75
76         JMenuItem viewStore = new JMenuItem("View store");
77         viewStore.addActionListener(new ActionListener() {
78             @Override
79             public void actionPerformed(ActionEvent e) {
80                 frame.dispose();
81                 new StoreManagerScreen(Main.store);
82             }
83         });
84         menu.add(viewStore);
85
86         JMenu smUpdateStore = new JMenu("Update Store");
87         JMenuItem addBook = new JMenuItem("Add Book");
88         addBook.addActionListener(new ActionListener() {
89             @Override
90             public void actionPerformed(ActionEvent e) {
91                 frame.dispose();
92                 new AddBookToStoreScreen();
93             }
94         });
95
96         JMenuItem addCD = new JMenuItem("Add CD");
97         addCD.setEnabled(false);
98         smUpdateStore.add(addCD);
99
100        JMenuItem addDVD = new JMenuItem("Add DVD");
101        addDVD.addActionListener(new ActionListener() {
102            @Override
103            public void actionPerformed(ActionEvent e) {
104                frame.dispose();
105                new AddDigitalVideoDiscToStoreScreen();
106            }
107        });

```

```

108     smUpdateStore.add(addDVD);
109     menu.add(smUpdateStore);
110
111     JMenuBar menuBar = new JMenuBar();
112     menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
113     menuBar.add(menu);
114
115     return menuBar;
116 }
117
118* @Override
119 public JButton createButton() {
120     JButton btnFinish = new JButton("Finish");
121*     btnFinish.addActionListener(new ActionListener() {
122*         @Override
123         public void actionPerformed(ActionEvent e) {
124             String length = lengthInput.getText();
125             if(length.equals("")) {
126                 length = "0";
127             }
128             Main.store.addMedia(new CompactDisc(0, getTitle(),
129                 getCategory(), artistInput.getText(), directorInput.getText(),
130                 Integer.parseInt(length),
131                 Float.parseFloat(getCost())));
132             btnFinish.setEnabled(false);
133             setTitle("");
134             setCategory("");
135             artistInput.setText("");
136             directorInput.setText("");
137             lengthInput.setText("");
138             setCost("");
139         }
140     });
141     return btnFinish;
142 }
143 }
```

For AddBookToStoreScreen class:

```

1 package hust.soict.globalict.aims.screen.manager;
2
3* import java.awt.BorderLayout;[]
22
23 public class AddBookToStoreScreen extends AddItemToStoreScreen {
24     private JFrame frame = new JFrame();
25
26*     public AddBookToStoreScreen() {
27         super();
28         JPanel panel = new JPanel(new BorderLayout());
29
30         panel.add(createNorth("Book"), BorderLayout.NORTH);
31         getMainPanel().setBorder(new EmptyBorder(10, 200, 200, 200));
32         panel.add(getMainPanel(), BorderLayout.CENTER);
33         panel.add(createSouth(), BorderLayout.SOUTH);
34
35         frame.add(panel);
36         frame.setTitle("Adding book");
37         frame.setSize(700, 500);
38         frame.setLocationRelativeTo(null);
39         frame.setVisible(true);
40     }
41
42*     @Override
43     public JMenuBar createMenuBar() {
44         JMenu menu = new JMenu("Options");

```

```

46     JMenuItem viewStore = new JMenuItem("View store");
47     viewStore.addActionListener(new ActionListener() {
48         @Override
49         public void actionPerformed(ActionEvent e) {
50             frame.dispose();
51             new StoreManagerScreen(Main.store);
52         }
53     });
54     menu.add(viewStore);
55
56     JMenu smUpdateStore = new JMenu("Update Store");
57     JMenuItem addBook = new JMenuItem("Add Book");
58     addBook.setEnabled(false);
59     smUpdateStore.add(addBook);
60
61     JMenuItem addCD = new JMenuItem("Add CD");
62     addCD.addActionListener(new ActionListener() {
63         @Override
64         public void actionPerformed(ActionEvent e) {
65             frame.dispose();
66             new AddCompactDiscToStoreScreen();
67         }
68     });
69     smUpdateStore.add(addCD);
70
71     JMenuItem addDVD = new JMenuItem("Add DVD");
72     addDVD.addActionListener(new ActionListener() {
73         @Override
74         public void actionPerformed(ActionEvent e) {

```

```

75             frame.dispose();
76             new AddDigitalVideoDiscToStoreScreen();
77         }
78     });
79     smUpdateStore.add(addDVD);
80     menu.add(smUpdateStore);
81
82     JMenuBar menuBar = new JMenuBar();
83     menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
84     menuBar.add(menu);
85
86     return menuBar;
87 }

```

```

89     @Override
90     public JButton createButton() {
91         JButton btnFinish = new JButton("Finish");
92         btnFinish.addActionListener(new ActionListener() {
93             @Override
94             public void actionPerformed(ActionEvent e) {
95                 Main.store.addMedia(new Book(0, getTitle(),
96                     getCategory(), Float.parseFloat(getCost())));
97                 btnFinish.setEnabled(false);
98                 setTitle("");
99                 setCategory("");
100                 setCost("");
101             }
102         });
103         return btnFinish;
104     }
105 }

```

Modifying MediaStore class:

```

37         if(media instanceof Playable) {
38             JButton playButton = new JButton("Play");
39             playButton.addActionListener(new ActionListener() {
40                 @Override
41                 public void actionPerformed(ActionEvent e) {
42                     JDialog d = new JDialog();
43                     d.setTitle("Playing media");
44                     d.setLayout(new BorderLayout());
45                     JLabel l = new JLabel(media.getTitle() + " is playing at the present");
46                     d.add(l, BorderLayout.CENTER);
47                     d.setSize(300, 150);
48                     d.setLocationRelativeTo(null);
49                     d.setVisible(true);
50                 }
51             });
52             container.add(playButton);
53         }

```

Modifying StoreManagerScreen class:

```

26 public class StoreManagerScreen extends JFrame {
27     private Store store;
28     private Container cp;
29     private JFrame frame;
30
31     public StoreManagerScreen(Store store) {
32         this.store = store;
33         frame = new JFrame();
34
35         cp = getContentPane();
36         cp.setLayout(new BorderLayout());
37         cp.add(createNorth(), BorderLayout.NORTH);
38         cp.add(createCenter(), BorderLayout.CENTER);
39
40         frame.add(cp);
41         frame.setTitle("Store");
42         frame.setSize(1024, 680);
43         frame.setLocationRelativeTo(null);
44         frame.setVisible(true);
45     }

```

```

55     JMenuBar createMenuBar() {
56         JMenu menu = new JMenu("Options");
57
58         JMenuItem viewStore = new JMenuItem("View store");
59         viewStore.setEnabled(false);
60         menu.add(viewStore);
61
62         JMenu smUpdateStore = new JMenu("Update Store");
63         JMenuItem addBook = new JMenuItem("Add Book");
64         addBook.addActionListener(new ActionListener() {
65             @Override
66             public void actionPerformed(ActionEvent e) {
67                 frame.dispose();
68                 new AddBookToStoreScreen();
69             }
70         });
71         smUpdateStore.add(addBook);
72
73         JMenuItem addCD = new JMenuItem("Add CD");
74         addCD.addActionListener(new ActionListener() {
75             @Override
76             public void actionPerformed(ActionEvent e) {
77                 frame.dispose();
78                 new AddCompactDiscToStoreScreen();
79             }
80         });
81         smUpdateStore.add(addCD);

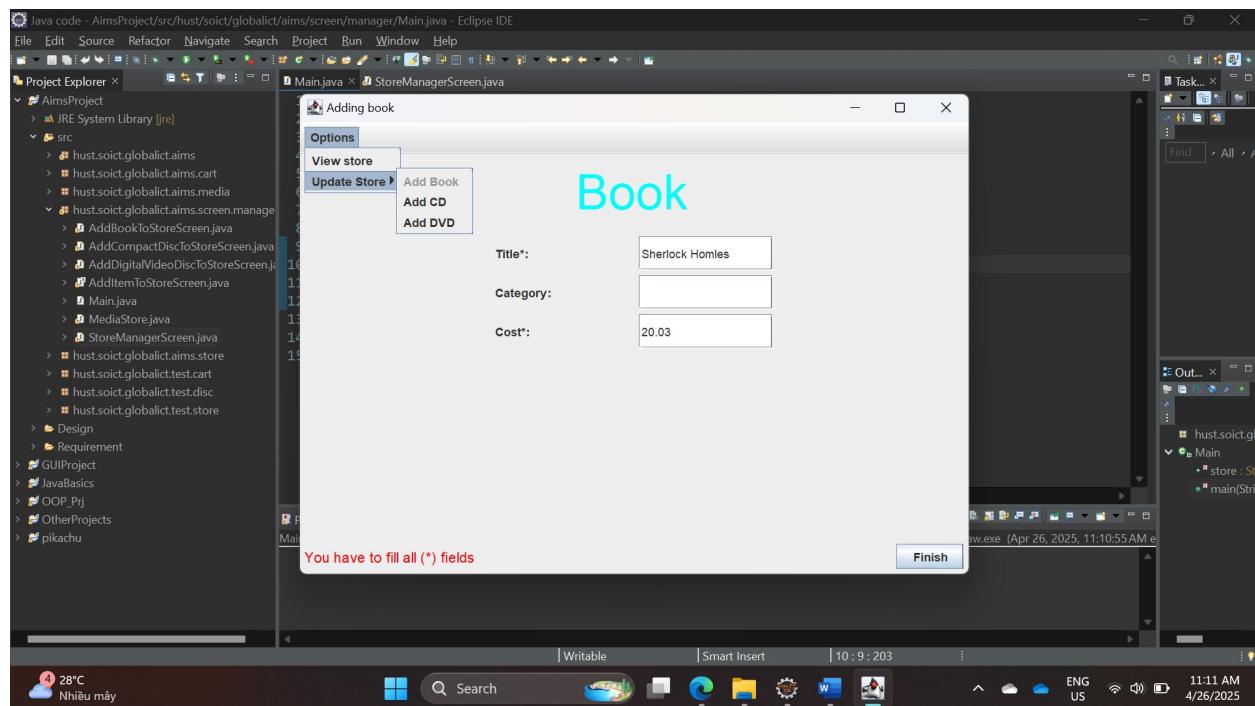
```

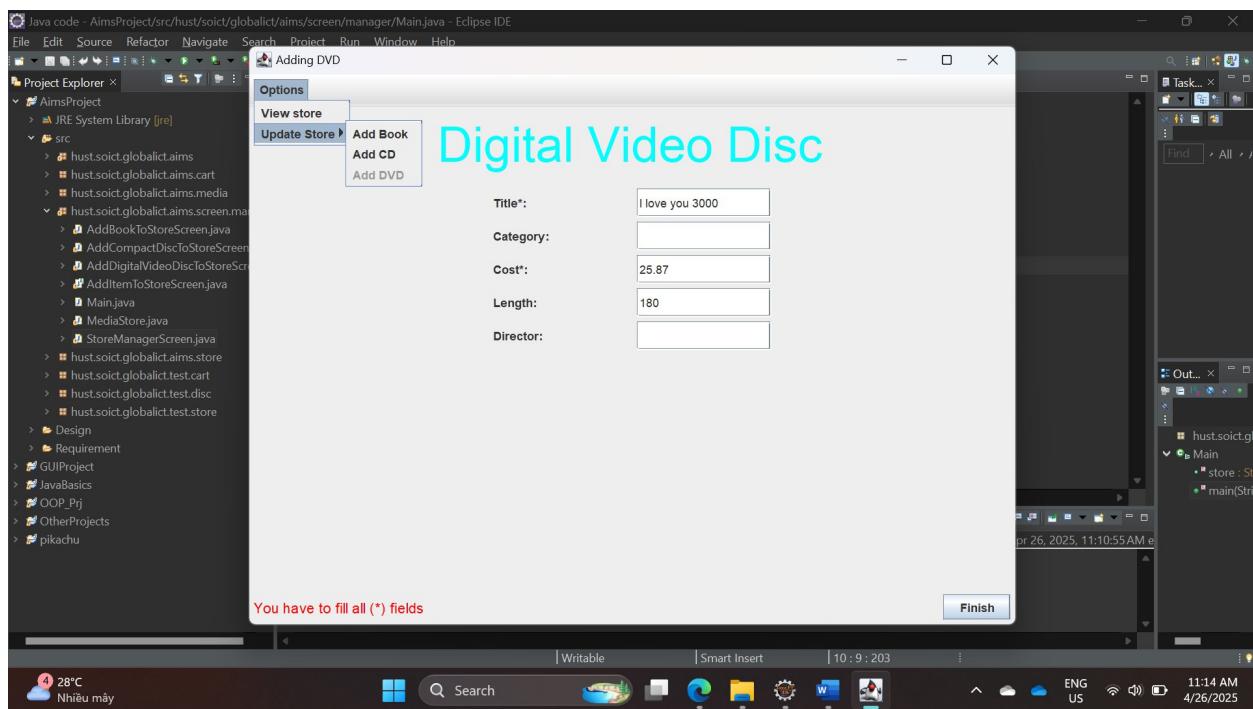
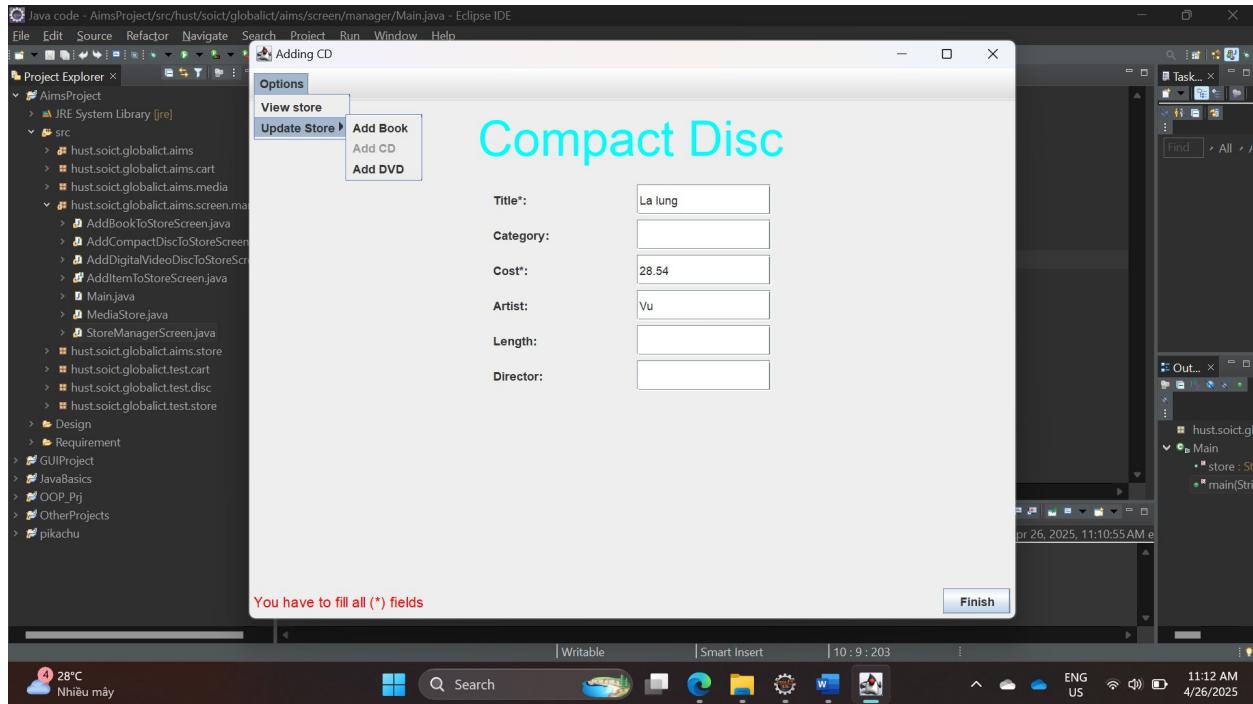
```

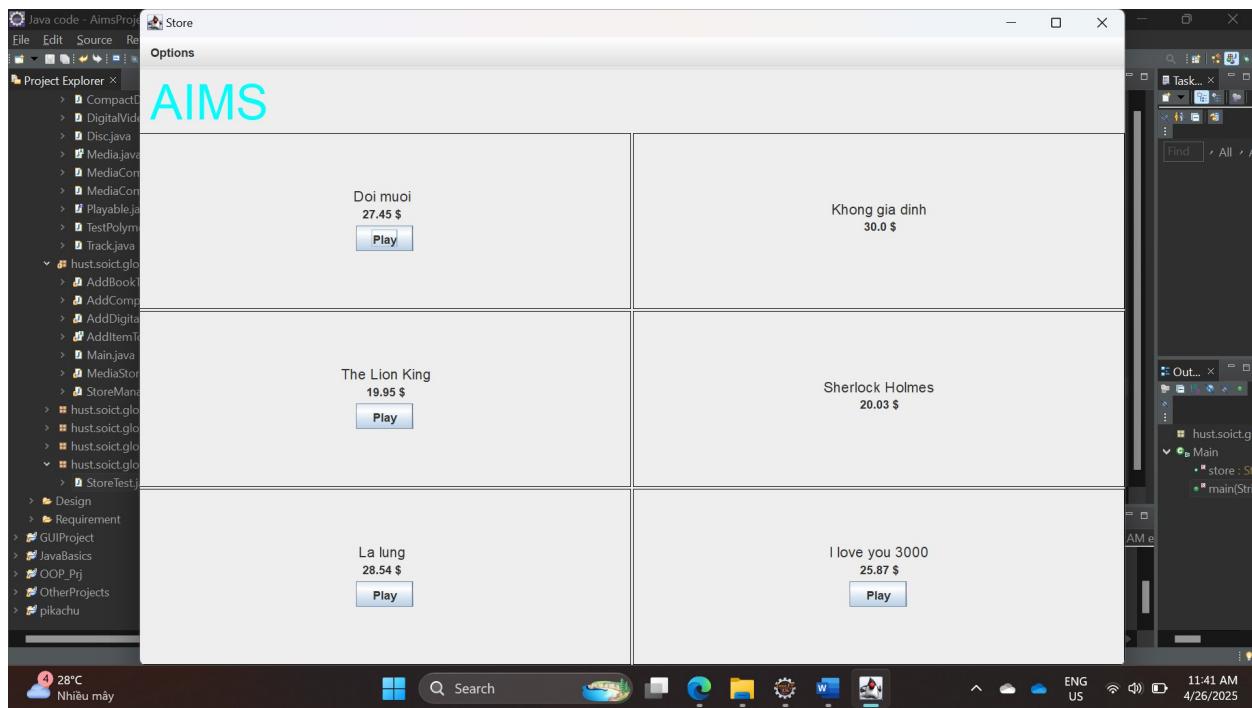
83     JMenuItem addDVD = new JMenuItem("Add DVD");
84     addDVD.addActionListener(new ActionListener() {
85         @Override
86         public void actionPerformed(ActionEvent e) {
87             frame.dispose();
88             new AddDigitalVideoDiscToStoreScreen();
89         }
90     });
91     smUpdateStore.add(addDVD);
92     menu.add(smUpdateStore);
93
94     JMenuBar menuBar = new JMenuBar();
95     menuBar.setLayout(new FlowLayout(FlowLayout.LEFT));
96     menuBar.add(menu);
97
98     return menuBar;
99 }

```

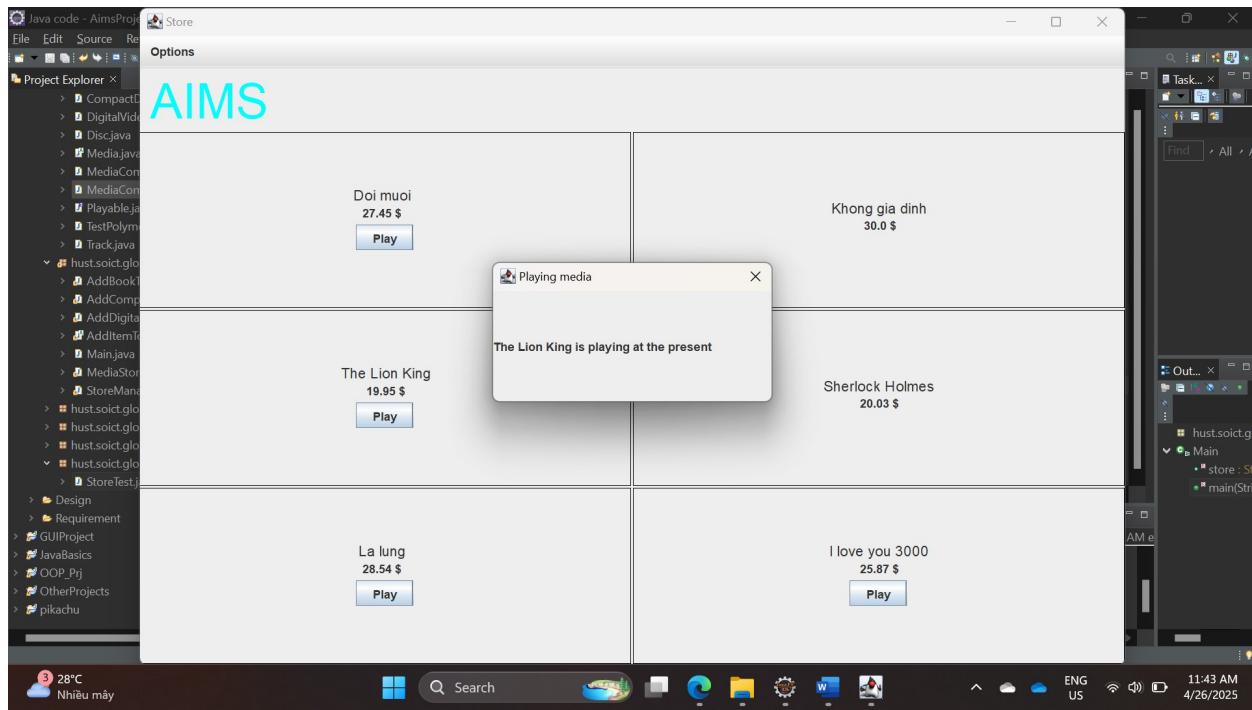
Test program:







Play a media:



- Update the UML class diagram for the **AimsProject**.

