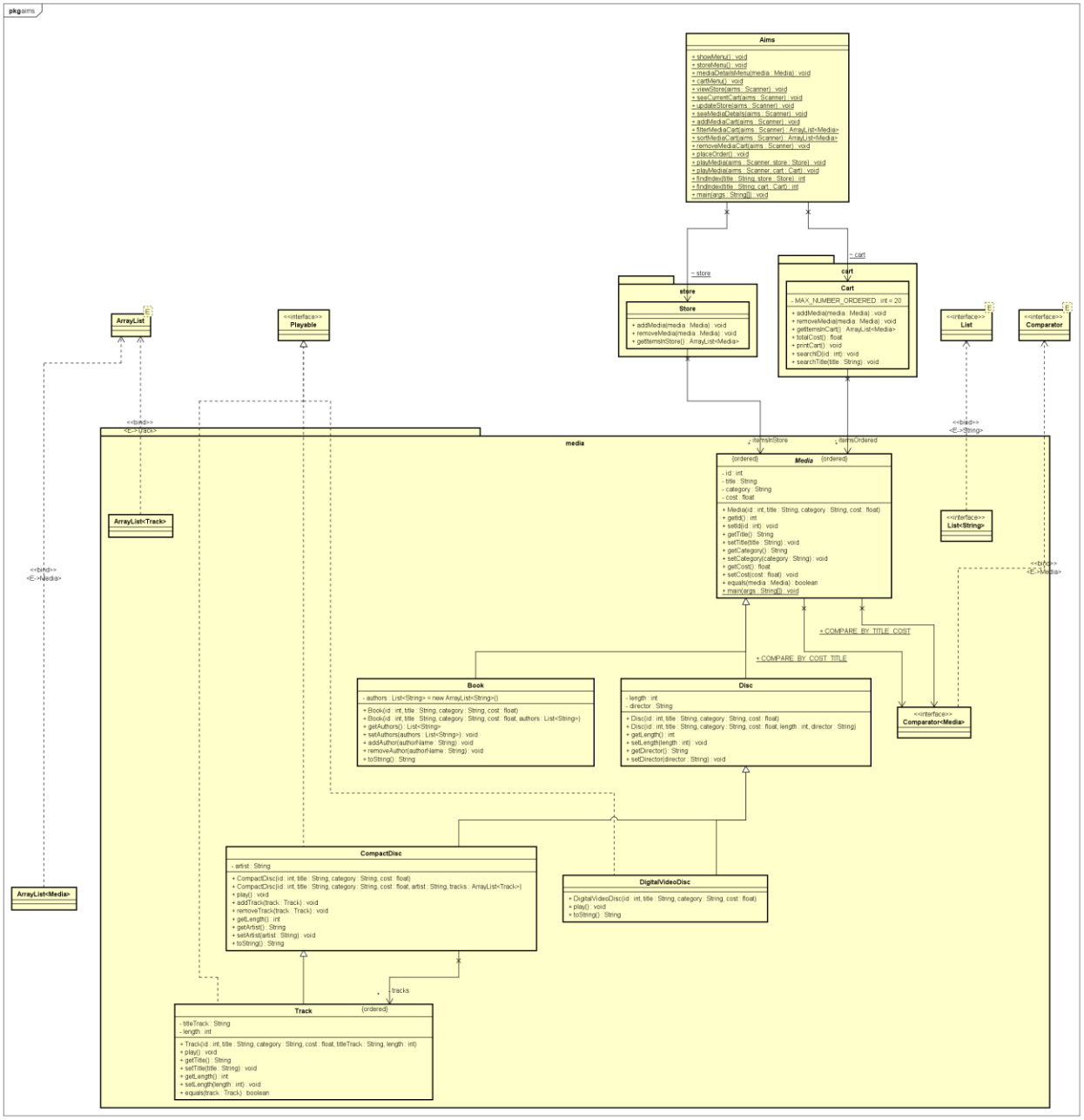
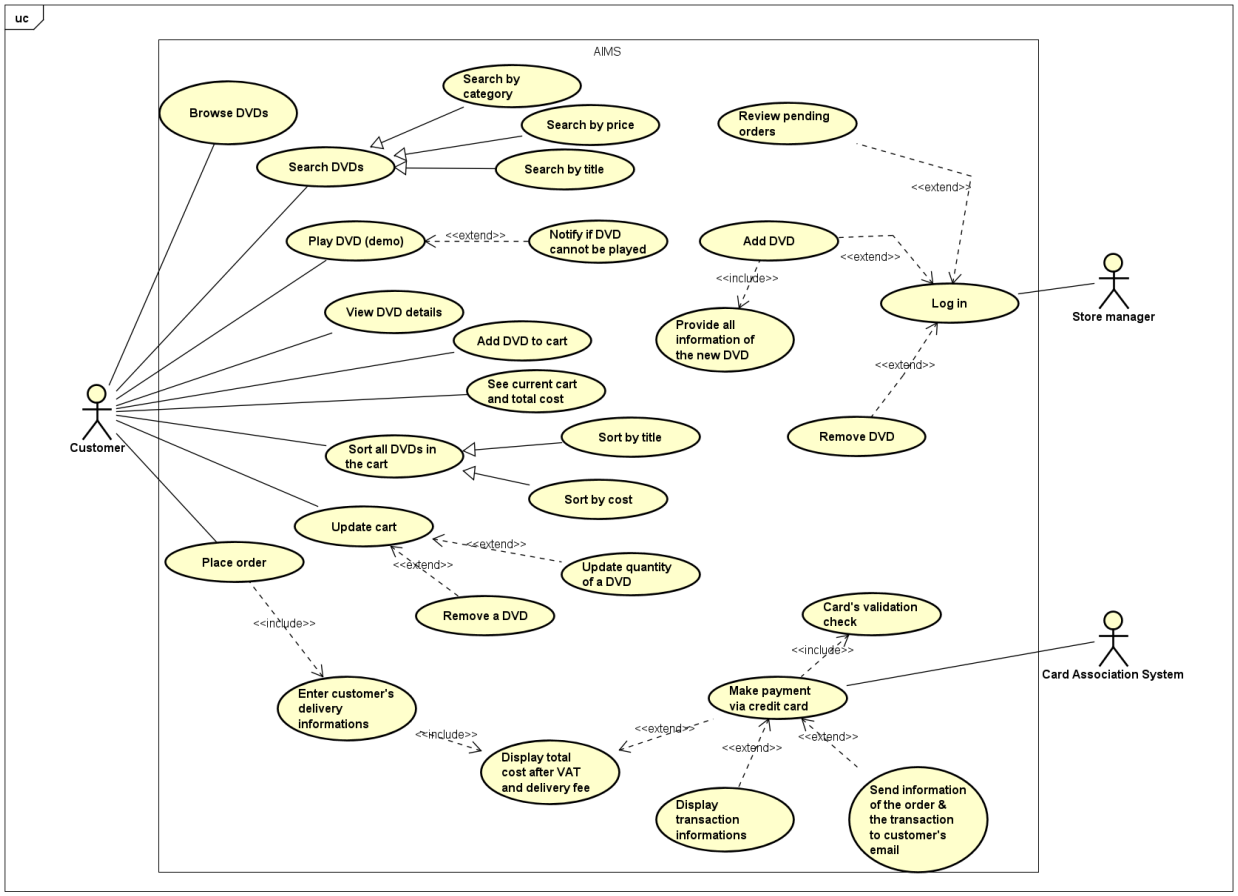
Class Diagram



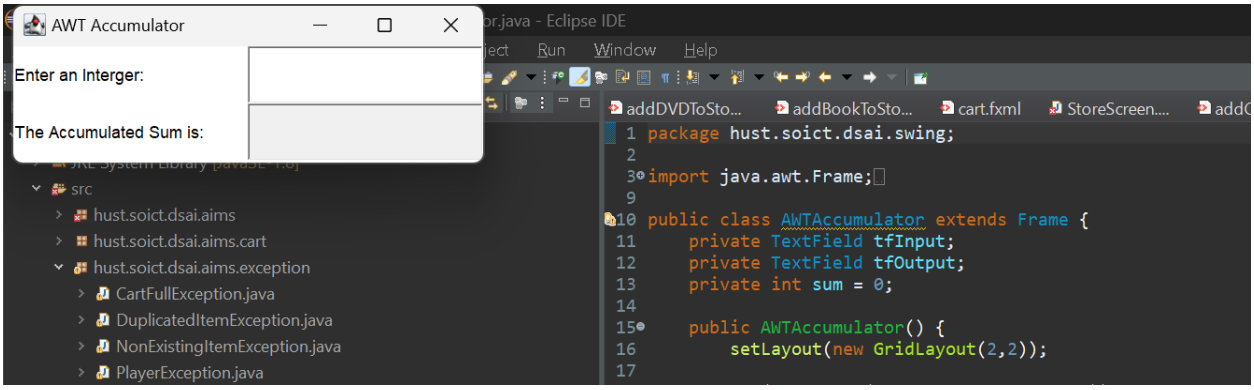
Usecase Diagram



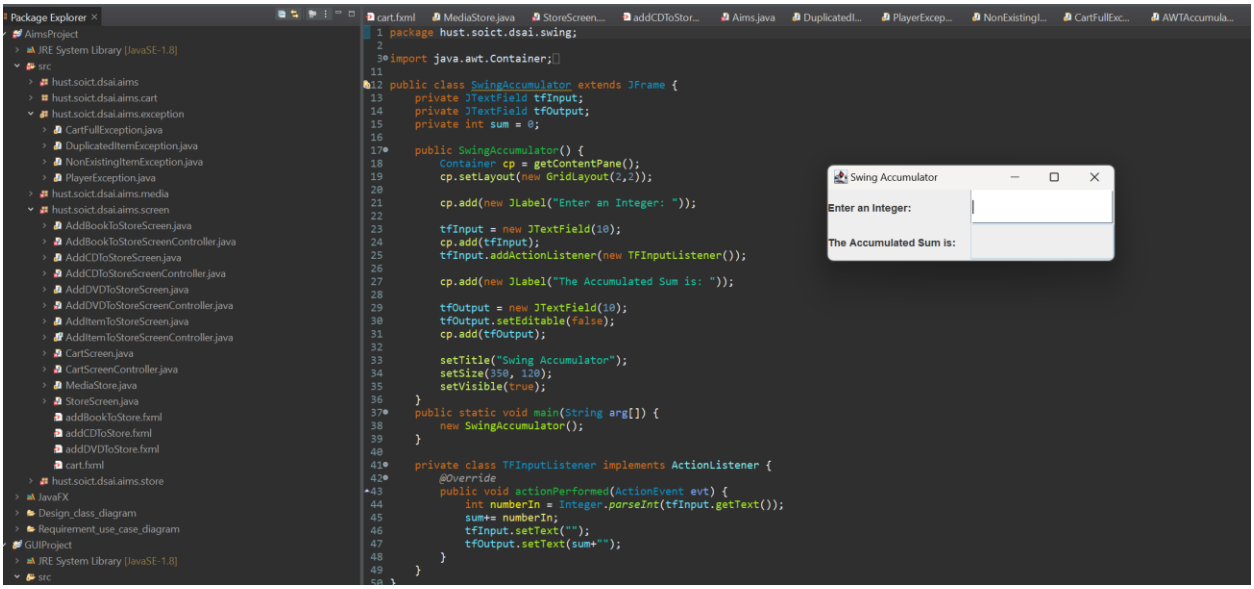
1. Swing Components

1.1. AWT Accumulator

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

import java.awt.Frame;

public class AWTAccumulator extends Frame {
    private TextField tfInput;
    private TextField tfOutput;
    private int sum = 0;

    public AWTAccumulator() {
        setLayout(new GridLayout(2,2));

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

        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 arg[]) {
        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 +"");
        }
    }
}
```



1.2. Swing Accumulator



Compare Swing and AWT elements

The Top-Level Containers:

- AWT: The top-level container in AWT is called "Frame".

- Swing: The top-level container in Swing is called "JFrame". It is an extension of Frame and provides additional features and functionality.

The Class Names of Components:

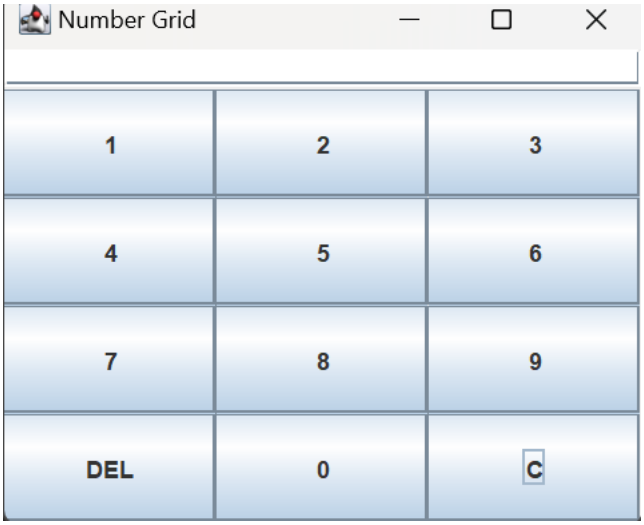
- AWT: The class names of AWT components usually start with a capital letter, such as Button, TextField, and Label.

- Swing: The Swing components generally have the same names as their AWT counterparts but start with a "J" prefix. For example, the equivalent Swing components would be JButton, JTextField, and JLabel.

1.3. NumberGrid

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

import java.awt.BorderLayout;

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);
    }

```

```
    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")) {
                //handles the "DEL" button
                String text = tfDisplay.getText();
                if (text.length() > 0) {
                    tfDisplay.setText(text.substring(0, text.length() - 1));
                }
            }
            else {
                //handles the "C" button
                tfDisplay.setText("");
            }
        }
    }

    public static void main(String arg[]) {
        new NumberGrid();
    }
}
```



2. Create a graphical user interface for AIMS with Swing

2.1. View Store Screen

2.1.1. Create the StoreScreen class

```
public class StoreScreen extends JFrame {
    private Store store;
    private Cart cart;

    public static void main(String[] args) throws Exception {
        // Test
        Store myStore = new Store();
        Cart myCart = new Cart();
```

2.1.2. 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;
    }
```

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

        JMenu smUpdateStore = new JMenu("Update Store");
        JMenuItem addBook = new JMenuItem("Add Book");
        addBook.addActionListener(new AddBookListener());
        smUpdateStore.add(addBook);
        JMenuItem addCD = new JMenuItem("Add CD");
        addCD.addActionListener(new AddCDListener());
        smUpdateStore.add(addCD);
        JMenuItem addDVD = new JMenuItem("Add DVD");
        addDVD.addActionListener(new AddDVDListener());
        smUpdateStore.add(addDVD);

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

        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;
    }
```

2.1.3. The CENTER Component

```
    JPanel createCenter() {
        JPanel center = new JPanel();
        center.setLayout(new GridLayout(5, 5, 2, 2));

        ArrayList<Media> mediaInStore = store.getItemsInStore();
        for (Media media : mediaInStore) {
            MediaStore cell = new MediaStore(media, cart);
            center.add(cell);
        }
```

2.1.4. The MediaStore Class

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

import hust.soict.dsai.aims.media.*;

public class MediaStore extends JPanel {
    private Media media;
    private Cart cart;

    public MediaStore(Media media, Cart 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("" + media.getCost() + " $");
        cost.setAlignmentX(CENTER_ALIGNMENT);

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

        JButton addToCartButton = new JButton("Add to cart");
        addToCartButton.addActionListener(new AddToCartListener());
        container.add(addToCartButton);

        JButton detailsButton = new JButton("View details");
        detailsButton.addActionListener(new DetailsListener());
        container.add(detailsButton);

        if (media instanceof Playable) {
            JButton playButton = new JButton("Play");
            playButton.addActionListener(new PlayButtonListener());
            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));
    }
```

2.1.5. Putting it all together

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

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

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

    Dimension dim = Toolkit.getDefaultToolkit().getScreenSize();
    int w = getSize().width;
    int h = getSize().height;
    int x = (dim.width - w) / 2;
    int y = (dim.height - h) / 2;
    setLocation(x, y);
}
```

2.2. *Adding more user interaction*

3. JavaFX API

4. 

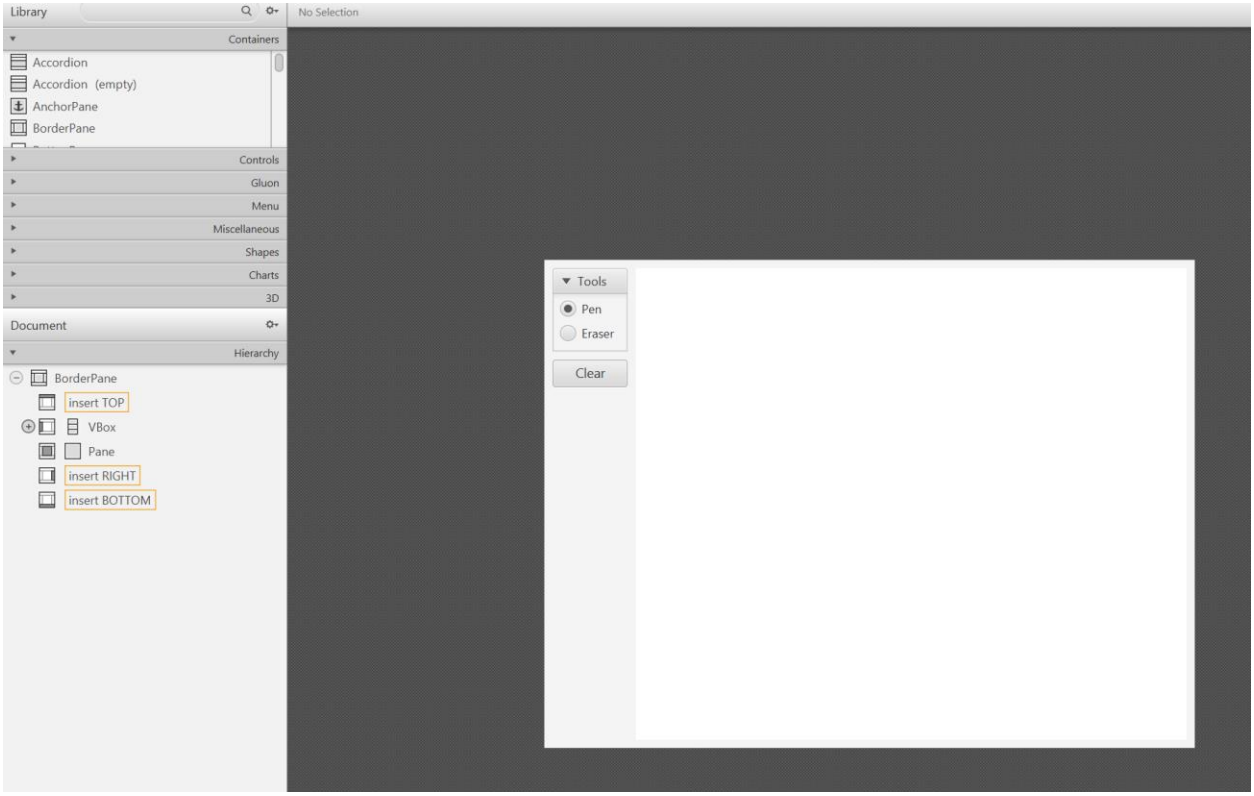
3.2. Creater the Controller Class

```java
package hust.soict.dsai.javafx;
import javafx.event.ActionEvent;

public class PainterController {

    boolean isEraserMode = false;

    @FXML
    private ToggleGroup identical;

    @FXML
    private Pane drawingAreaPane;

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

    @FXML
    void drawingAreaMouseDragged(MouseEvent event) {
        // Check if the target is the drawing area
        if (event.getTarget() == drawingAreaPane) {
            if (isEraserMode) {
                Circle eraser = new Circle(event.getX(), event.getY(), 4, Color.WHITE);
                drawingAreaPane.getChildren().add(eraser);
            } else {
                Circle pen = new Circle(event.getX(), event.getY(), 4, Color.BLACK);
                drawingAreaPane.getChildren().add(pen);
            }
        }
    }

    @FXML
    void penMode(ActionEvent event) {
        isEraserMode = false;
    }

    @FXML
    void eraserMode (ActionEvent event) {
        isEraserMode = true;
    }

}
```

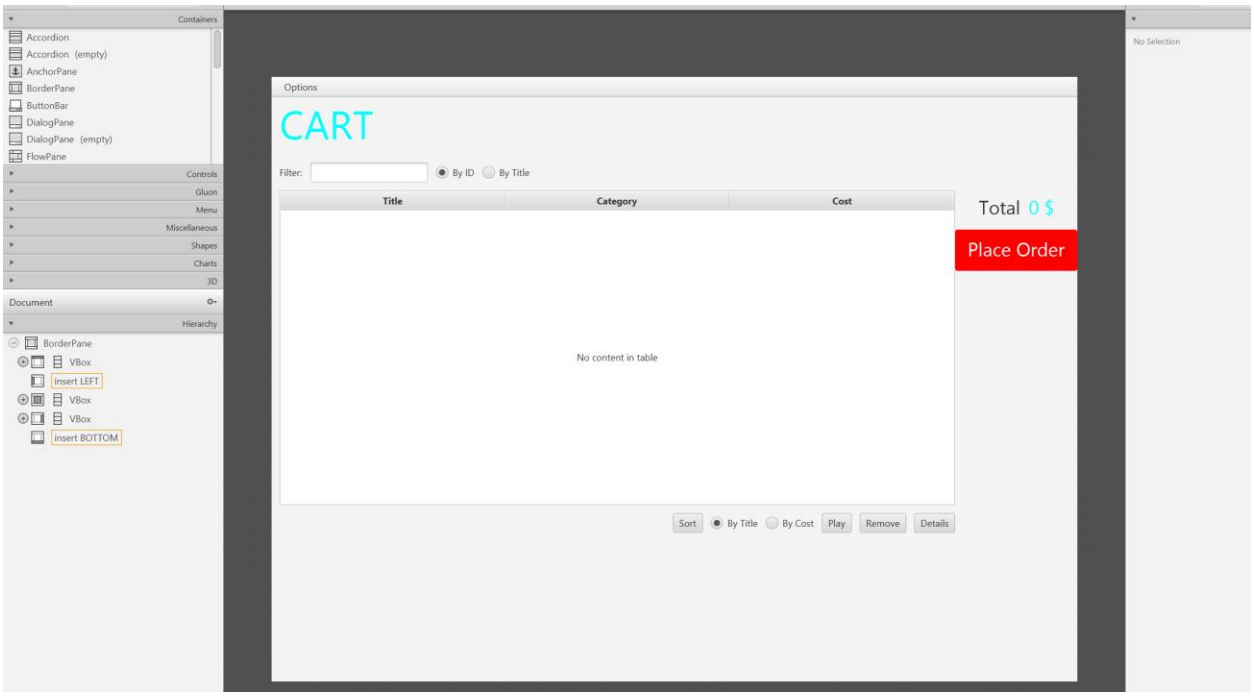
3.3. Create the application

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

import javafx.application.Application;

public class Painter extends Application {

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

        Scene scene = new Scene(root);
        stage.setTitle("Painter");
        stage.setScene(scene);
        stage.show();

    }
    public static void main(String[] args) {
        launch(args);
    }
}
```

3.4 Practice Excersise



4. Setting up the View Cart Screen with ScreenBuilder



5. Integrating JavaFX into Swing Application

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

import java.awt.event.WindowAdapter;

public class CartScreen extends JFrame {
    private Store store;
    private Cart cart;

    public CartScreen(Store store, Cart cart) {
        super();

        this.store = store;
        this.cart = cart;
        this.setSize(1024,768);
        JFXPanel fxPanel = new JFXPanel();
        this.add(fxPanel);

        this.setTitle("Cart");
        this.setVisible(true);
        JFrame frame = this;

        this.addWindowListener(new WindowAdapter() {
            @Override
            public void windowClosing(WindowEvent e) {
                new StoreScreen(store, cart);
                dispose();
            }
        });

        Platform.runLater(new Runnable() {

            @Override
            public void run() {

                try {
                    FXMLLoader loader = new FXMLLoader(
                            getClass().getResource("cart.fxml"));
                    CartScreenController controller = new CartScreenController(store, cart, frame);
                    loader.setController(controller);
                    Parent root = loader.load();
                    fxPanel.setScene(new Scene(root));
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }

        });
    }
```

6. View the items in Cart JavaFX data-driven UI

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

import javax.swing.JFrame;

public class CartScreenController {
    private Store store;
    private Cart cart;
    private boolean filterByID = true;
    private boolean sortByTitle = true;
    private FilteredList<Media> filteredCart;
    private JFrame stage;

    @FXML
    private TableView<Media> tblMedia;

    @FXML
    private TableColumn<Media, String> colMediaTitle;

    @FXML
    private TableColumn<Media, String> colMediaCategory;

    @FXML
    private TableColumn<Media, String> colMediaCost;

    @FXML
    private Button btnPlay;

    @FXML
    private Button btnRemove;

    @FXML
    private Button btnDetails;

    @FXML
    private TextField tfFilter;

    @FXML
    private Label costLabel;


    public CartScreenController(Store store, Cart cart, JFrame stage) {
        super();
        this.store = store;
        this.cart = cart;
        this.stage = stage;
    }

```

```
66
67●     @FXML
68      public void initialize() {
69          filteredCart = new FilteredList<Media>(this.cart.getItemsOrdered(), s -> true);
70
71          colMediaTitle.setCellValueFactory(new PropertyValueFactory<Media, String>("title"));
72          colMediaCategory.setCellValueFactory(new PropertyValueFactory<Media, String>("category"));
73          colMediaCost.setCellValueFactory(new PropertyValueFactory<Media, String>("cost"));
74          tblMedia.setItems(filteredCart);
75
76          btnPlay.setVisible(false);
77          btnRemove.setVisible(false);
78          btnDetails.setVisible(false);
79
80          costLabel.setText(String.valueOf(this.cart.totalCost()));
81
82●          tblMedia.getSelectionModel().selectedItemProperty().addListener(new ChangeListener<Media>() {
83
84●                      @Override
85                      public void changed(ObservableValue<? extends Media> observable, Media oldValue, Media newValue) {
86                          updateButtonBar(newValue);
87                      }
88          });
89
90●          tfFilter.textProperty().addListener(new ChangeListener<String> () {
91
92●              @Override
93              public void changed(ObservableValue<? extends String> observable, String oldValue, String newValue) {
94                  showFilteredMedia(newValue);
95              }
96          });
97
98      }
99
```

7. Updating buttons based on selected item in TableView ChangeListener

```
68      public void initialize() {
69          filteredCart = new FilteredList<Media>(this.cart.getItemsOrdered(), s -> true);
70
71          colMediaTitle.setCellValueFactory(new PropertyValueFactory<Media, String>("title"));
72          colMediaCategory.setCellValueFactory(new PropertyValueFactory<Media, String>("category"));
73          colMediaCost.setCellValueFactory(new PropertyValueFactory<Media, String>("cost"));
74          tblMedia.setItems(filteredCart);
75
76          btnPlay.setVisible(false);
77          btnRemove.setVisible(false);
78          btnDetails.setVisible(false);
79
80          costLabel.setText(String.valueOf(this.cart.totalCost()));
81
82          tblMedia.getSelectionModel().selectedItemProperty().addListener(new ChangeListener<Media>() {
83
84                      @Override
85                      public void changed(ObservableValue<? extends Media> observable, Media oldValue, Media newValue) {
86                          updateButtonBar(newValue);
87                      }
88          });
89
90          tfFilter.textProperty().addListener(new ChangeListener<String> () {
91
92              @Override
93              public void changed(ObservableValue<? extends String> observable, String oldValue, String newValue) {
94                  showFilteredMedia(newValue);
95              }
96          });
97
```

```
99
100     private void updateButtonBar(Media media) {
101         if (media == null) {
102             btnRemove.setVisible(false);
103             btnDetails.setVisible(false);
104             btnPlay.setVisible(false);
105         } else {
106             btnRemove.setVisible(true);
107             btnDetails.setVisible(true);
108             if (media instanceof Playable) {
109                 btnPlay.setVisible(true);
110             } else {
111                 btnPlay.setVisible(false);
112             }
113         }
114     }
115
116     private void showFilteredMedia(String filter) {
117         if (filter == null || filter.length() == 0) {
118             filteredCart.setPredicate(s -> true);
119         } else {
120             if (filterByID) {
121                 try {
122                     filteredCart.setPredicate(s -> s.getID() == Integer.parseInt(filter));
123                 } catch (NumberFormatException e) {}
124             } else {
125                 filteredCart.setPredicate(s -> s.getTitle().toLowerCase().contains(filter));
126             }
127         }
128     }
```

8. Deleting a media

```java
@FXML
private void removeButtonPressed(ActionEvent event) {
    Media media = tblMedia.getSelectionModel().getSelectedItem();
    try {
        this.cart.removeMedia(media);
    } catch (NonExistingItemException e) {
        Alert alert = new Alert(AlertType.ERROR);
        alert.setTitle("Notification");
        alert.setHeaderText("Failed to remove");
        alert.setContentText("Media not in cart");
        alert.showAndWait();
    }
    costLabel.setText(String.valueOf(this.cart.totalCost()));
}
```

9. Filter items in cart FilteredList

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

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

10. Complete the Aims GUI Application

Cart Screen

```java
private void placeOrderPressed(ActionEvent event) {
    if (this.cart.getSize() > 0) {
        Alert alert = new Alert(AlertType.INFORMATION);
        alert.setTitle("Notification");
        alert.setHeaderText("Success!");
        alert.setContentText("Your order has been placed.");
        alert.showAndWait();
        this.cart.empty();
        costLabel.setText(String.valueOf(this.cart.totalCost()));
    } else {
        Alert alert = new Alert(AlertType.ERROR);
        alert.setTitle("Notification");
        alert.setHeaderText("ERROR: Failed to place order.");
        alert.setContentText("Your cart is empty");
        alert.showAndWait();
    }

}
```

```java
@FXML
private void playButtonPressed(ActionEvent event) {
    Media media = this.tblMedia.getSelectionModel().getSelectedItem();
    try {
        ((Playable)media).play();
    } catch (PlayerException e) {
        Alert alert = new Alert(AlertType.WARNING);
        alert.setTitle("Media Player");
        alert.setHeaderText("ERROR: Media length is non-positive.");
        alert.setContentText("Media cannot be played.");
        alert.showAndWait();
    }
}
```

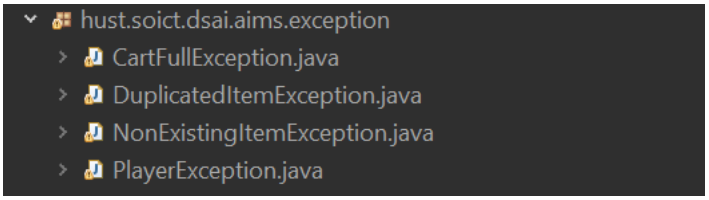
```java
@FXML
private void detailsButtonPressed(ActionEvent event) {
    Media media = this.tblMedia.getSelectionModel().getSelectedItem();
    Alert alert = new Alert(AlertType.INFORMATION);
    alert.setTitle("Detail infomation");
    alert.setHeaderText("Viewing " + media.getTitle() + " detail infomation.");
    alert.setContentText(media.getDetails());
    alert.showAndWait();
}
```

Store Screen

Update Store Screen

```java
private class AddDVDListener implements ActionListener {

    @Override
    public void actionPerformed(ActionEvent e) {
        new AddDVDToStoreScreen(store, cart);
        dispose();
    }
}

private class AddBookListener implements ActionListener {

    @Override
    public void actionPerformed(ActionEvent e) {
        new AddBookToStoreScreen(store, cart);
        dispose();
    }
}

private class AddCDListener implements ActionListener {

    @Override
    public void actionPerformed(ActionEvent e) {
        new AddCDToStoreScreen(store, cart);
        dispose();
    }
```

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




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

public class PlayerException extends Exception {

    public PlayerException() {
        // TODO Auto-generated constructor stub
    }

    public PlayerException(String message) {
        super(message);
        // TODO Auto-generated constructor stub
    }

    public PlayerException(Throwable cause) {
        super(cause);
        // TODO Auto-generated constructor stub
    }

    public PlayerException(String message, Throwable cause) {
        super(message, cause);
        // TODO Auto-generated constructor stub
    }

    public PlayerException(String message, Throwable cause, boolean enableSuppression, boolean writableStackTrace) {
        super(message, cause, enableSuppression, writableStackTrace);
        // TODO Auto-generated constructor stub
    }

}
```