ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÁO CÁO THỰC HÀNH
IT3103-744527-2024.1

BÀI THỰC HÀNH -LAB05

Họ và tên sv: Vương Quốc Huy

MSSV: 20225637

Lớp: Việt Nhật 07-K67

GVHD: Lê Thị Hoa

HTGD: Đặng Mạnh Cường

Hà Nội 12/2024

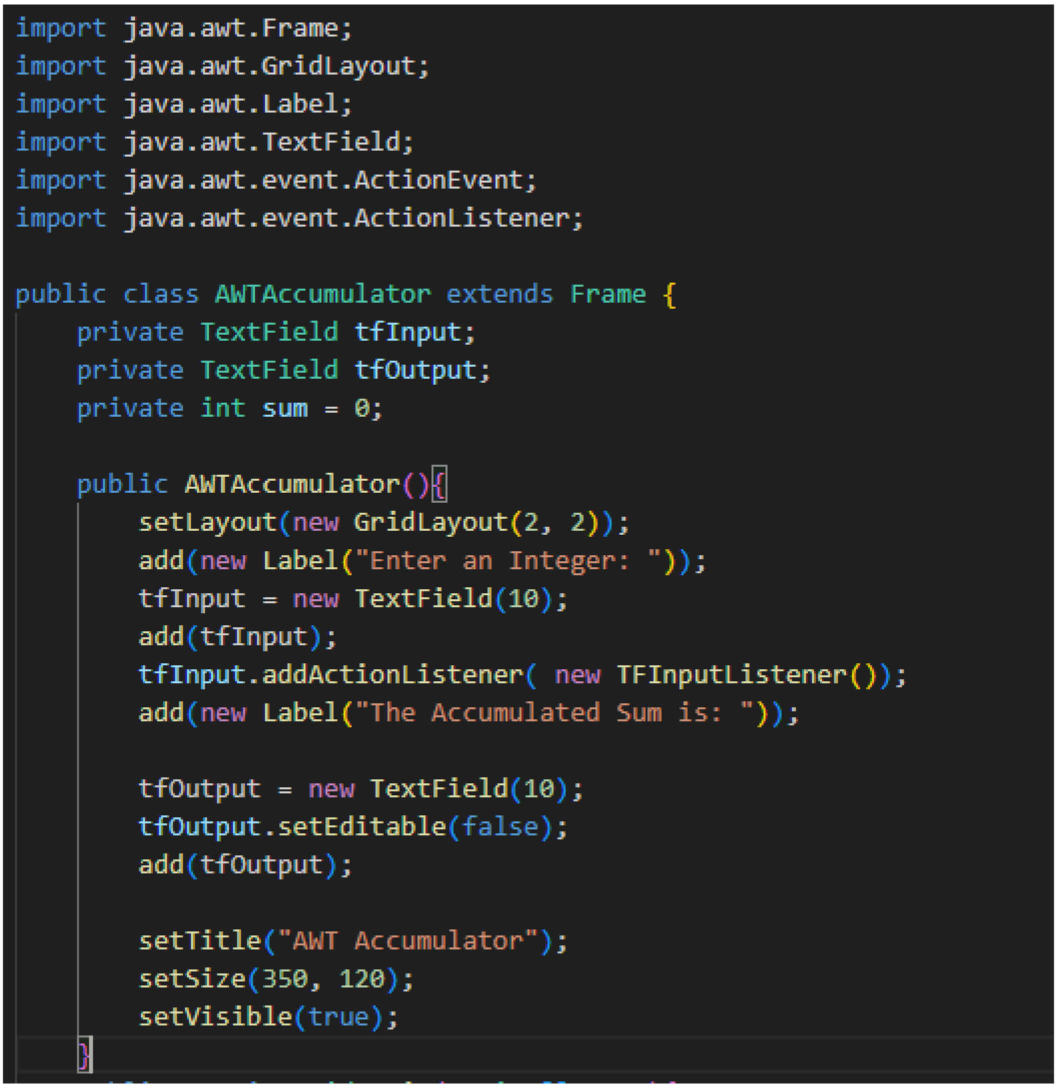
BÁO CÁO THỰC HÀNH LAB 5

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

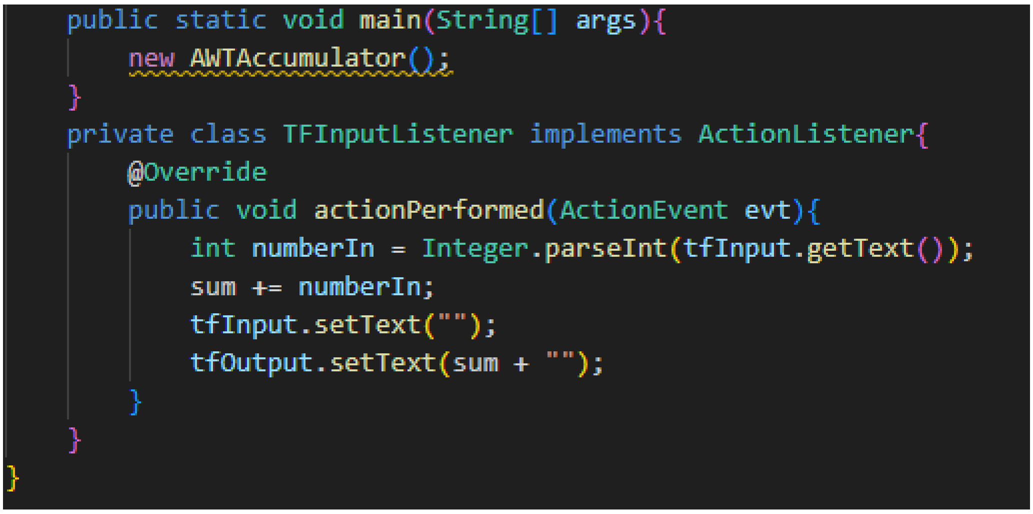
Table of Contents

1. Swing components

1.1 AWTAccumulator Class

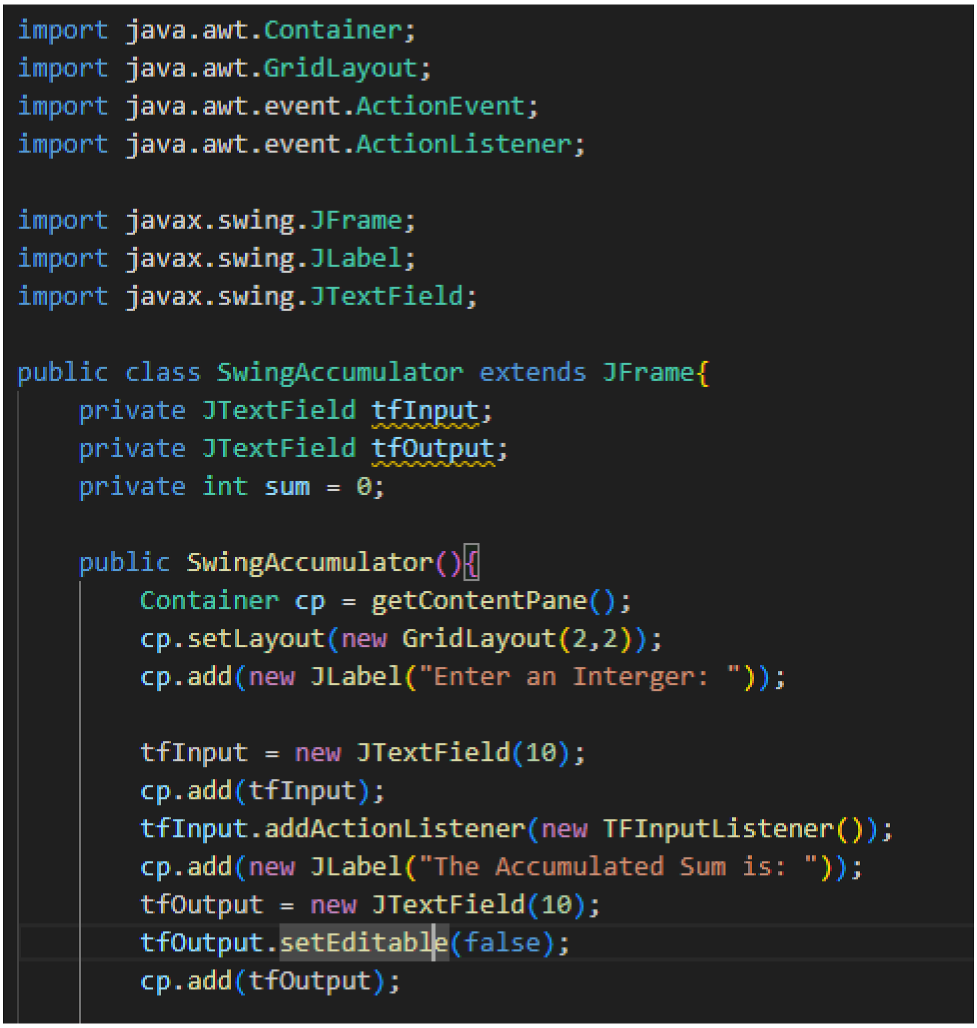


```java
import java.awt.Frame;
import java.awt.GridLayout;
import java.awt.Label;
import java.awt.TextField;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

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 Integer: "));
        tfInput = new TextField(10);
        add(tfInput);
        tfInput.addActionListener( new TFInputListener());
        add(new Label("The Accumulated Sum is: "));

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

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

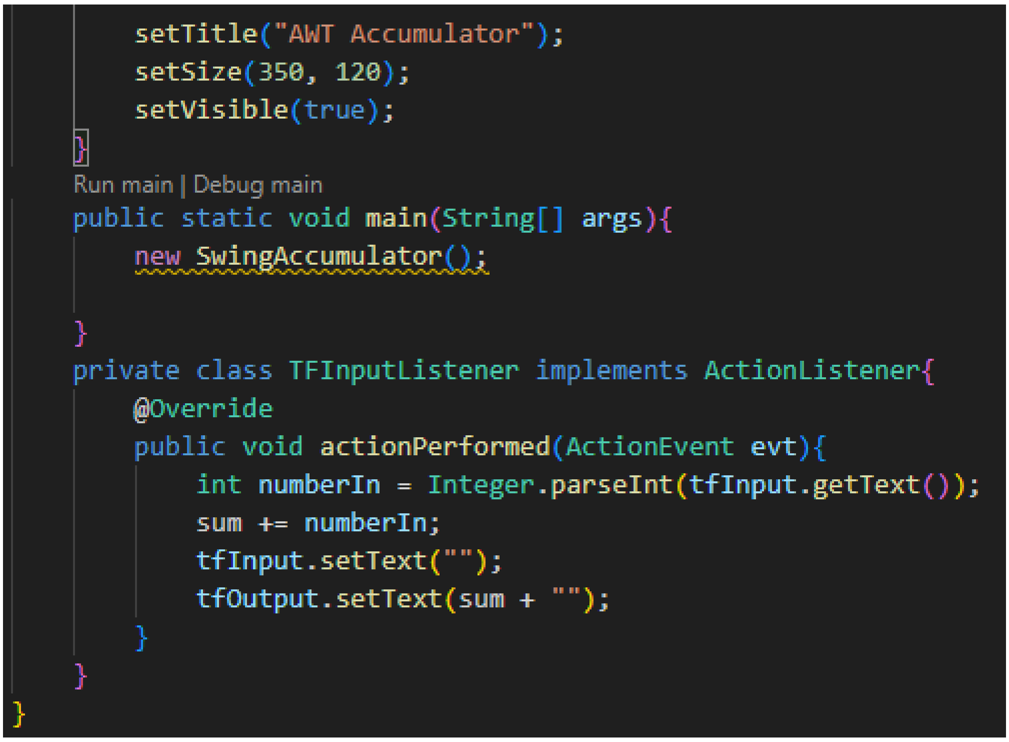


```java
    public static void main(String[] args){
        new AWTAccumulator();
    }
    private class TFInputListener implements ActionListener{
        @Override
        public void actionPerformed(ActionEvent evt){
            int numberIn = Integer.parseInt(tfInput.getText());
            sum += numberIn;
            tfInput.setText("");
            tfOutput.setText(sum + "");
        }
    }
}
```

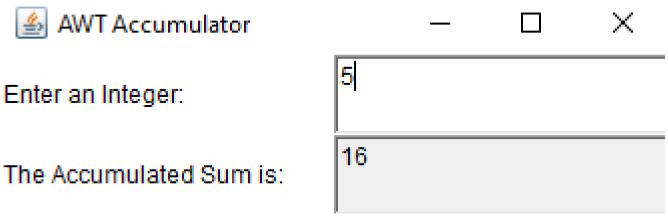
1.2 SwingAccumulator Class



```java
import java.awt.Container;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextField;

public class SwingAccumulator extends JFrame{
    private JTextField tfInput;
    private JTextField tfOutput;
    private int sum = 0;

    public SwingAccumulator(){
        Container cp = getContentPane();
        cp.setLayout(new GridLayout(2,2));
        cp.add(new JLabel("Enter an Interger: "));

        tfInput = new JTextField(10);
        cp.add(tfInput);
        tfInput.addActionListener(new TFInputListener());
        cp.add(new JLabel("The Accumulated Sum is: "));
        tfOutput = new JTextField(10);
        tfOutput.setEditable(false);
        cp.add(tfOutput);
```

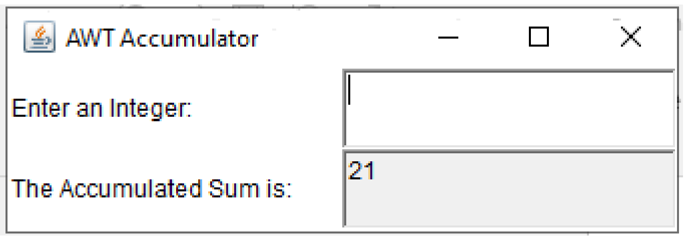


```java
        setTitle("AWT Accumulator");
        setSize(350, 120);
        setVisible(true);
    }
    Run main | Debug main
    public static void main(String[] args){
        new SwingAccumulator();

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



Enter an Integer: 5

The Accumulated Sum is: 16

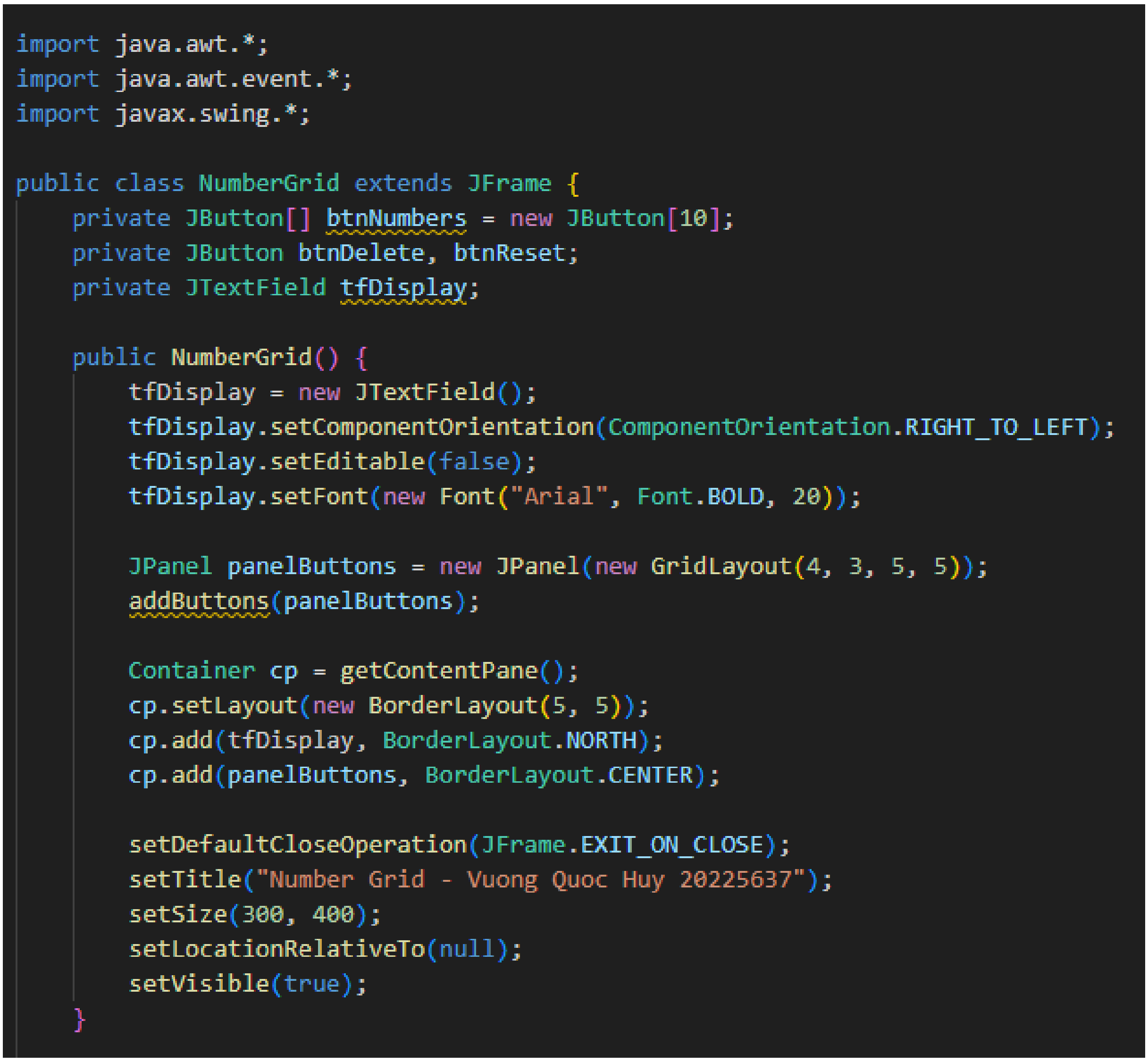


Enter an Integer:

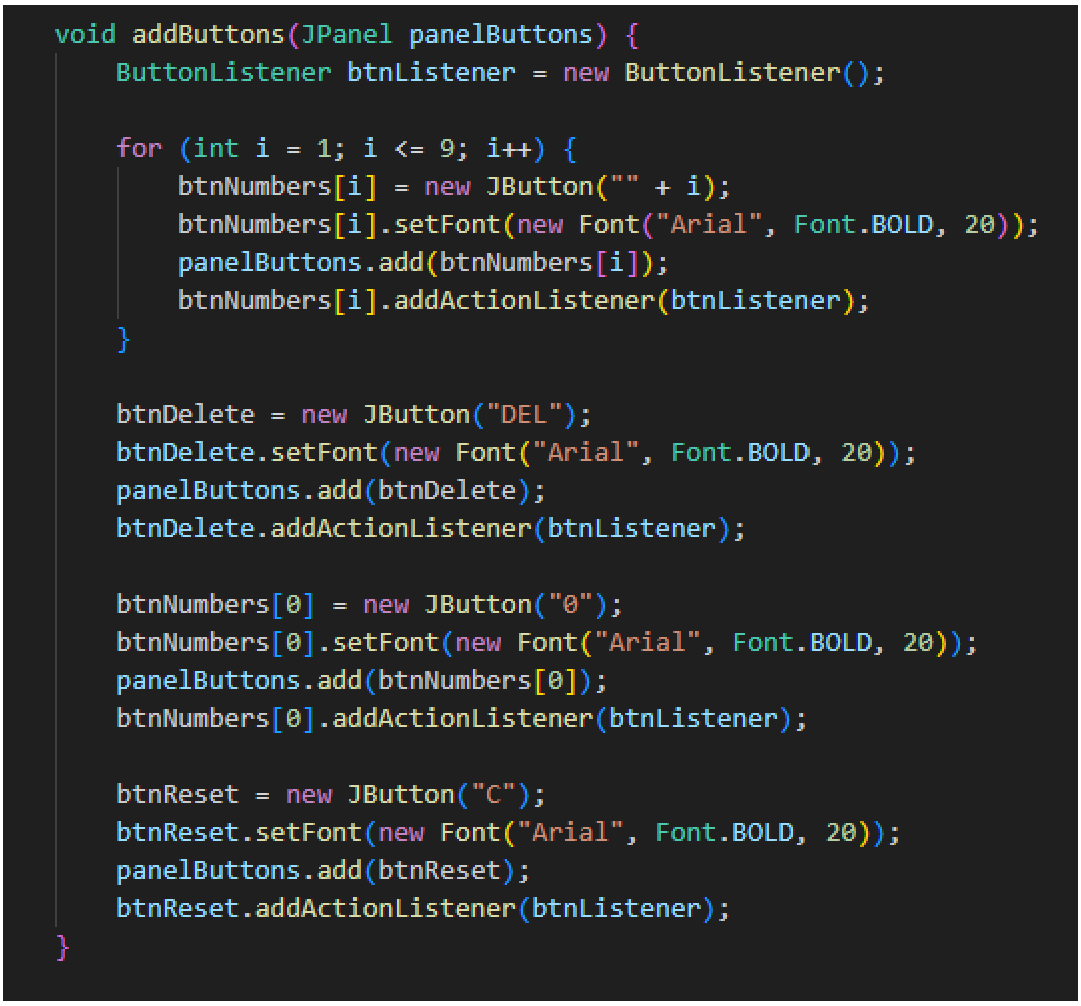
The Accumulated Sum is: 21

2. Organizing Swing components with Layout Managers

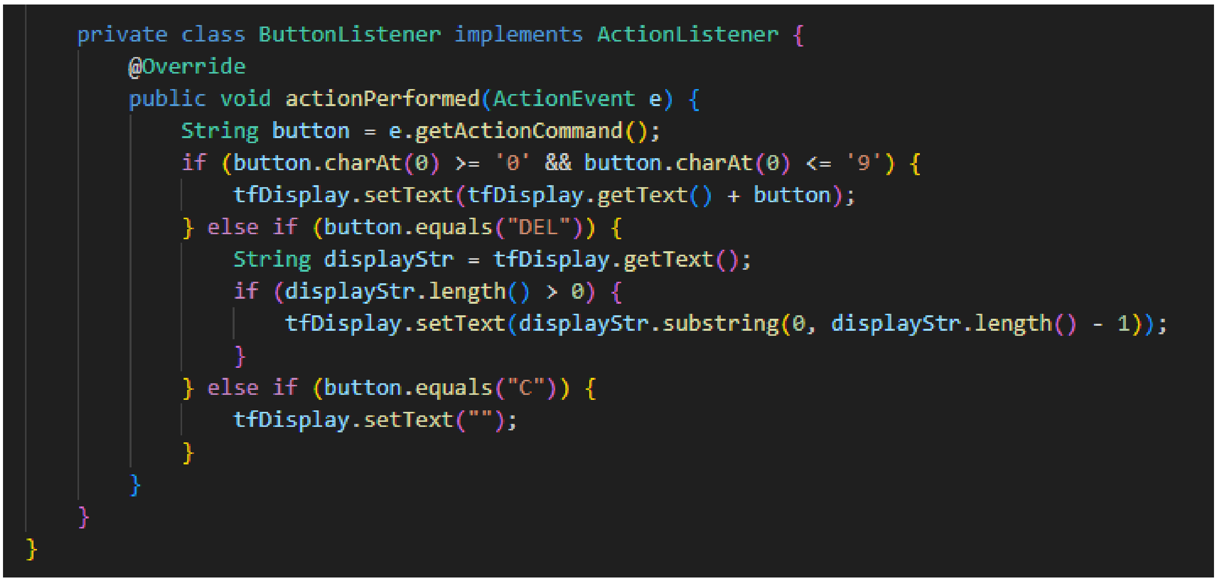
2.1 NumberGrid Class



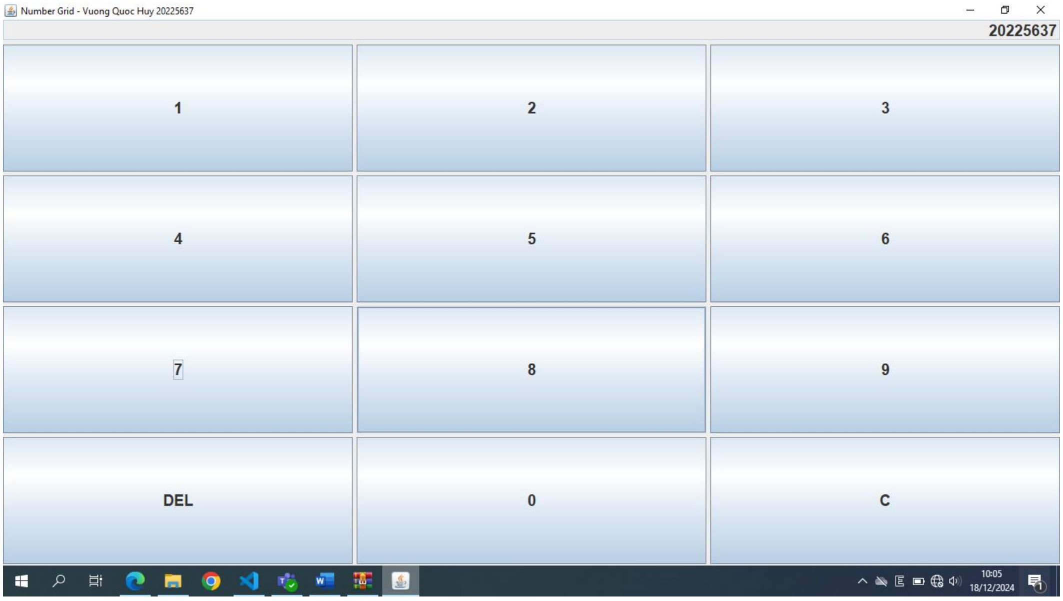
```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class NumberGrid extends JFrame {
    private JButton[] btnNumbers = new JButton[10];
    private JButton btnDelete, btnReset;
    private JTextField tfDisplay;

    public NumberGrid() {
        tfDisplay = new JTextField();
        tfDisplay.setComponentOrientation(ComponentOrientation.RIGHT_TO_LEFT);
        tfDisplay.setEditable(false);
        tfDisplay.setFont(new Font("Arial", Font.BOLD, 20));

        JPanel panelButtons = new JPanel(new GridLayout(4, 3, 5, 5));
        addButtons(panelButtons);

        Container cp = getContentPane();
        cp.setLayout(new BorderLayout(5, 5));
        cp.add(tfDisplay, BorderLayout.NORTH);
        cp.add(panelButtons, BorderLayout.CENTER);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setTitle("Number Grid - Vuong Quoc Huy 20225637");
        setSize(300, 400);
        setLocationRelativeTo(null);
        setVisible(true);
    }
```



```java
void addButtons(JPanel panelButtons) {
    ButtonListener btnListener = new ButtonListener();

    for (int i = 1; i <= 9; i++) {
        btnNumbers[i] = new JButton("" + i);
        btnNumbers[i].setFont(new Font("Arial", Font.BOLD, 20));
        panelButtons.add(btnNumbers[i]);
        btnNumbers[i].addActionListener(btnListener);
    }

    btnDelete = new JButton("DEL");
    btnDelete.setFont(new Font("Arial", Font.BOLD, 20));
    panelButtons.add(btnDelete);
    btnDelete.addActionListener(btnListener);

    btnNumbers[0] = new JButton("0");
    btnNumbers[0].setFont(new Font("Arial", Font.BOLD, 20));
    panelButtons.add(btnNumbers[0]);
    btnNumbers[0].addActionListener(btnListener);

    btnReset = new JButton("C");
    btnReset.setFont(new Font("Arial", Font.BOLD, 20));
    panelButtons.add(btnReset);
    btnReset.addActionListener(btnListener);
}
```

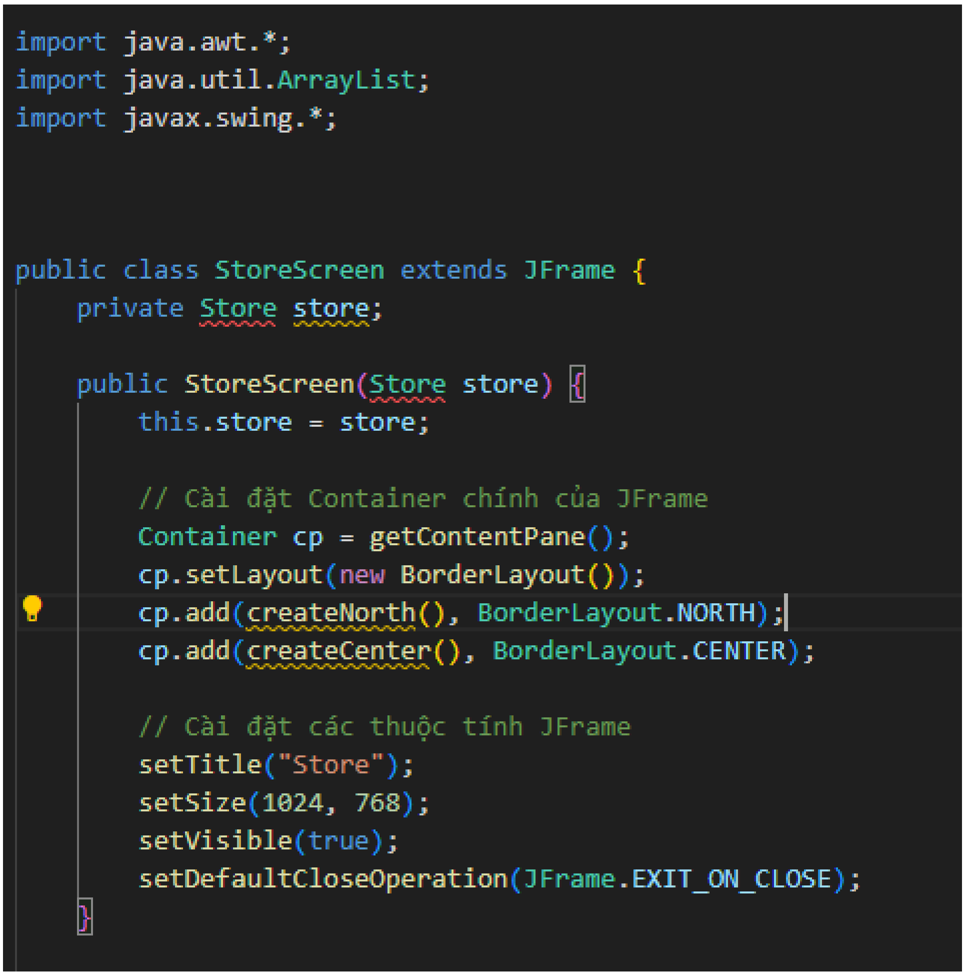


```java
    private class ButtonListener implements ActionListener {
        @Override
        public void actionPerformed(ActionEvent e) {
            String button = e.getActionCommand();
            if (button.charAt(0) >= '0' && button.charAt(0) <= '9') {
                tfDisplay.setText(tfDisplay.getText()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

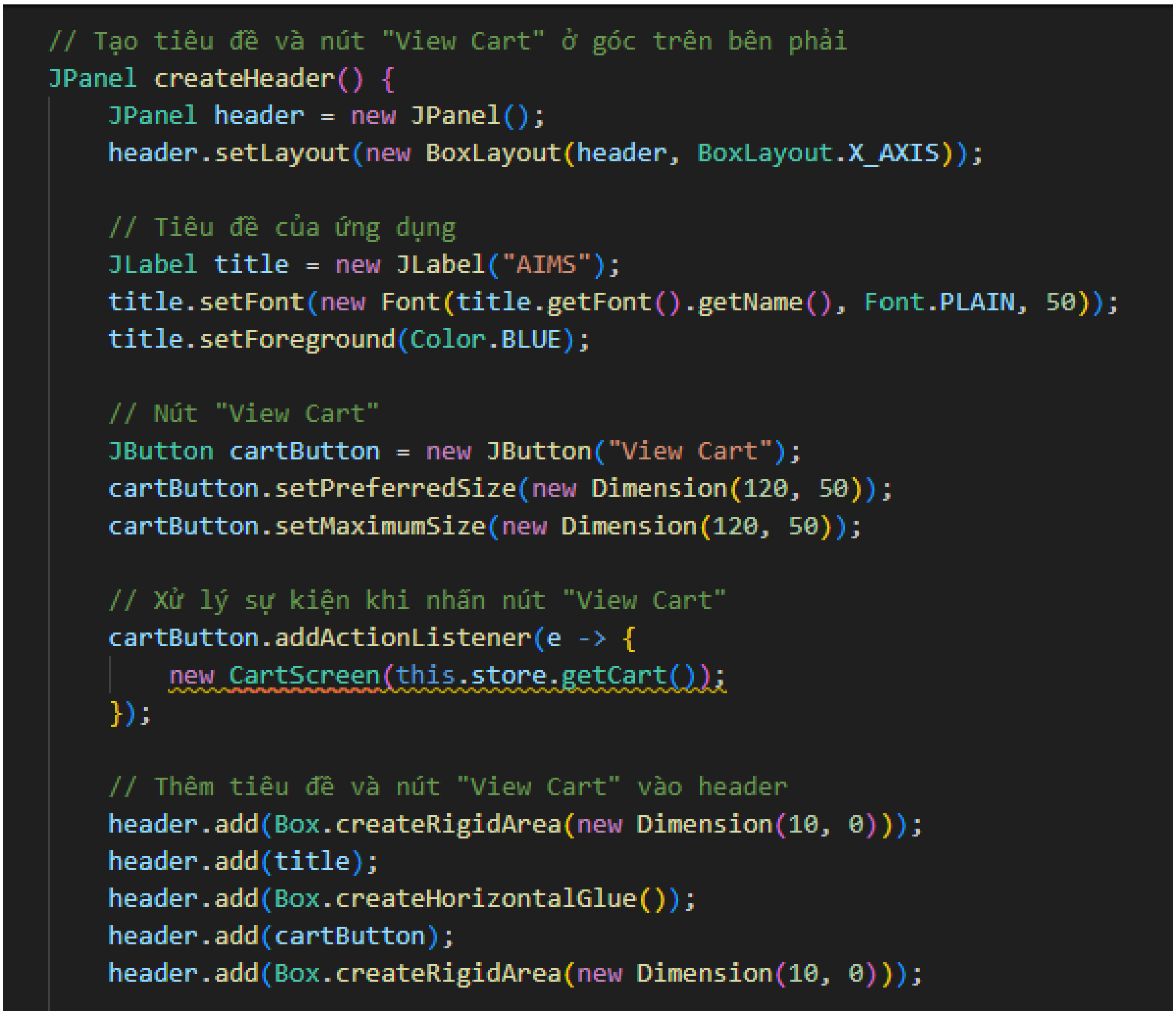


3. Create a graphical user interface for AIMS with Swing

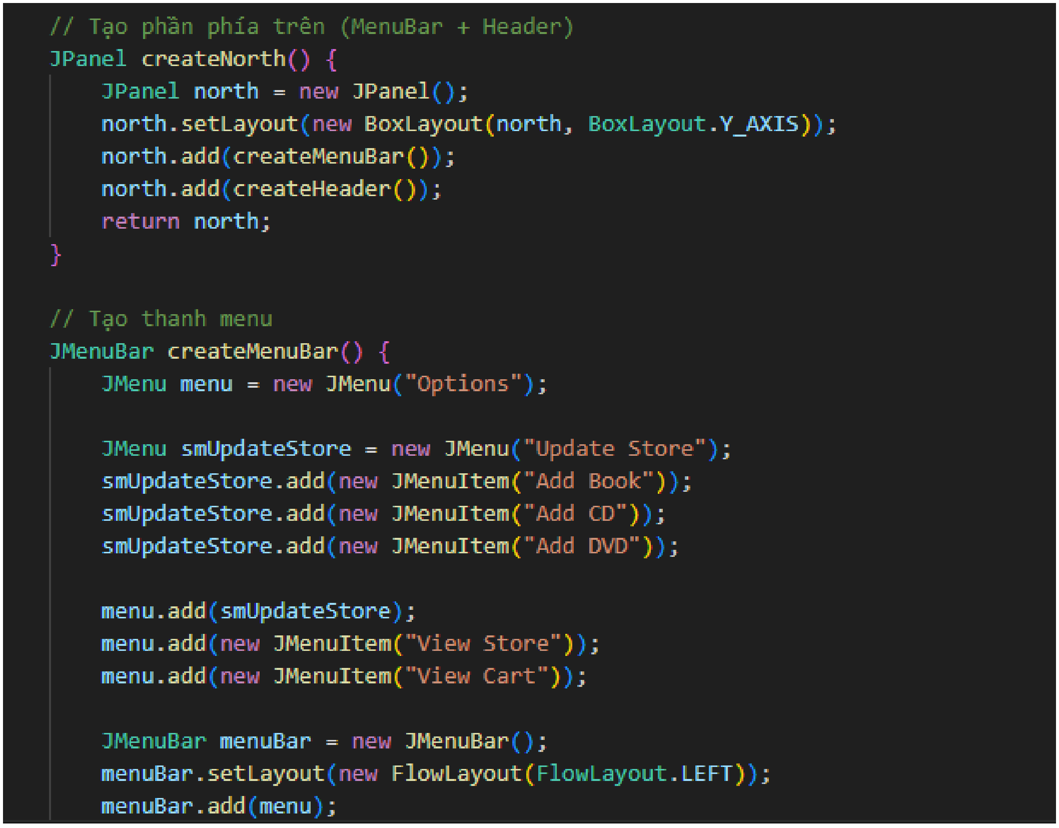
3.1 View Store Screen



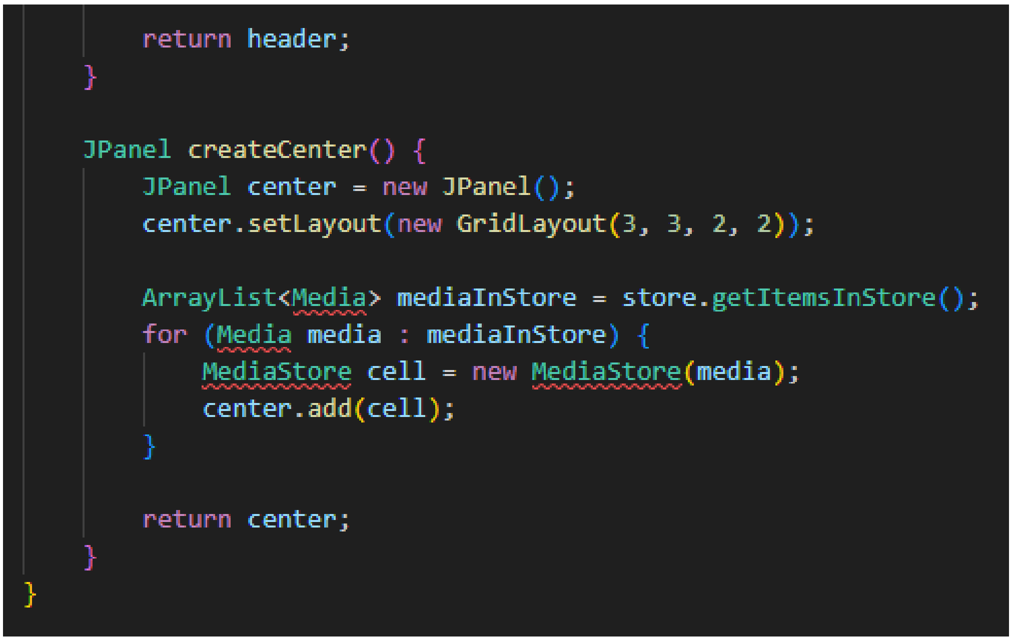
```
import java.awt.*;
import java.util.ArrayList;
import javax.swing.*;



public class StoreScreen extends JFrame {
    private Store store;

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

        // Cài đặt Container chính của JFrame
        Container cp = getContentPane();
        cp.setLayout(new BorderLayout());
        cp.add(createNorth(), BorderLayout.NORTH);
        cp.add(createCenter(), BorderLayout.CENTER);

        // Cài đặt các thuộc tính JFrame
        setTitle("Store");
        setSize(1024, 768);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
```



```java
// Tạo tiêu đề và nút "View Cart" ở góc trên bên phải
JPanel createHeader() {
    JPanel header = new JPanel();
    header.setLayout(new BoxLayout(header, BoxLayout.X_AXIS));

    // Tiêu đề của ứng dụng
    JLabel title = new JLabel("AIMS");
    title.setFont(new Font(title.getFont().getName(), Font.PLAIN, 50));
    title.setForeground(Color.BLUE);

    // Nút "View Cart"
    JButton cartButton = new JButton("View Cart");
    cartButton.setPreferredSize(new Dimension(120, 50));
    cartButton.setMaximumSize(new Dimension(120, 50));

    // Xử lý sự kiện khi nhấn nút "View Cart"
    cartButton.addActionListener(e -> {
        new CartScreen(this.store.getCart());
    });

    // Thêm tiêu đề và nút "View Cart" vào header
    header.add(Box.createRigidArea(new Dimension(10, 0)));
    header.add(title);
    header.add(Box.createHorizontalGlue());
    header.add(cartButton);
    header.add(Box.createRigidArea(new Dimension(10, 0)));
```

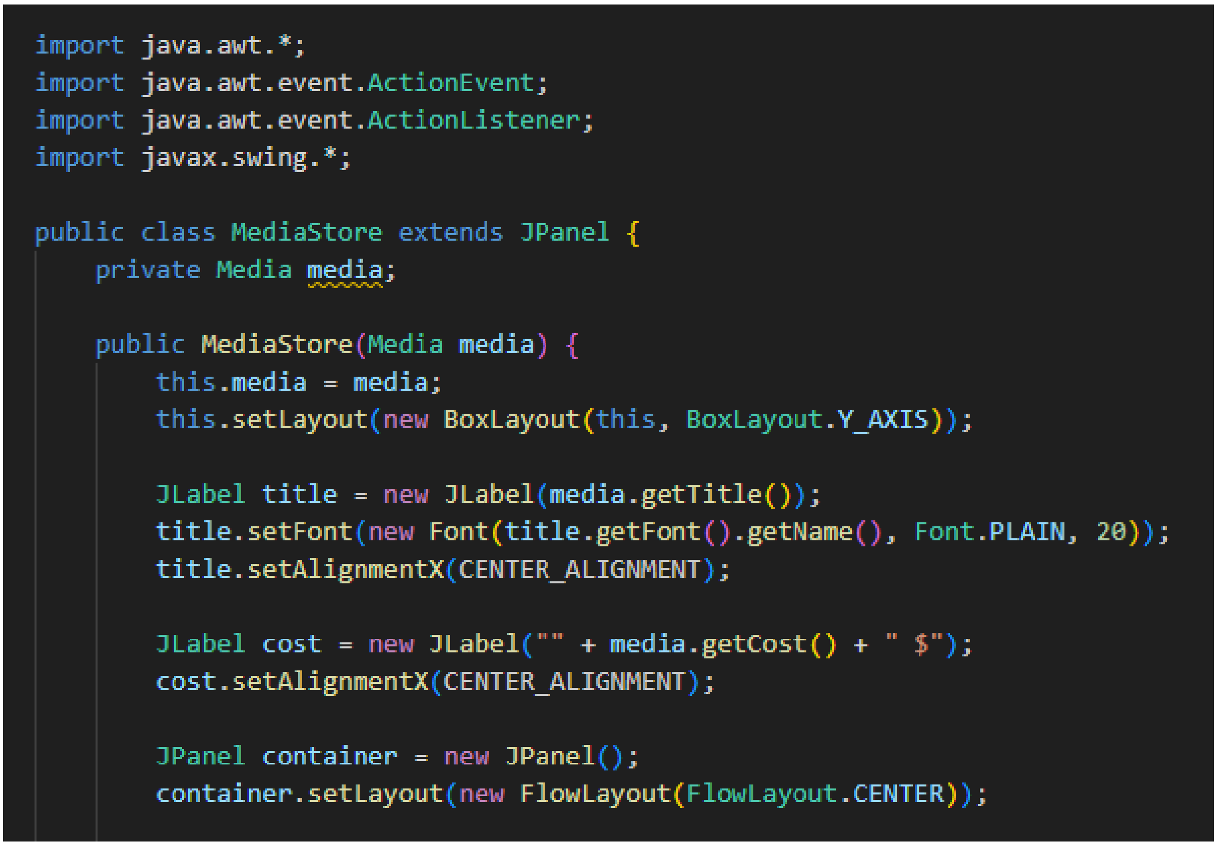


```java
// Tạo phần phía trên (MenuBar + Header)
JPanel createNorth() {
    JPanel north = new JPanel();
    north.setLayout(new BoxLayout(north, BoxLayout.Y_AXIS));
    north.add(createMenuBar());
    north.add(createHeader());
    return north;
}

// Tạo thanh menu
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);
```

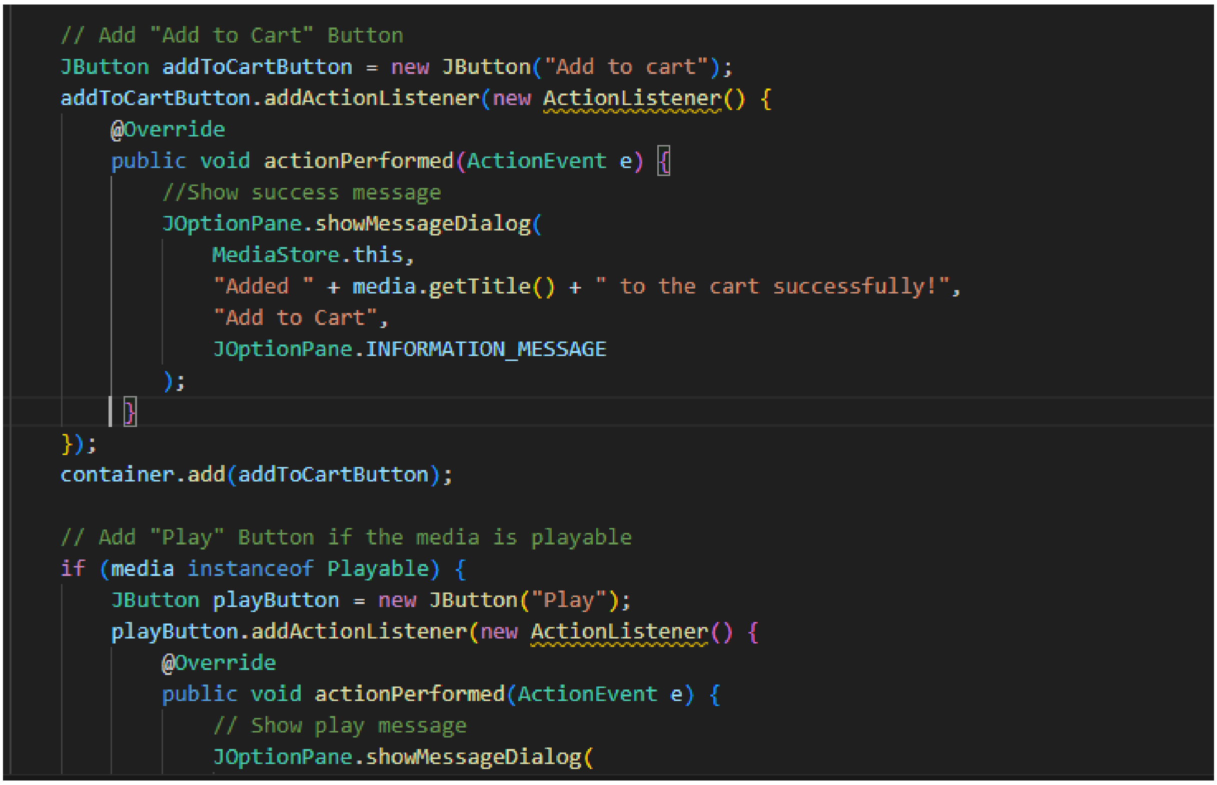


```java
        return header;
    }

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

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

        return center;
    }
}
```

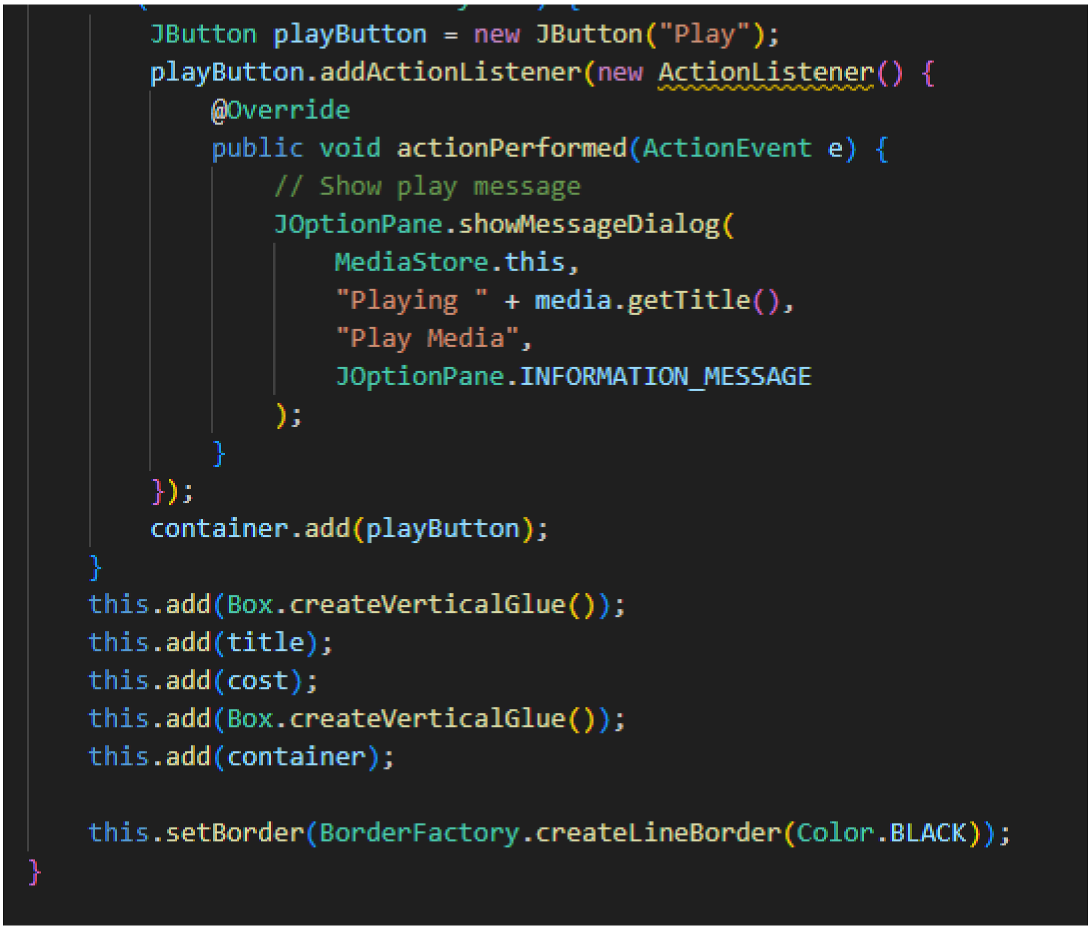
3.2 MediaStore



```java
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class MediaStore extends JPanel {
    private Media media;

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

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

        JLabel cost = new JLabel("" + media.getCost() + " $");
        cost.setAlignmentX(CENTER_ALIGNMENT);

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



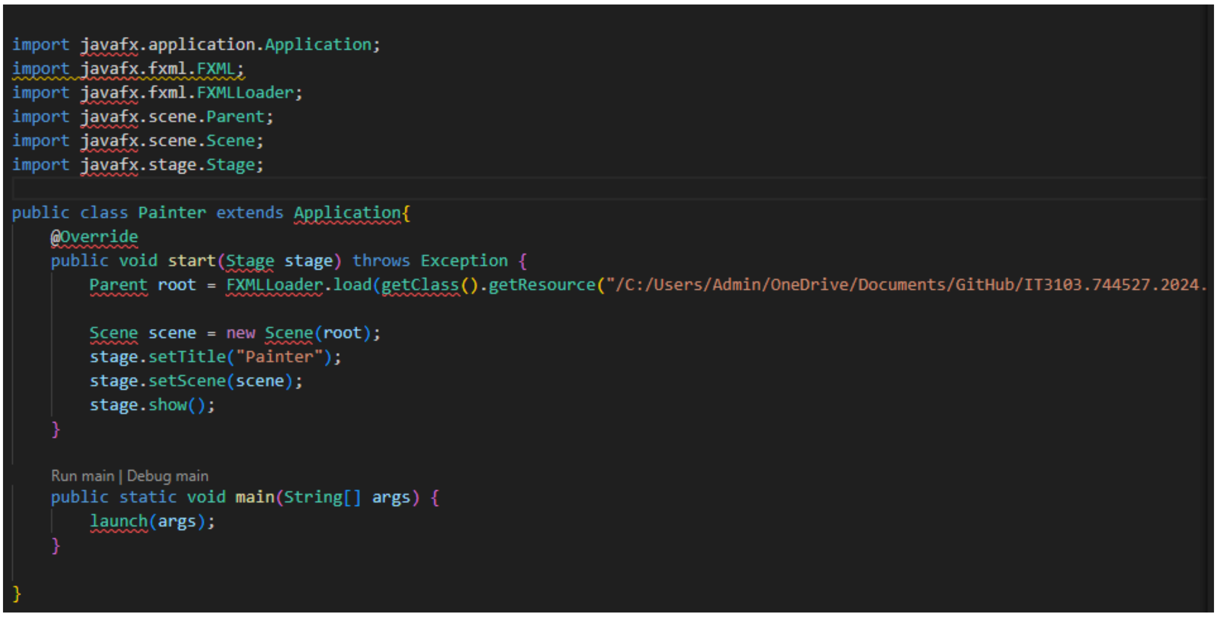
```
    // Add "Add to Cart" Button
    JButton addToCartButton = new JButton("Add to cart");
    addToCartButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            //Show success message
            JOptionPane.showMessageDialog(
                MediaStore.this,
                "Added " + media.getTitle() + " to the cart successfully!",
                "Add to Cart",
                JOptionPane.INFORMATION_MESSAGE
            );
        }
    });
    container.add(addToCartButton);

    // Add "Play" Button if the media is playable
    if (media instanceof Playable) {
        JButton playButton = new JButton("Play");
        playButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // Show play message
                JOptionPane.showMessageDialog(
```



```
        JButton playButton = new JButton("Play");
        playButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // Show play message
                JOptionPane.showMessageDialog(
                    MediaStore.this,
                    "Playing " + media.getTitle(),
                    "Play Media",
                    JOptionPane.INFORMATION_MESSAGE
                );
            }
        });
        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));
}
```

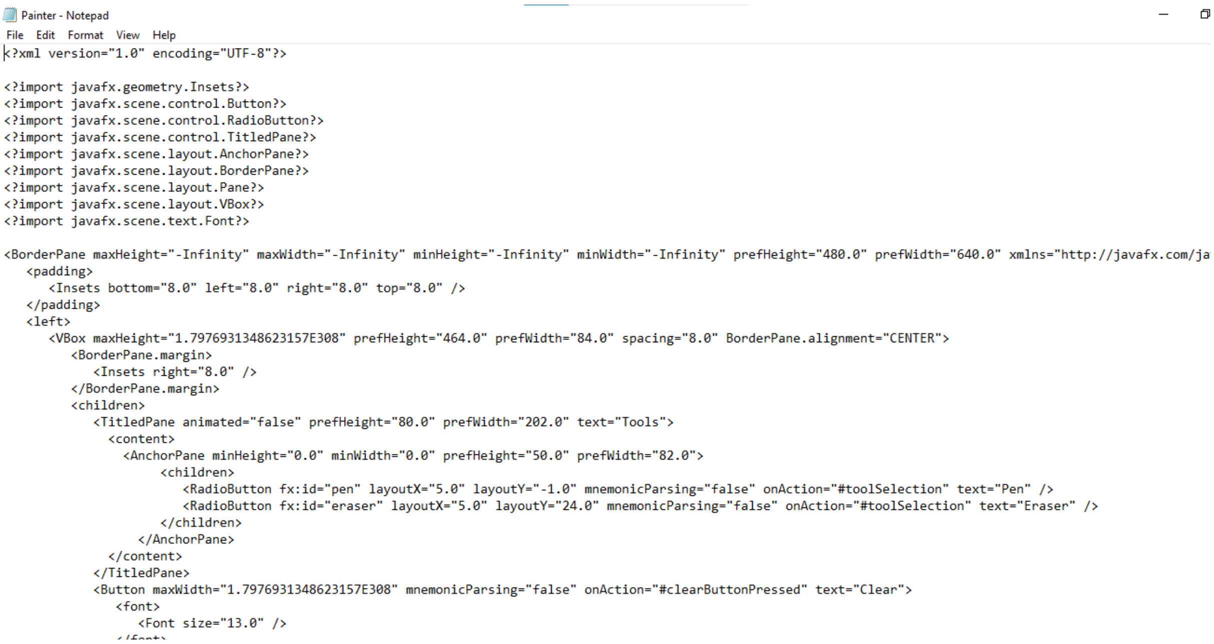
4. JavaFX API

4.1 Create class Painter



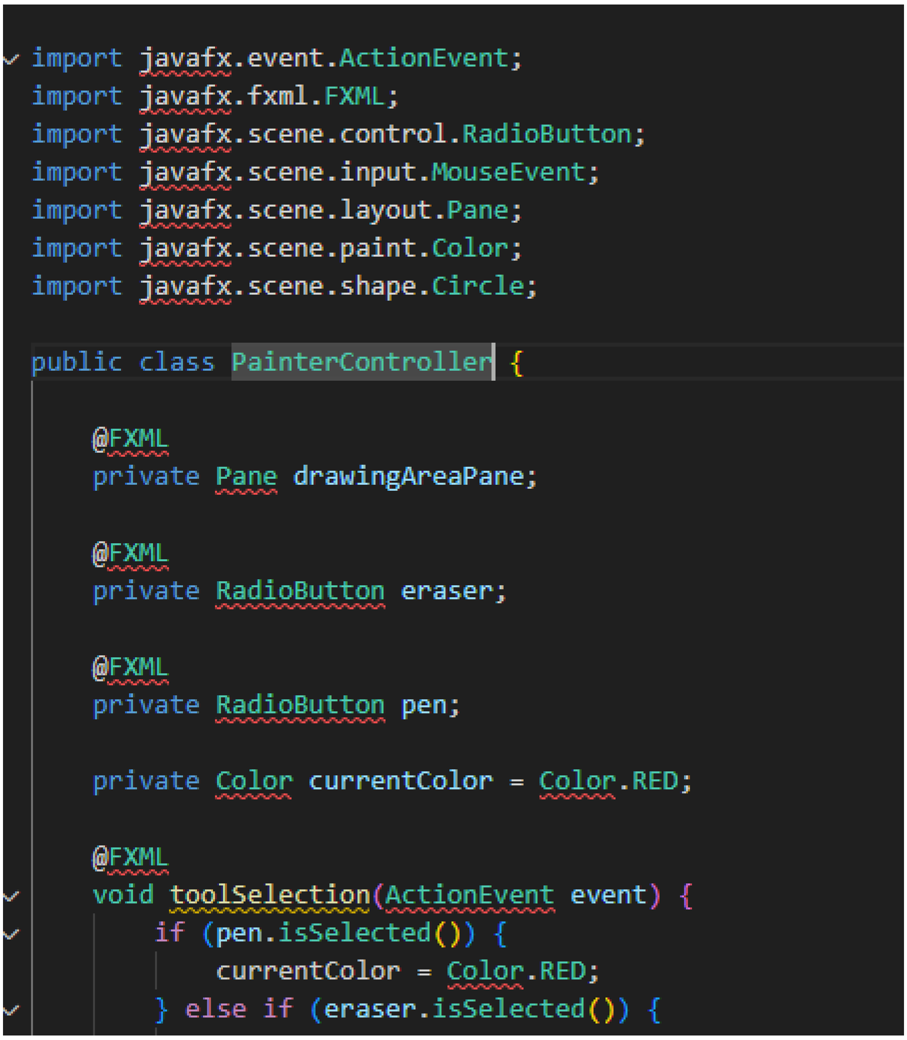
```
import javafx.application.Application;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class Painter extends Application{
    @Override
    public void start(Stage stage) throws Exception {
        Parent root = FXMLLoader.load(getClass().getResource("/C:/Users/Admin/OneDrive/Documents/GitHub/IT3103.744527.2024.

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

    Run main | Debug main
    public static void main(String[] args) {
        launch(args);
    }

}
```

4.2 Create Painter.fxml

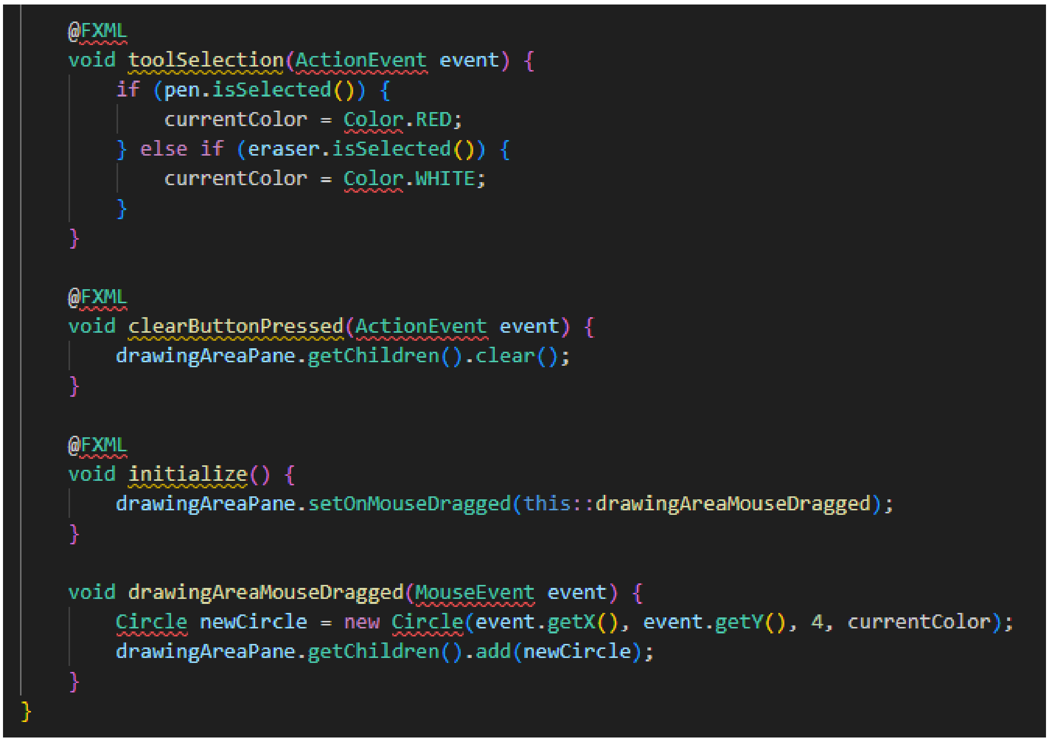


```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.RadioButton?>
<?import javafx.scene.control.TitledPane?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.Pane?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>

<BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="480.0" prefWidth="640.0" xmlns="http://javafx.com/ja
   <padding>
      <Insets bottom="8.0" left="8.0" right="8.0" top="8.0" />
   </padding>
   <left>
      <VBox maxHeight="1.7976931348623157E308" prefHeight="464.0" prefWidth="84.0" spacing="8.0" BorderPane.alignment="CENTER">
         <BorderPane.margin>
            <Insets right="8.0" />
         </BorderPane.margin>
         <children>
            <TitledPane animated="false" prefHeight="80.0" prefWidth="202.0" text="Tools">
              <content>
                <AnchorPane minHeight="0.0" minWidth="0.0" prefHeight="50.0" prefWidth="82.0">
                     <children>
                        <RadioButton fx:id="pen" layoutX="5.0" layoutY="-1.0" mnemonicParsing="false" onAction="#toolSelection" text="Pen" />
                        <RadioButton fx:id="eraser" layoutX="5.0" layoutY="24.0" mnemonicParsing="false" onAction="#toolSelection" text="Eraser" />
                     </children>
                  </AnchorPane>
              </content>
            </TitledPane>
            <Button maxWidth="1.7976931348623157E308" mnemonicParsing="false" onAction="#clearButtonPressed" text="Clear">
               <font>
                  <Font size="13.0" />
               </font>
```

4.3 Create class PainterController



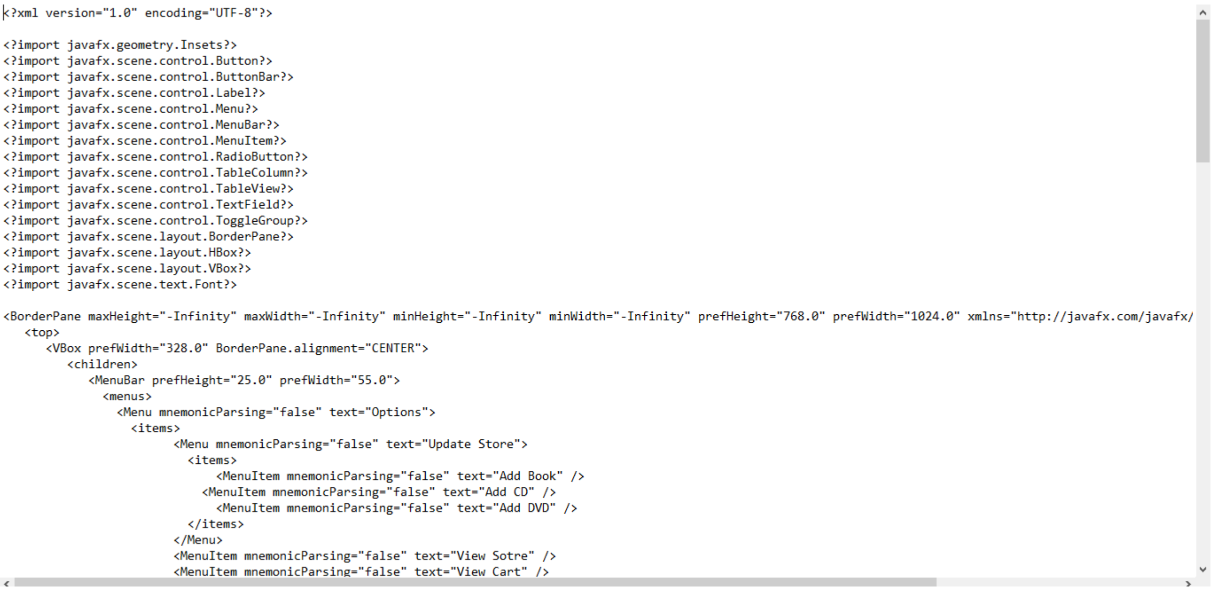
```java
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.RadioButton;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;

public class PainterController {

    @FXML
    private Pane drawingAreaPane;

    @FXML
    private RadioButton eraser;

    @FXML
    private RadioButton pen;

    private Color currentColor = Color.RED;

    @FXML
    void toolSelection(ActionEvent event) {
        if (pen.isSelected()) {
            currentColor = Color.RED;
        } else if (eraser.isSelected()) {
```



```java
    @FXML
    void toolSelection(ActionEvent event) {
        if (pen.isSelected()) {
            currentColor = Color.RED;
        } else if (eraser.isSelected()) {
            currentColor = Color.WHITE;
        }
    }

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

    @FXML
    void initialize() {
        drawingAreaPane.setOnMouseDragged(this::drawingAreaMouseDragged);
    }

    void drawingAreaMouseDragged(MouseEvent event) {
        Circle newCircle = new Circle(event.getX(), event.getY(), 4, currentColor);
        drawingAreaPane.getChildren().add(newCircle);
    }
}
```

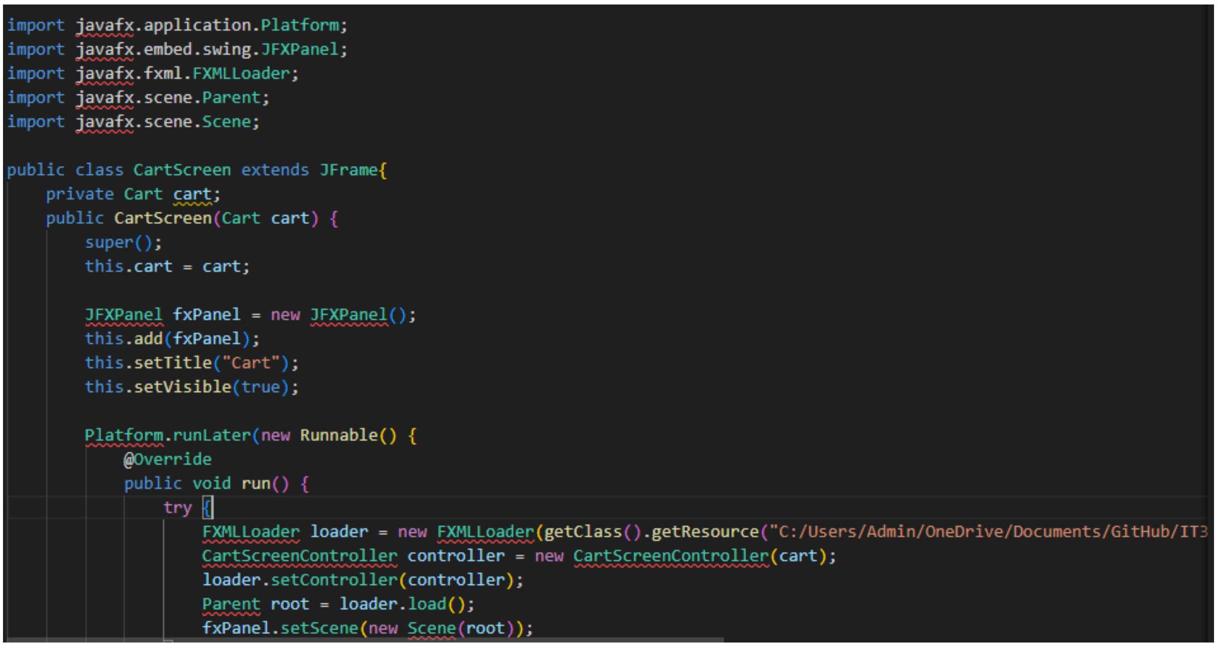
5. View Cart Screen

5.1 Create Cart.fxml

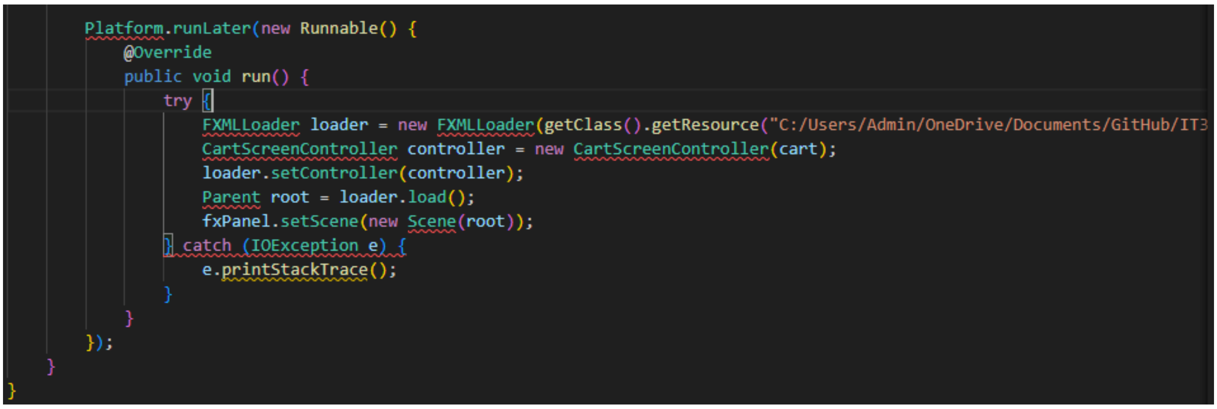


```xml
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.ButtonBar?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.Menu?>
<?import javafx.scene.control.MenuBar?>
<?import javafx.scene.control.MenuItem?>
<?import javafx.scene.control.RadioButton?>
<?import javafx.scene.control.TableColumn?>
<?import javafx.scene.control.TableView?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.control.ToggleGroup?>
<?import javafx.scene.layout.BorderPane?>
<?import javafx.scene.layout.HBox?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>

<BorderPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="768.0" prefWidth="1024.0" xmlns="http://javafx.com/javafx/
   <top>
      <VBox prefWidth="328.0" BorderPane.alignment="CENTER">
         <children>
            <MenuBar prefHeight="25.0" prefWidth="55.0">
              <menus>
                <Menu mnemonicParsing="false" text="Options">
                  <items>
                        <Menu mnemonicParsing="false" text="Update Store">
                          <items>
                              <MenuItem mnemonicParsing="false" text="Add Book" />
                            <MenuItem mnemonicParsing="false" text="Add CD" />
                              <MenuItem mnemonicParsing="false" text="Add DVD" />
                          </items>
                        </Menu>
                        <MenuItem mnemonicParsing="false" text="View Sotre" />
                        <MenuItem mnemonicParsing="false" text="View Cart" />
```

5.2 Create class CartScreen

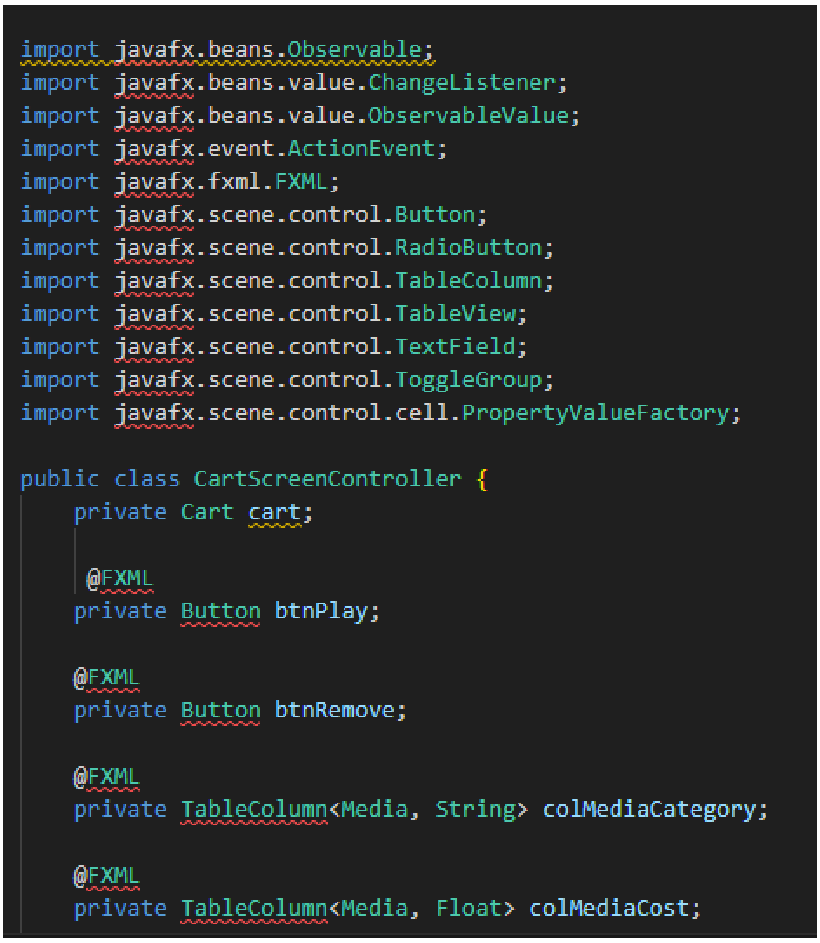


```java
import javafx.application.Platform;
import javafx.embed.swing.JFXPanel;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;

public class CartScreen extends JFrame{
    private Cart cart;
    public CartScreen(Cart cart) {
        super();
        this.cart = cart;

        JFXPanel fxPanel = new JFXPanel();
        this.add(fxPanel);
        this.setTitle("Cart");
        this.setVisible(true);

        Platform.runLater(new Runnable() {
            @Override
            public void run() {
                try {
                    FXMLLoader loader = new FXMLLoader(getClass().getResource("C:/Users/Admin/OneDrive/Documents/GitHub/IT3
                    CartScreenController controller = new CartScreenController(cart);
                    loader.setController(controller);
                    Parent root = loader.load();
                    fxPanel.setScene(new Scene(root));
```

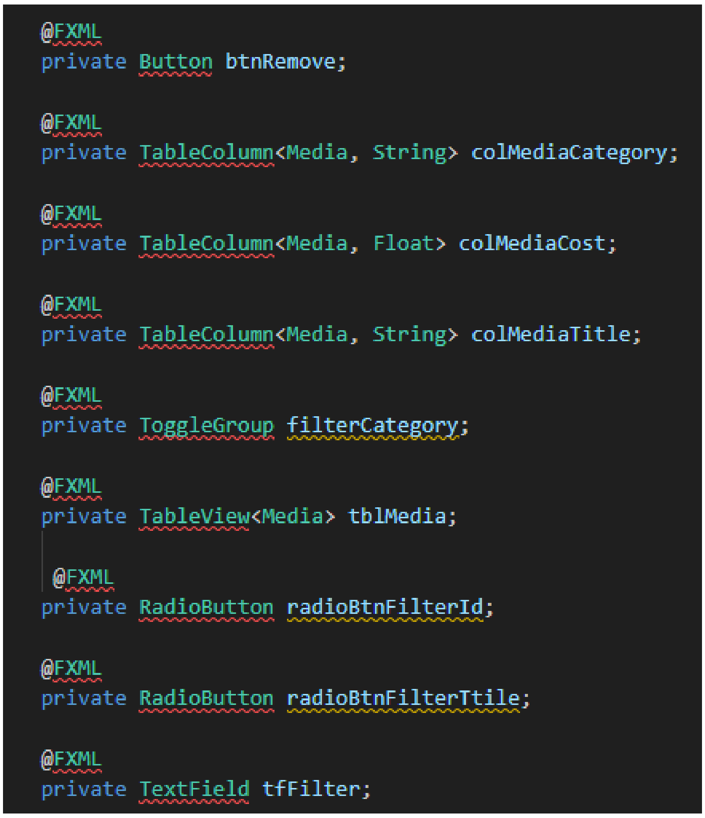


```java
        Platform.runLater(new Runnable() {
            @Override
            public void run() {
                try {
                    FXMLLoader loader = new FXMLLoader(getClass().getResource("C:/Users/Admin/OneDrive/Documents/GitHub/IT3
                    CartScreenController controller = new CartScreenController(cart);
                    loader.setController(controller);
                    Parent root = loader.load();
                    fxPanel.setScene(new Scene(root));
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

5.3 Create class CartScreenController



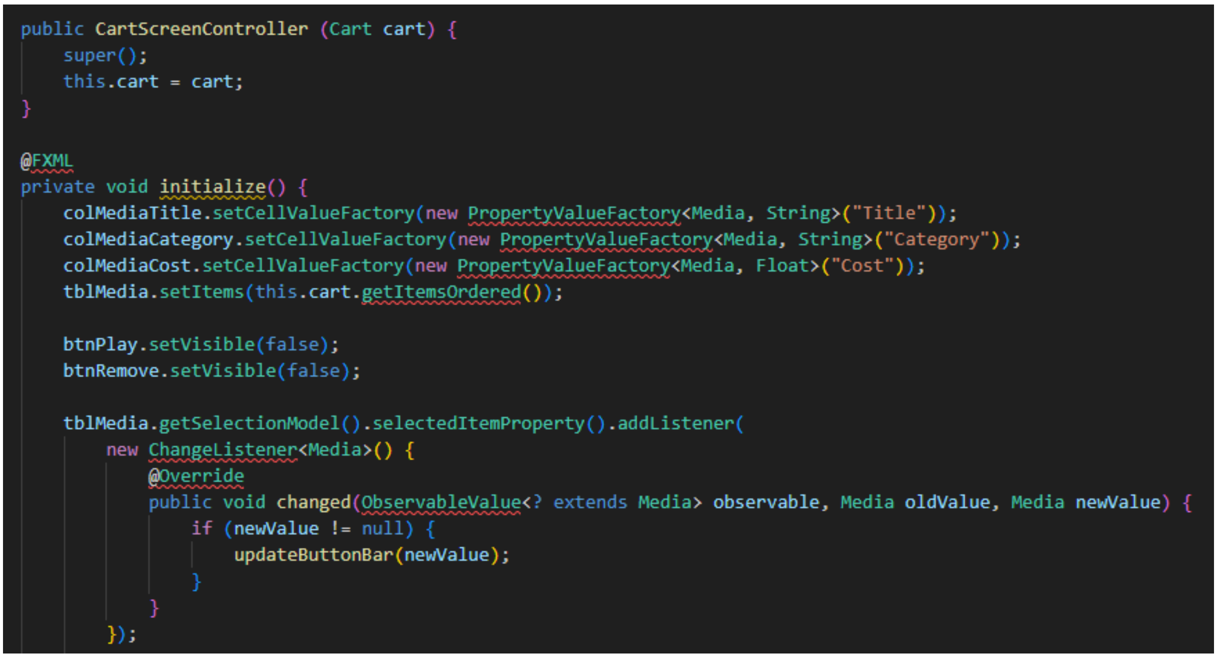
```java
import javafx.beans.Observable;
import javafx.beans.value.ChangeListener;
import javafx.beans.value.ObservableValue;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Button;
import javafx.scene.control.RadioButton;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.TextField;
import javafx.scene.control.ToggleGroup;
import javafx.scene.control.cell.PropertyValueFactory;

public class CartScreenController {
    private Cart cart;

     @FXML
    private Button btnPlay;

    @FXML
    private Button btnRemove;

    @FXML
    private TableColumn<Media, String> colMediaCategory;

    @FXML
    private TableColumn<Media, Float> colMediaCost;
```



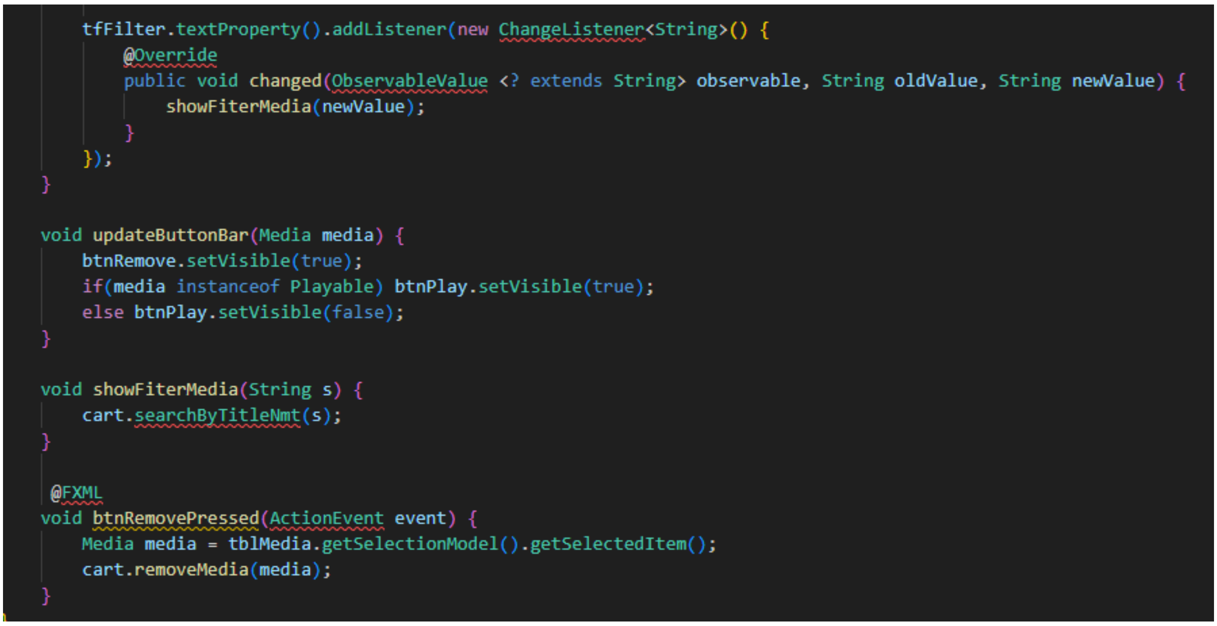
```java
    @FXML
    private Button btnRemove;

    @FXML
    private TableColumn<Media, String> colMediaCategory;

    @FXML
    private TableColumn<Media, Float> colMediaCost;

    @FXML
    private TableColumn<Media, String> colMediaTitle;

    @FXML
    private ToggleGroup filterCategory;

    @FXML
    private TableView<Media> tblMedia;

     @FXML
    private RadioButton radioBtnFilterId;

    @FXML
    private RadioButton radioBtnFilterTtile;

    @FXML
    private TextField tfFilter;
```

6. Updating buttons based on selected item in TableView – ChangeListener

6.1 Edit class CartScreenController

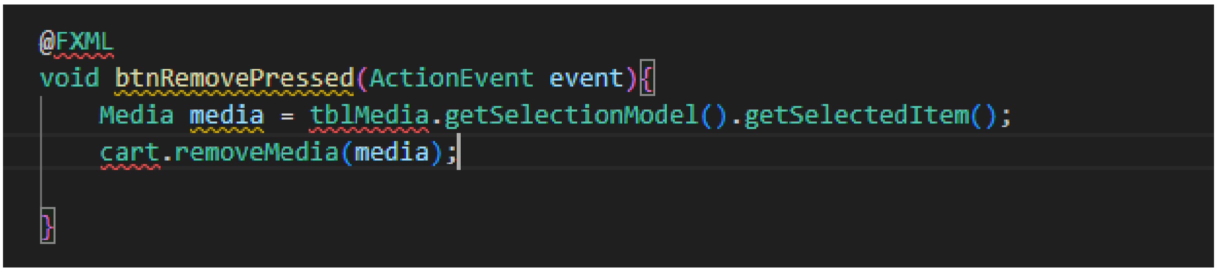


```java
public CartScreenController (Cart cart) {
    super();
    this.cart = cart;
}

@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());

    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);
                }
            }
        });
```



```java
        tfFilter.textProperty().addListener(new ChangeListener<String>() {
            @Override
            public void changed(ObservableValue <? extends String> observable, String oldValue, String newValue) {
                showFiterMedia(newValue);
            }
        });
    }

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

    void showFiterMedia(String s) {
        cart.searchByTitleNmt(s);
    }

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

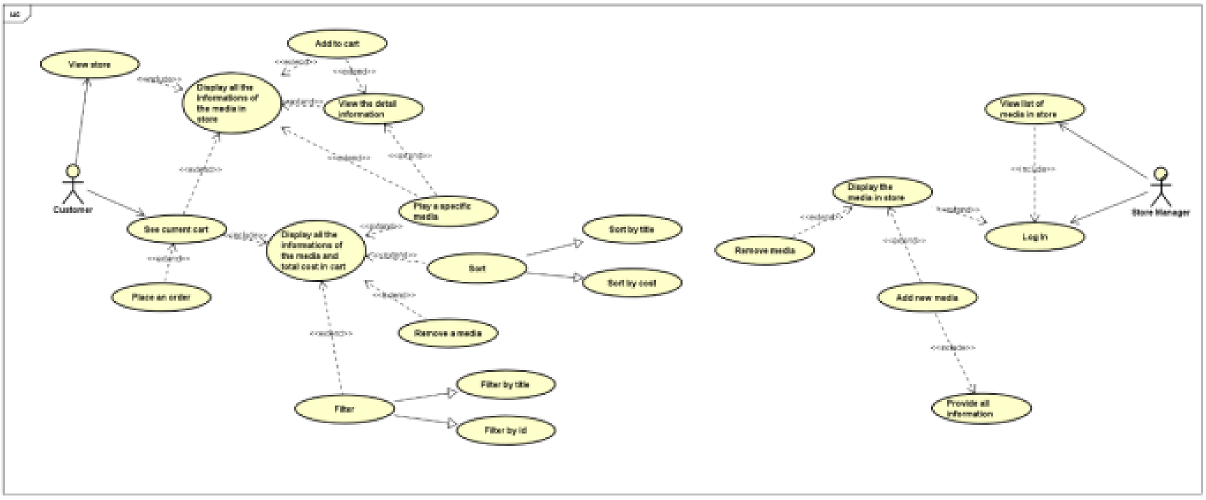
7. Deleting a media



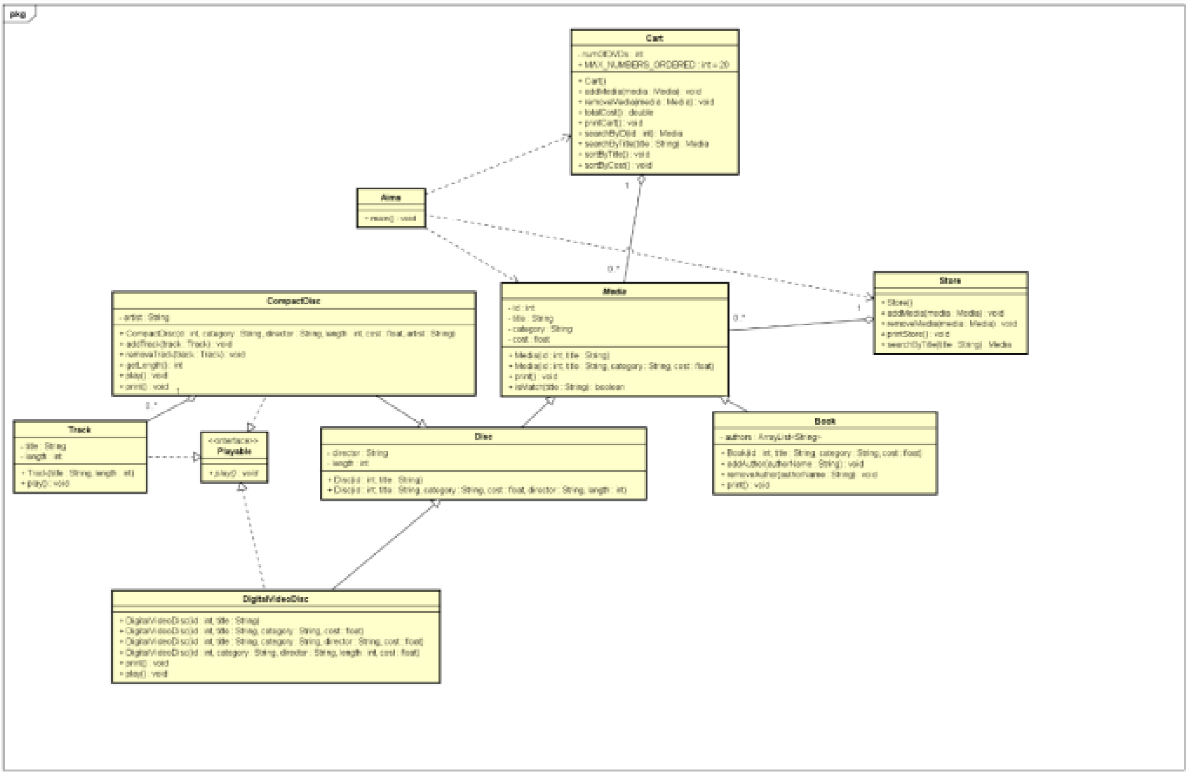
```
@FXML
void btnRemovePressed(ActionEvent event){
    Media media = tblMedia.getSelectionModel().getSelectedItem();
    cart.removeMedia(media);

}
```

8. Complete the Aims GUI application

9. Usecase diagram



10. Class diagram


pkg
Cart
- numOfDVDs : int
+ MAX_NUMBERS_ORDERED : int = 20
+ Cart()
+ addMedia(media : Media) : void
+ removeMedia(media : Media) : void
+ totalCost() : double
+ printCart() : void
+ searchByID(id : int) : Media
+ searchByTitle(title : String) : Media
+ sortByTitle() : void
+ sortByCost() : void
Aims
+ main() : void
1
0..*
Store
+ Store()
+ addMedia(media : Media) : void
+ removeMedia(media : Media) : void
+ printStore() : void
+ searchByTitle(title : String) : Media
1
0..*
Media
- id : int
- title : String
- category : String
- cost : float
+ Media(id : int, title : String)
+ Media(id : int, title : String, category : String, cost : float)
+ print() : void
+ isMatch(title : String) : boolean
CompactDisc
- artist : String
+ CompactDisc(id : int, category : String, director : String, length : int, cost : float, artist : String)
+ addTrack(track : Track) : void
+ removeTrack(track : Track) : void
+ getLength() : int
+ play() : void
+ print() : void
1
1..*
Track
- title : String
- length : int
+ Track(title : String, length : int)
+ play() : void
<<interface>>
Playable
+ play() : void
Disc
- director : String
- length : int
+ Disc(id : int, title : String)
+ Disc(id : int, title : String, category : String, cost : float, director : String, length : int)
Book
- authors : ArrayList<String>
+ Book(id : int, title : String, category : String, cost : float)
+ addAuthor(authorName : String) : void
+ removeAuthor(authorName : String) : void
+ print() : void
DigitalVideoDisc
+ DigitalVideoDisc(id : int, title : String)
+ DigitalVideoDisc(id : int, title : String, category : String, cost : float)
+ DigitalVideoDisc(id : int, title : String, category : String, director : String, cost : float)
+ DigitalVideoDisc(id : int, category : String, director : String, length : int, cost : float)
+ print() : void
+ play() : void