UNIVERSITY OF WESTMINSTER

INFORMATICS INSTITUTE OF TECHNOLOGY

Informatics Institute of Technology
Department of Computing
BEng (Hons) Software Engineering

Programing Principles 2

COURSEWORK 2

Name : Dinura Sonal
Student ID : 2018238 UOW no : w1714958
Group C

interFace.java

Code

```
package sample;

import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.stage.Stage;
import javafx.scene.control.Button;
import javafx.scene.control.TextField;
import javafx.scene.control.Label;
import javafx.scene.layout.Pane;
import javafx.scene.layout.StackPane;

import java.awt.*;
import java.io.IOException;

import javafx.scene.control.Button;

import java.io.IOException;
import java.math.BigDecimal;

import javafx.scene.layout.*;
import model.MusicItem;
import model.StoreManager;
import model.WestminsterMusicStoreManager;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;


public class interFace {

    final static StoreManager storeManger = new
WestminsterMusicStoreManager();

    public static void home(Stage primaryStage) throws
IOException {
        primaryStage.setTitle("online Music Store");
        //creat the main pane
        Pane rootforMain = new Pane();
```

```
        TextField Additem = new TextField();
        VBox layout = new VBox();


        Button bt01 = new Button();
        bt01.setText("Add Item");
        bt01.setLayoutX(20);
        bt01.setLayoutY(20);
        rootforMain.getChildren().add(bt01);

        Button bt02 = new Button();
        bt02.setText("Delete Item");
        bt02.setLayoutX(20);
        bt02.setLayoutY(60);
        rootforMain.getChildren().add(bt02);

        Button bt03 = new Button();
        bt03.setText("View Table");
        bt03.setLayoutX(20);
        bt03.setLayoutY(100);
        rootforMain.getChildren().add(bt03);

        Button bt04 = new Button();
        bt04.setText("sort items");
        bt04.setLayoutX(20);
        bt04.setLayoutY(140);
        rootforMain.getChildren().add(bt04);

        Button bt05 = new Button();
        bt05.setText("Buy item");
        bt05.setLayoutX(20);
        bt05.setLayoutY(180);
        rootforMain.getChildren().add(bt05);

        Button bt06 = new Button();
        bt06.setText("Generate a Report");
        bt06.setLayoutX(20);
        bt06.setLayoutY(220);
        rootforMain.getChildren().add(bt06);

        Button bt07 = new Button();
        bt07.setText("Exit");
        bt07.setLayoutX(20);
        bt07.setLayoutY(260);
        rootforMain.getChildren().add(bt07);
```

```
//01 ADD ITEM
        Pane rootforAddItem = new Pane();
        Scene sceneforAddItem = new Scene(rootforAddItem, 450,
500);

        Button bAddItem = new Button();
        bAddItem.setText("Add");
        bAddItem.setLayoutX(120);
        bAddItem.setLayoutY(460);
        rootforAddItem.getChildren().add(bAddItem);
        bt01.setOnAction(e ->
primaryStage.setScene(sceneforAddItem));

//         Controller controller = new Controller();
//         bAddItem.setOnAction(new EventHandler<ActionEvent>() {
//             @Override
//             public void handle(ActionEvent event) {
//                 controller.testButtonAction1();
//             }
//         });


        Button backBu1 = new Button();
        backBu1.setText("Back to Home");
        backBu1.setLayoutY(460);
        backBu1.setLayoutX(200);
        rootforAddItem.getChildren().add(backBu1);

        //itemID
        Label ID = new Label("Item ID:");
        rootforAddItem.getChildren().add(ID);
        ID.setLayoutX(50);
        ID.setLayoutY(50);

        TextField textFieldforID = new TextField();
        textFieldforID.setLayoutX(200);
        textFieldforID.setLayoutY(50);
        rootforAddItem.getChildren().add(textFieldforID);


        //title
        Label Title = new Label("Song Title:");
        rootforAddItem.getChildren().add(Title);
        Title.setLayoutX(50);
```

```
        Title.setLayoutY(100);

        TextField textFieldforTitle = new TextField();
        textFieldforTitle.setLayoutX(200);
        textFieldforTitle.setLayoutY(100);
        rootforAddItem.getChildren().add(textFieldforTitle);


        //genre
        Label Genre = new Label("Song Genre:");
        rootforAddItem.getChildren().add(Genre);
        Genre.setLayoutX(50);
        Genre.setLayoutY(150);

        TextField textFieldforGenre = new TextField();
        textFieldforGenre.setLayoutX(200);
        textFieldforGenre.setLayoutY(150);
        rootforAddItem.getChildren().add(textFieldforGenre);


        //released date
        Label Date = new Label("Enter Released Date:");
        rootforAddItem.getChildren().add(Date);
        Date.setLayoutX(50);
        Date.setLayoutY(200);

        TextField textFieldfordate = new TextField();
        textFieldfordate.setLayoutX(200);
        textFieldfordate.setLayoutY(200);
        rootforAddItem.getChildren().add(textFieldfordate);


        //artist
        Label artist = new Label("Enter Artist Name:");
        rootforAddItem.getChildren().add(artist);
        artist.setLayoutX(50);
        artist.setLayoutY(250);

        TextField textFieldforArtist = new TextField();
        textFieldforArtist.setLayoutX(200);
        textFieldforArtist.setLayoutY(250);
        rootforAddItem.getChildren().add(textFieldforArtist);


        //price
        Label Price = new Label("Enter Price:");
```

```
        rootforAddItem.getChildren().add(Price);
        Price.setLayoutX(50);
        Price.setLayoutY(300);

        TextField textFieldforPrice = new TextField();
        textFieldforPrice.setLayoutX(200);
        textFieldforPrice.setLayoutY(300);
        rootforAddItem.getChildren().add(textFieldforPrice);


        bAddItem.setOnAction(new EventHandler<ActionEvent>() {
            @Override
            public void handle(ActionEvent arg0) {
                //convert int to string
                int numAsString01 =
Integer.parseInt(textFieldforID.getText());
                //convert double to string
                int numAsString02 =
Integer.parseInt(textFieldforPrice.getText());

                MusicItem mus01 = new MusicItem(numAsString01,
textFieldforTitle.getText(), textFieldforGenre.getText(),
textFieldfordate.getText(), textFieldforArtist.getText(),
numAsString02);
                storeManger.addItem(mus01);

            }
        });


//02 DELETE item
        Pane rootforDeleteItem = new Pane();
        Scene sceneforDeleteItem = new Scene(rootforDeleteItem,
450, 500);

        Button bDeleteItem = new Button();
        bDeleteItem.setText("Delete");
        bDeleteItem.setLayoutX(120);
        bDeleteItem.setLayoutY(460);
        rootforDeleteItem.getChildren().add(bDeleteItem);
        bt02.setOnAction(e ->
primaryStage.setScene(sceneforDeleteItem));

        Button backBu2 = new Button();
        backBu2.setText("Back to Home");
        backBu2.setLayoutY(460);
```

```
        backBu2.setLayoutX(200);
        rootforDeleteItem.getChildren().add(backBu2);
//        backBu2.setOnAction(e ->
primaryStage.setScene(scene2));


        Label delete = new Label("Enter itemID for delete:");
        rootforDeleteItem.getChildren().add(delete);
        delete.setLayoutX(50);
        delete.setLayoutY(50);

        TextField textFieldforDelete = new TextField();
        textFieldforDelete.setLayoutX(200);
        textFieldforDelete.setLayoutY(50);
        rootforDeleteItem.getChildren().add(textFieldforDelete);

        bDeleteItem.setOnAction(new EventHandler<ActionEvent>()
{
            @Override
            public void handle(ActionEvent arg0) {
                //convert int to string
                int numAsString03 =
Integer.parseInt(textFieldforDelete.getText());
//                StoreManager storeManager = new
WestminsterMusicStoreManager();
                storeManger.deleteItem(numAsString03);

            }
        });


//03 VIEW table

        Pane rootViewTable = new Pane();
        Scene sceneforViewTable = new Scene(rootViewTable, 450,
500);


        Button bView = new Button();
        bView.setText("View");
        bView.setLayoutX(120);
        bView.setLayoutY(460);
        rootViewTable.getChildren().add(bView);
        bt03.setOnAction(e ->
primaryStage.setScene(sceneforViewTable));
```

```
        Button backBu3 = new Button();
        backBu3.setText("Back to Home");
        backBu3.setLayoutY(460);
        backBu3.setLayoutX(200);
        rootViewTable.getChildren().add(backBu3);
//        backBu3.setOnAction(e ->
primaryStage.setScene(scene2));

        bView.setOnAction(new EventHandler<ActionEvent>() {

            @Override
            public void handle(ActionEvent arg0) {
                storeManger.printList();
            }
        });



//04 sorted list of item

        Pane rootforPrintList = new Pane();
        Scene scenePrintList = new Scene(rootforPrintList, 450,
500);


        Button bSort = new Button();
        bSort.setText("Sort");
        bSort.setLayoutX(120);
        bSort.setLayoutY(460);
        rootforPrintList.getChildren().add(bSort);
        bt04.setOnAction(e ->
primaryStage.setScene(scenePrintList));

        Button backBu4 = new Button();
        backBu4.setText("Back to Home");
        backBu4.setLayoutY(460);
        backBu4.setLayoutX(200);
        rootforPrintList.getChildren().add(backBu4);

        bSort.setOnAction(new EventHandler<ActionEvent>() {

            @Override
            public void handle(ActionEvent arg0) {
```

```
                storeManger.sortedList();
            }
        });


// 05 Buy item
        Pane rootforSort = new Pane();
        Scene sceneforSort = new Scene(rootforSort, 450, 500);


        Button baSort = new Button();
        baSort.setText("Buy");
        baSort.setLayoutX(120);
        baSort.setLayoutY(460);
        rootforSort.getChildren().add(baSort);
        bt05.setOnAction(e ->
primaryStage.setScene(sceneforSort));

        Button backBu5 = new Button();
        backBu5.setText("Back to Home");
        backBu5.setLayoutY(460);
        backBu5.setLayoutX(200);
        rootforSort.getChildren().add(backBu5);


//06 generate a REPORT
        Pane rootforBuy = new Pane();
        Scene sceneforBuy = new Scene(rootforBuy, 450, 500);


        Button bBuy = new Button();
        bBuy.setText("Generate a Report");
        bBuy.setLayoutX(120);
        bBuy.setLayoutY(460);
        rootforBuy.getChildren().add(bBuy);
        bt06.setOnAction(e ->
primaryStage.setScene(sceneforBuy));

        Button backBu6 = new Button();
        backBu6.setText("Back to Home");
        backBu6.setLayoutY(460);
        backBu6.setLayoutX(200);
        rootforBuy.getChildren().add(backBu6);
```

```
//07 exit
        Pane rootforReport = new Pane();
        Scene sceneforReport = new Scene(rootforReport, 450,
500);


        Button bReport = new Button();
        bReport.setText("Exit");
        bReport.setLayoutX(100);
        bReport.setLayoutY(460);
        rootforReport.getChildren().add(bReport);
        bt07.setOnAction(e ->
primaryStage.setScene(sceneforReport));

        Button backBu7 = new Button();
        backBu7.setText("Back to Home");
        backBu7.setLayoutY(460);
        backBu7.setLayoutX(250);
        rootforReport.getChildren().add(backBu7);


//creat main Scene
        Scene scene1 = new Scene(rootforMain, 250, 330);
        primaryStage.setScene(scene1);
        primaryStage.show();
        //call all back to menu button
        backBu1.setOnAction(e -> primaryStage.setScene(scene1));
        backBu2.setOnAction(e -> primaryStage.setScene(scene1));
        backBu3.setOnAction(e -> primaryStage.setScene(scene1));
        backBu4.setOnAction(e -> primaryStage.setScene(scene1));
        backBu5.setOnAction(e -> primaryStage.setScene(scene1));
        backBu6.setOnAction(e -> primaryStage.setScene(scene1));
        backBu7.setOnAction(e -> primaryStage.setScene(scene1));


    }
}


//            public void handle(ActionEvent arg0) {
//                BigDecimal bigDecimalPrice = new
BigDecimal(textFieldforPrice.getText());
//
//            }
//        });
```

StoreManager.java

Code

```
package model;

public interface StoreManager{
    //creat a methods
    public void addItem(MusicItem item);
    public void deleteItem (int itemId);
    public void printList();
    public void sortedList();
}
```

MusicItem.java

Code

```
package model;

import java.util.Objects;
//implements Comparable<MusicItem>
public class MusicItem implements Comparable<MusicItem> {
    protected int itemID;
    protected String title;
    protected String genre;
    protected String releaseDate;
    protected String artist;
    protected int price;

    public MusicItem(int itemID, String title, String genre,
String releaseDate, String artist, int price) {
        super();
        this.itemID = itemID;
        this.title = title;
        this.genre = genre;
        this.releaseDate = releaseDate;
        this.artist = artist;
        this.price = price;
    }

    public int getItemID() {
```

```
        return itemID;
    }

    public void setItemID(int itemID) {
        this.itemID = itemID;
    }

    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public String getGenre() {
        return genre;
    }

    public void setGenre(String genre) {
        this.genre = genre;
    }

    public String getReleaseDate() {
        return releaseDate;
    }

    public void setReleaseDate(String releaseDate) {
        this.releaseDate = releaseDate;
    }

    public String getArtist() {
        return artist;
    }

    public void setArtist(String artist) {
        this.artist = artist;
    }

    public int getPrice() {
        return price;
    }

    public void setPrice(int price) {
        this.price = price;
    }
```

```
//set equals
    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return
false;
        MusicItem musicItem = (MusicItem) o;
        return itemID == musicItem.itemID &&
                price == musicItem.price &&
                title.equals(musicItem.title) &&
                genre.equals(musicItem.genre) &&
                releaseDate.equals(musicItem.releaseDate) &&
                artist.equals(musicItem.artist);
    }
//generate hashCode
    @Override
    public int hashCode() {
        return Objects.hash(itemID, title, genre, releaseDate,
artist, price);
    }


    @Override
    public int compareTo(MusicItem obj) {
        return this.title.compareTo(obj.getTitle());
    }

    @Override
    public String toString() {
        return "MusicItem{" +
                "itemID='" + itemID + '\'' +
                ", title='" + title + '\'' +
                ", genre='" + genre + '\'' +
                ", releaseDate='" + releaseDate + '\'' +
                ", artist='" + artist + '\'' +
                ", price=" + price +
                '}';
    }
}
```

WestminsterMusicStoreManager.java

Code

```
package model;

import java.sql.*;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.List;

public class WestminsterMusicStoreManager implements
StoreManager {

    final int MAXCOUNT=1000;

    @Override
    public void addItem(MusicItem item) {

        try {
            // create a mysql database connection
            String myDriver = "org.gjt.mm.mysql.Driver";
            String myUrl = "jdbc:mysql://localhost:8889/
WestminsterMusicStore";
            Class.forName(myDriver);
            Connection conn = DriverManager.getConnection(myUrl,
"root", "root");

            String query1 = "SELECT COUNT(*) FROM musicitem";
            Statement stat = conn.createStatement();
            ResultSet result = stat.executeQuery(query1);
            result.next();

            int count =result.getInt(1);
            if (count<MAXCOUNT) {

                // the mysql insert statement
                String query = " insert into musicitem (itemID,
title, genre, releaseDate, artist, price)"
                        + " values (?, ?, ?, ?, ?, ?)";

                // create the mysql insert preparedstatement
                PreparedStatement preparedStmt =
conn.prepareStatement(query);
```

```
                preparedStmt.setInt(1, item.getItemID());
                preparedStmt.setString(2, item.getTitle());
                preparedStmt.setString(3, item.getGenre());
                preparedStmt.setString(4,
item.getReleaseDate());
                preparedStmt.setString(5, item.getArtist());
                preparedStmt.setDouble(6, item.getPrice());

                System.out.println("DATA ENTERED");
//              execute the preparedstatement
                preparedStmt.execute();



            }else {
                System.out.println("you entered maximum items
for database ");
            }
            conn.close();
        } catch (Exception e) {
            System.err.println("Got an exception!");
            System.err.println(e.getMessage());
        }
    }

    @Override
    public void deleteItem(int itemId) {
        try {
            // create a mysql database connection
            String myDriver = "org.gjt.mm.mysql.Driver";
            String myUrl = "jdbc:mysql://localhost:8889/
WestminsterMusicStore";
            Class.forName(myDriver);
            Connection conn = DriverManager.getConnection(myUrl,
"root", "root");

            // create the mysql delete statement.
            String query = "delete from musicitem where itemID =
?";
            PreparedStatement preparedStmt =
conn.prepareStatement(query);
            preparedStmt.setInt(1, itemId);

            System.out.println("Item Deleted");
```

```
            // execute the preparedstatement
            preparedStmt.execute();

            conn.close();
        } catch (Exception e) {
            System.err.println("Got an exception! ");
            System.err.println(e.getMessage());
        }
    }

    @Override
    public void printList() {
        try {
            // create a mysql database connection
            String myDriver = "org.gjt.mm.mysql.Driver";
            String myUrl = "jdbc:mysql://localhost:8889/
WestminsterMusicStore";
            Class.forName(myDriver);
            Connection conn = DriverManager.getConnection(myUrl,
"root", "root");

            String query = "SELECT * FROM musicitem";

            // create the java statement
            Statement stat = conn.createStatement();

            // execute the query, and get a java resultset
            ResultSet result = stat.executeQuery(query);

            while (result.next()) {
                System.out.println("id : " +
result.getString("itemID"));
                System.out.println("title : " +
result.getString("title"));
                System.out.println("genre : " +
result.getString("genre"));
                System.out.println("date : " +
result.getString("releaseDate"));
                System.out.println("artist : " +
result.getString("artist"));
                System.out.println("price : " +
result.getString("price"));

System.out.println("////////////////////////////////////");
            }
        } catch (Exception e) {
```

```
            System.err.println("Got an exception");
            System.err.println(e.getMessage());
        }
    }
        @Override
        public void sortedList () {
            SortCategory.sortItems();
        }
```

SortCatagory.java

Code

```
package model;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class SortCategory {

    public static void sortItems(){
        try {
            // create a mysql database connection
            String myDriver = "org.gjt.mm.mysql.Driver";
            String myUrl = "jdbc:mysql://localhost:8889/
WestminsterMusicStore";
            Class.forName(myDriver);
            Connection conn = DriverManager.getConnection(myUrl,
"root", "root");

            String query = "SELECT * FROM musicitem ORDER BY
title ASC";

            // create the java statement
            Statement stat = conn.createStatement();

            // execute the query, and get a java resultset
            ResultSet result = stat.executeQuery(query);
```

```
            while (result.next()) {
                System.out.println("id : " +
result.getString("itemID"));
                System.out.println("title : " +
result.getString("title"));
                System.out.println("genre : " +
result.getString("genre"));
                System.out.println("date : " +
result.getString("releaseDate"));
                System.out.println("artist : " +
result.getString("artist"));
                System.out.println("price : " +
result.getString("price"));

System.out.println("//////////////////////////////////");
            }
        } catch (Exception e) {
            System.err.println("Got an exception");
            System.err.println(e.getMessage());
        }
    }
}
```

Vinly.java

Code

```
package model;
import java.util.Objects;

class Vinyl extends MusicItem {
    private double speed;
    private double diameter;

    public Vinyl(int itemID, String title, String genre, String
releaseDate, String artist, int price, double speed, double
diameter) {
        super(itemID, title, genre, releaseDate, artist, price);
        this.speed = speed;
        this.diameter = diameter;
    }

    @Override
```

```
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return
false;
        if (!super.equals(o)) return false;
        Vinyl vinyl = (Vinyl) o;
        return Double.compare(vinyl.speed, speed) == 0 &&
                Double.compare(vinyl.diameter, diameter) == 0;
    }

    @Override
    public int hashCode() {
        return Objects.hash(super.hashCode(), speed, diameter);
    }

    public double getSpeed() {
        return speed;
    }

    public double getDiameter() {
        return diameter;
    }

    @Override
    public String toString() {
        return "Vinyl{" +
                "speed=" + speed +
                ", diameter=" + diameter +
                ", itemID='" + itemID + '\'' +
                ", title='" + title + '\'' +
                ", genre='" + genre + '\'' +
                ", releaseDate='" + releaseDate + '\'' +
                ", artist='" + artist + '\'' +
                ", price=" + price +
                '}';
    }
}
```

CD.java

Code

```java
package model;

public class CD extends MusicItem {
    private int duration;

    public CD(int itemID, String title, String genre, String
releaseDate, String artist, int price,
              int duration) {
        super(itemID, title, genre, releaseDate, artist, price);
        this.duration = duration;
    }


    public int getDuration() {
        return duration;
    }

    @Override
    public int hashCode() {
        final int prime = 31;
        int result = super.hashCode();
        result = prime * result + duration;
        return result;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (!super.equals(obj))
            return false;
        if (getClass() != obj.getClass())
            return false;
        CD other = (CD) obj;
        if (duration != other.duration)
            return false;
        return true;
    }
}
```

• Black Box Testing

Run the program

No	Input	Expected output	Actual Output	Bug
1	Select Add Item	Display Add Item page	Display Add Item page	No
2	Select Delete Item	Display Delete Item page	Display Delete Item page	No
3	Select View Table	Display View Table page	Display View Table page	No
4	Select Sort Item	Display Sort Item page	Display Sort Item page	No
5	Select Buy Item	Display Buy Item page	Display Buy Item page	No
	Select Generate a Report	Display Generate a Report page	Display Generate a Report page	No

Add Item

No	Input	Expected output	Actual output	bug
1	Input Item ID	Display Data Entered	Display Data Entered	No
2	Input Title	Display Data Entered	Display Data Entered	No
3	Input Genre	Display Data Entered	Display Data Entered	No
4	Input Release date	Display Data Entered	Display Data Entered	No
5	Input Artist Name	Display Data Entered	Display Data Entered	No

6	Input Price	Display Data Entered	Display Data Entered	No
7	Click Add button	Display Data Entered	Display Data Entered	No
8	Click back to home button	Display Main Menu	Display Main Menu	No

Delete Item

No	Input	Expected output	Actual output	bug
1	Input item ID for delete	Display Enter item id for delete	Display Enter item id for delete	No
2	Click back to home button	Display Main Menu	Display Main Menu	No
3	Click Delete button	Display Item Deleted	Display Item Deleted	No

View Table

No	Input	Expected output	Actual Output	Bug
1	View Table clicked	Display View Table page	Display View Table page	No
2	Click View button	Display musicItem table	Display musicItem table	No
3	Click back to home button	Display Main Menu	Display Main Menu	No

Sort Item

No	Input	Expected output	Actual Output	Bug
1	Sort item clicked	Display Sort page	Display Sort page	No
2	Click Sort button	Display items are sorted	Display items are sorted	No
3	Click back to home button	Display Main Menu	Display Main Menu	No

Buy item

No	Input	Expected output	Actual Output	Bug
1	Sort item clicked	Display Buy item page	Display Buy item page	No
2	Click buy button	Display brought it	Display brought it	No
3	Click back to home button	Display Main Menu	Display Main Menu	No

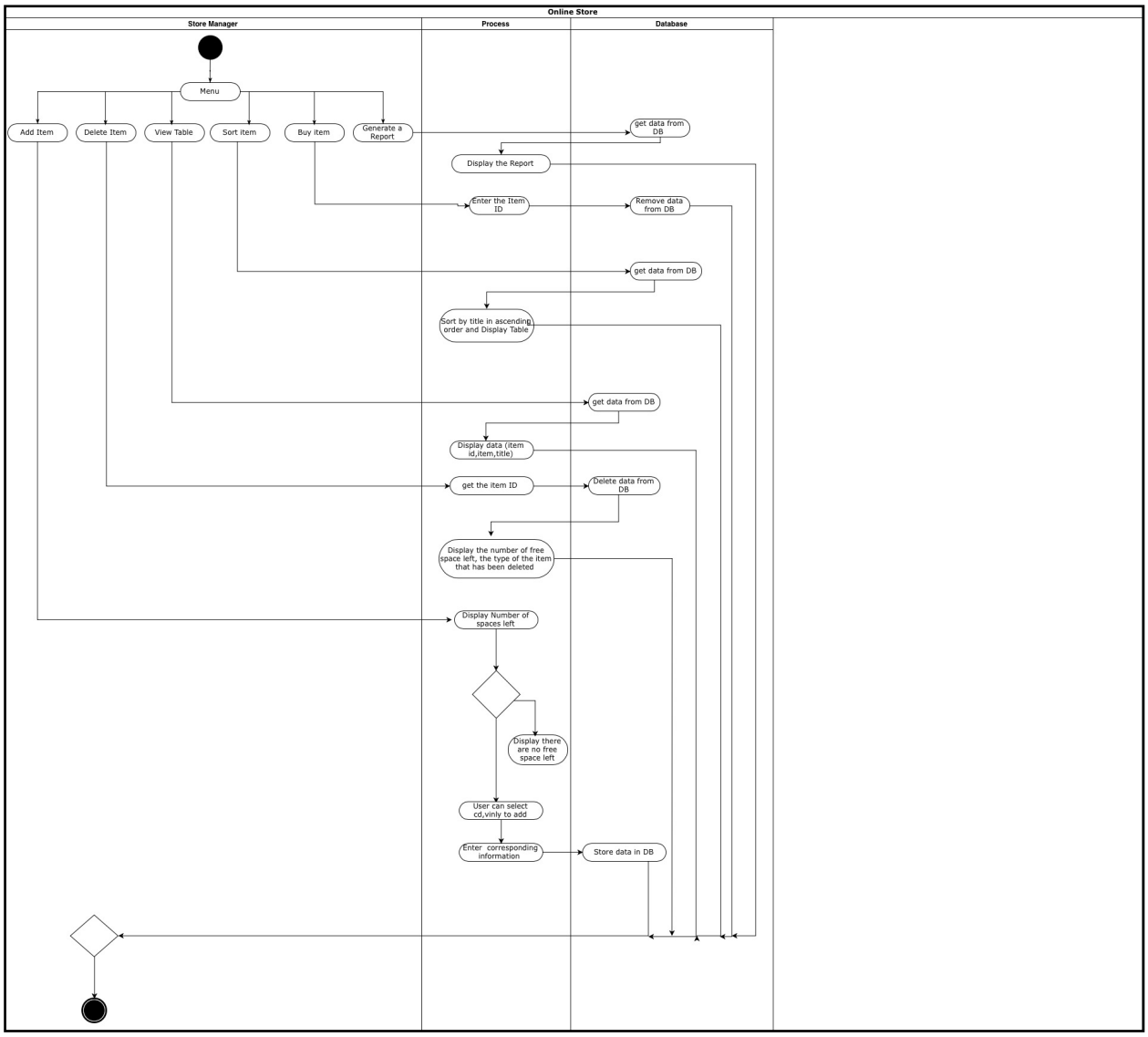
Generate a Report

No	Input	Expected output	Actual Output	Bug
1	Generate a Report clicked	Display Generate a Report page	Display Generate a Report page	No
2	Click Generate a Report button	Display Report	Display Report	No
3	Click back to home button	Display Main Menu	Display Main Menu	No

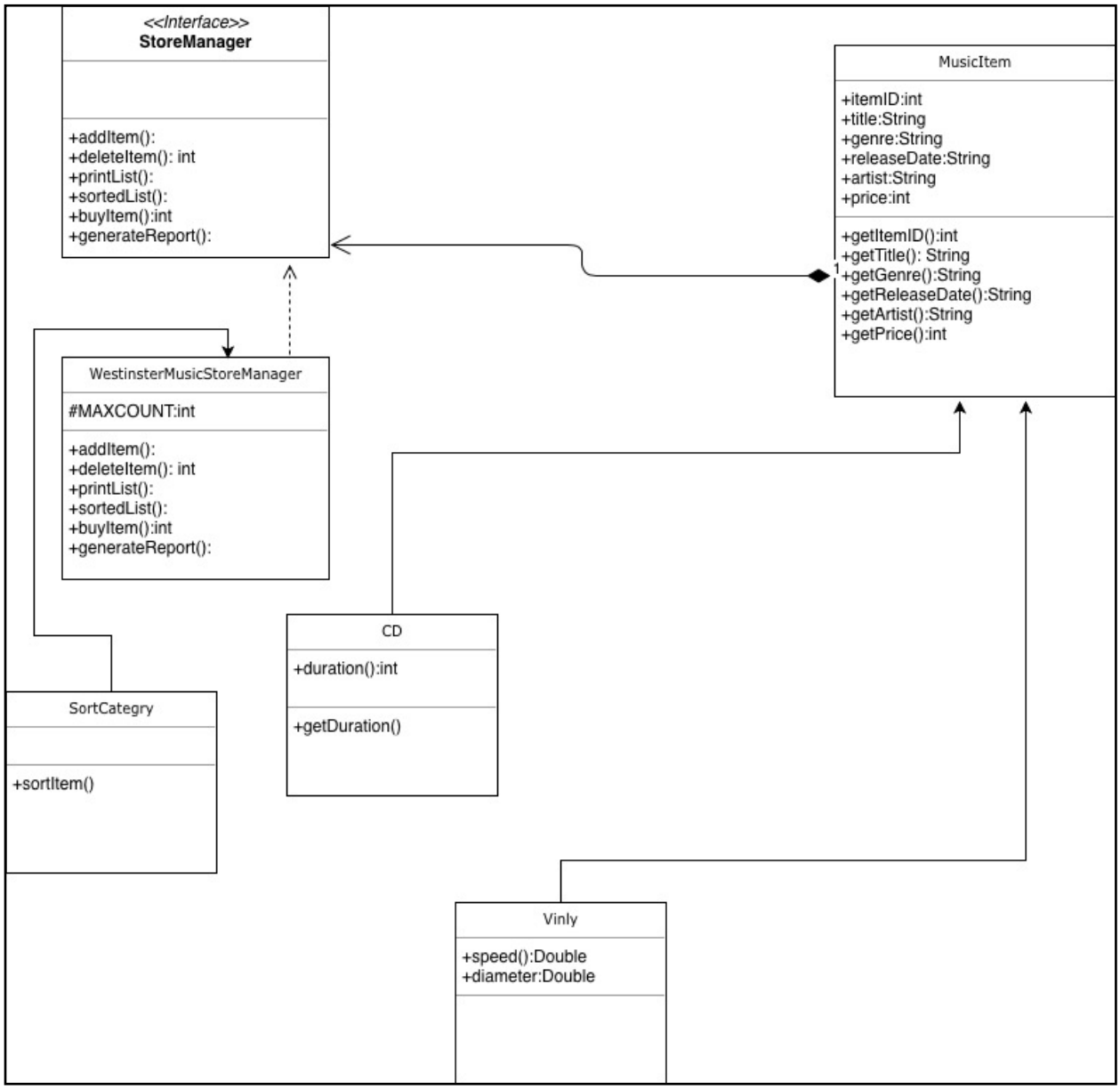
- White Box Testing

Serial No	Function	Condition	Path
01	Add Item	If (item less than 1000)	Path 1(True) Data added to Database Path 2(False) Display: you entered maximum items for database
02	Add item	If (item ID duplicated)	Path 1(True) Not added to database Path 2(False) Added to database

Activity Diagram



Class Diagram



Use case Diagram