SCHOOL OF COMPUTING (SOC)

SECOND SEMESTER SESSION 2022/2023 (A222)

STIA 1123 PROGRAMMING 2

GROUP E

GROUP PROJECT (20%)

TITLE:

MOOSIC CD LIBRARY MANAGEMENT SYSTEM

PREPARED FOR:

DR. ALAWIYAH BT ABD WAHAB

PREPARED BY:

GROUP "DRUMMERS"

TEAM MEMBERS	MATRIC NUMBER
LEE JUAN	280027
EDWIN LIM WEI BIN	281775

TABLE OF CONTENTS

Content **Page**

1.0 INTRODUCTION

1.1 Background of the Project

Libraries continue to change and develop as society enters the digital era in order to suit the demands of their users. The systematic arrangement and effective administration of CDs (Compact Discs) and other digital media resources is a crucial component of library management. An innovative method for streamlining the management of CD collections, ensuring easy access to audio files, and improving user experience is to program a CD Library Management System. This solution transforms how libraries manage and offer access to priceless audio resources by embracing technology and the digital world.

The Moosic CD Library Management System project aims to enhance the domain of a CD library by providing an efficient and user-friendly system for managing CD information. With the rise of digital media, CDs are still prevalent in some areas especially in CD collector communities, and a system to organize and track CD collections becomes essential. The project is designed to address the needs of users who wants to manage their physical CD collection in an organized manner, keeping track of essential details such as title, artist, genre, release year, price, and quantity.

1.2 Problem Statement

The traditional method of manually managing CD collections can be time-consuming and prone to errors. Without a dedicated system, users might struggle to keep track of their CD library, leading to duplicate purchases, difficulty in locating CDs, or even forgetting about certain CDs in their possession. Moreover, organizing CDs based on various attributes like genre, artist, or release year might become cumbersome without a systematic approach. The lack of a centralized repository to store CD information can lead to a disjointed and inefficient management process.

1.3 Objective

The main objective of the Moosic CD Library Management System project is to provide an automated and organized solution for CD collection management. The system aims to offer the following benefits:

i. Efficient CD Cataloguing: Enable users to add new CDs to the library, providing details like category, title, artist, genre, release year, price, and quantity.
ii. Easy Search and Edit: Allow users to search for CDs based on ID or other attributes and edit their information seamlessly.
iii. Quick Deletion: Facilitate the removal of CDs from the library, ensuring the information is appropriately updated.
iv. Statistics and Insights: Provide statistical information such as the total number of CDs, total copies, average unit price, and the total value of the CD collection.
v. Data Persistence: Save the CD information to a local file, ensuring that the library data remains available even after system restarts.

1.4 Scope of the Project

The scope of the Moosic CD Library Management System project can be categorized into user scope and system scope:

1.4.1 User Scope

- Users can interact with the Graphical User Interface (GUI) to display, add, search, edit, or delete CD information.
- Users can view statistical insights into their CD library, such as the total number of CDs, total copies, average unit price, and the total value of the collection.
- Users can reset all input fields and CD information list if desired.

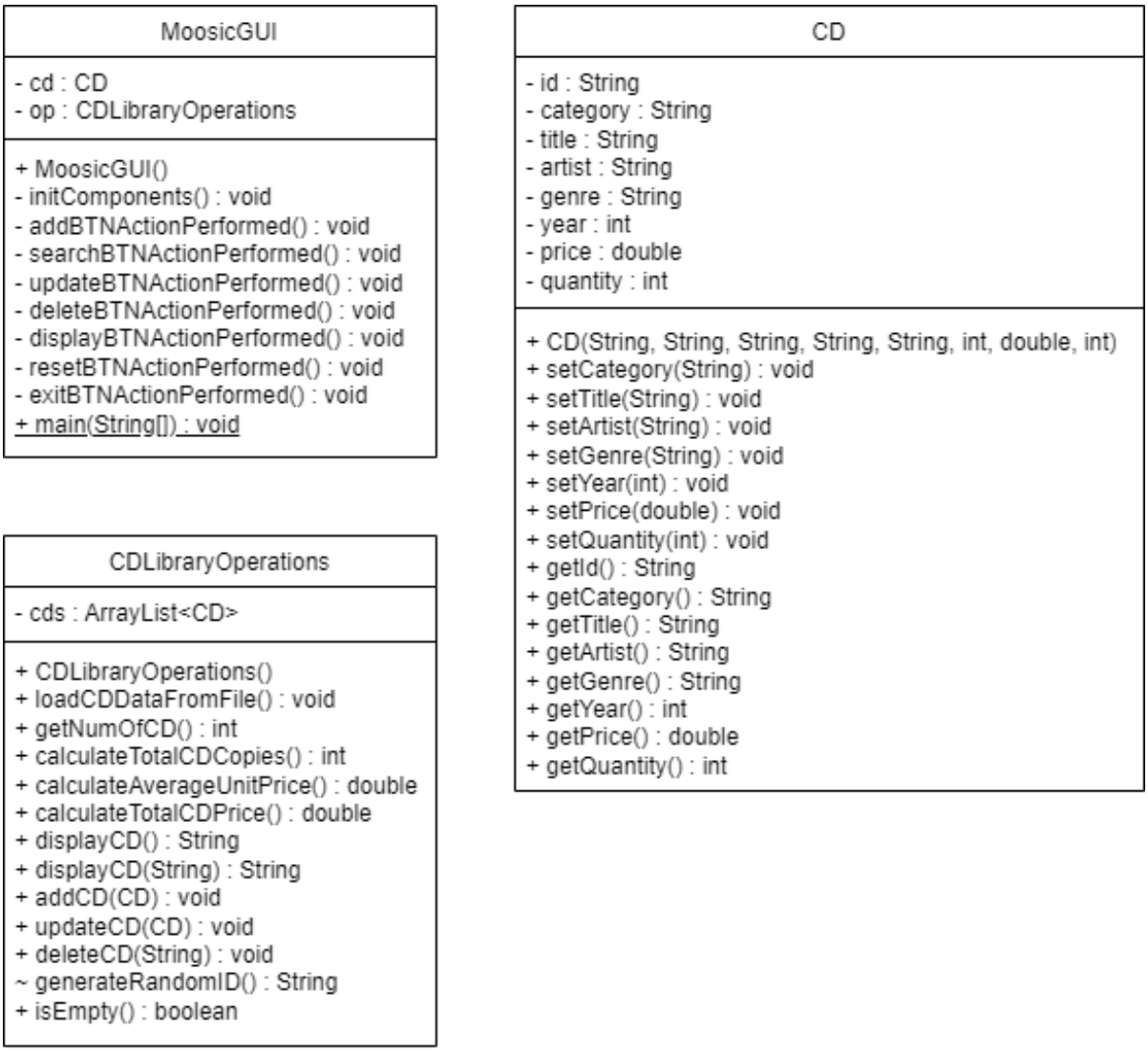
1.4.2 System Scope

The system scope for the Moosic CD Library Management System:

- The system provides a GUI-based application for users to manage their CD collection efficiently. Several containers and swing controls inside the system are used such as text field, panel, combo box, button etc.
- The system allows users to view and add CDs, and search for CDs based on ID and view their details, and users could update or delete the information of the searched CD.
- The system calculates and displays statistical insights based on the CD library data.
- The system utilises the ArrayList data structure to efficiently manage and organise CD data within the program.
- The system handles potential errors i.e. input errors, file errors to prevent unexpected crashes while providing informative feedback to the users.
- The system saves CD information to a local file to ensure data persistence.

2.0 UML CLASS DIAGRAM

The figure attached below is the UML class diagrams for our Moosic CD Library Management System Project, each representing the three classes: *MoosicGUI.java*, *CDLibraryOperations.java*, and *CD.java*.

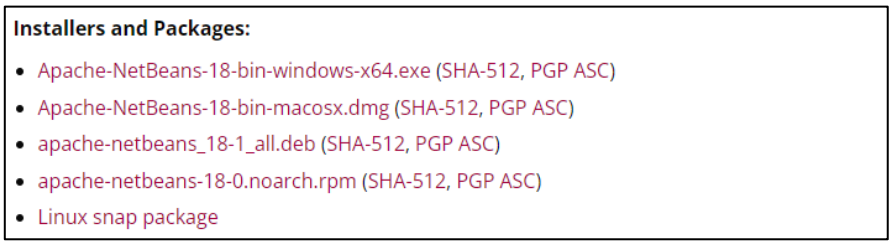


3.0 USER MANUAL

Below is a user manual for users to access, open and run the program:

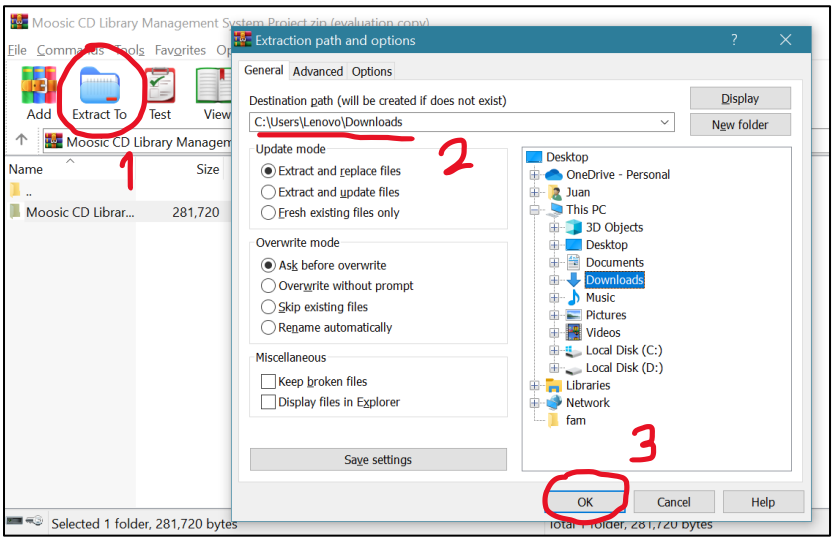
3.1 Setting Up the Program

1. To have NetBeans IDE on your computer, click here and find the section as shown in the screenshot below. Download the package for your operating system, e.g. “Apache-NetBeans-18-bin-windows-x64.exe”. After it is done, open the executable to start installing the application.



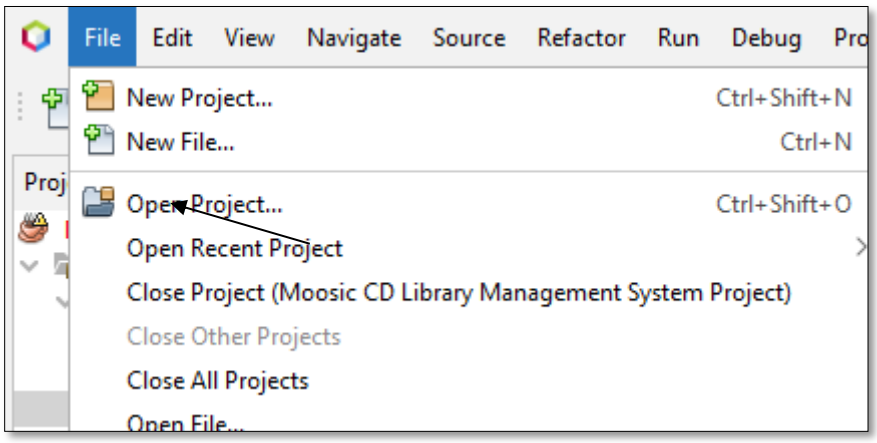
2. Next, after downloading the zipped folder of our program, extract it using a folder extract tool e.g. *WinRAR*. You may download WinRAR here.

3. In *WinRAR*, click “Extract To”, then select the folder or destination path that you want to save the program in. Click “OK”.

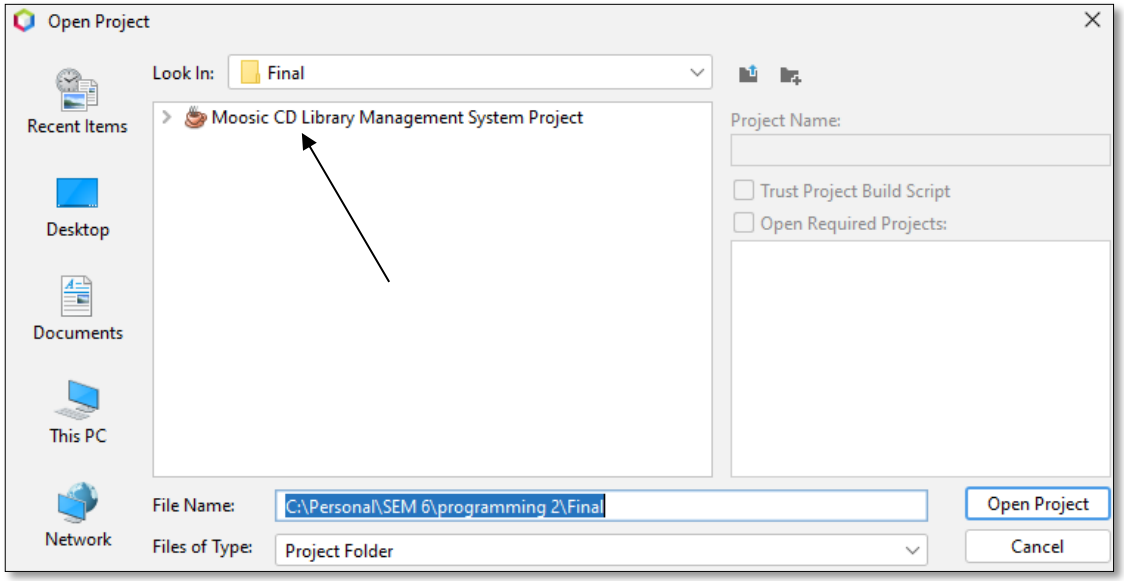


4. Once NetBeans IDE has done installed and the zipped folder extraction has completed, you can then open NetBeans IDE.

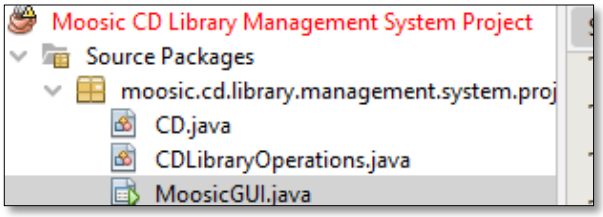
5. Click the “File” button at the top menu of the application, then click “Open Project…”.



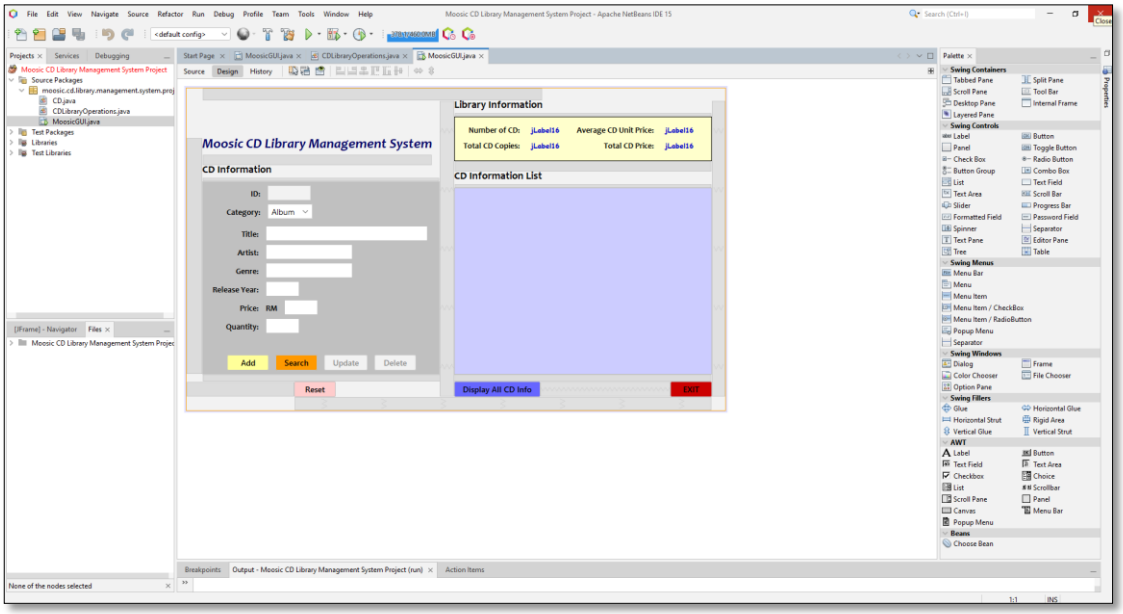
6. After that, the system will display a window to show the file in your device. You need to find the directory of the file you have just extracted to, Click on the file and open.



7. After opening the project, it will show three java files at the project panel.

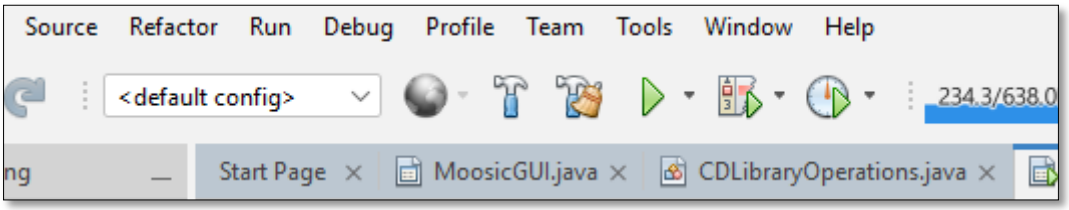


8. Click on the “*MoosicGUI.java*” tab, and you will be led to the interface as shown below:



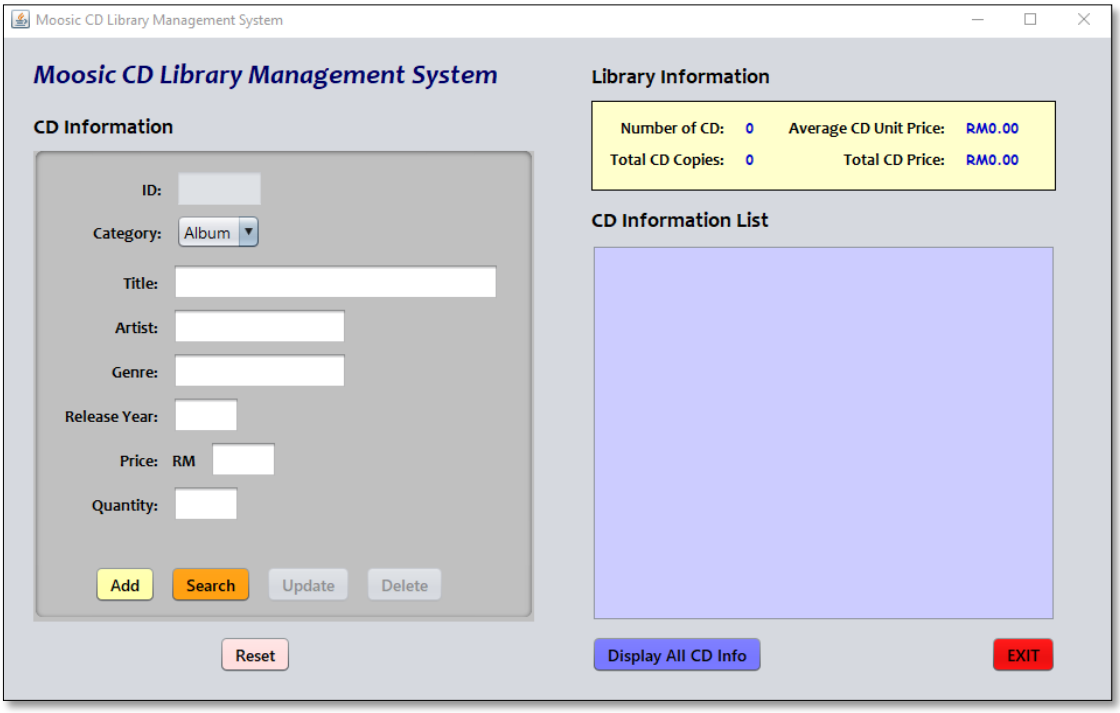
9. After opening *MoosicGUI.java*, you have two ways to run the program:

- Click the “Run Project” button at the tool bar above the NetBeans application:

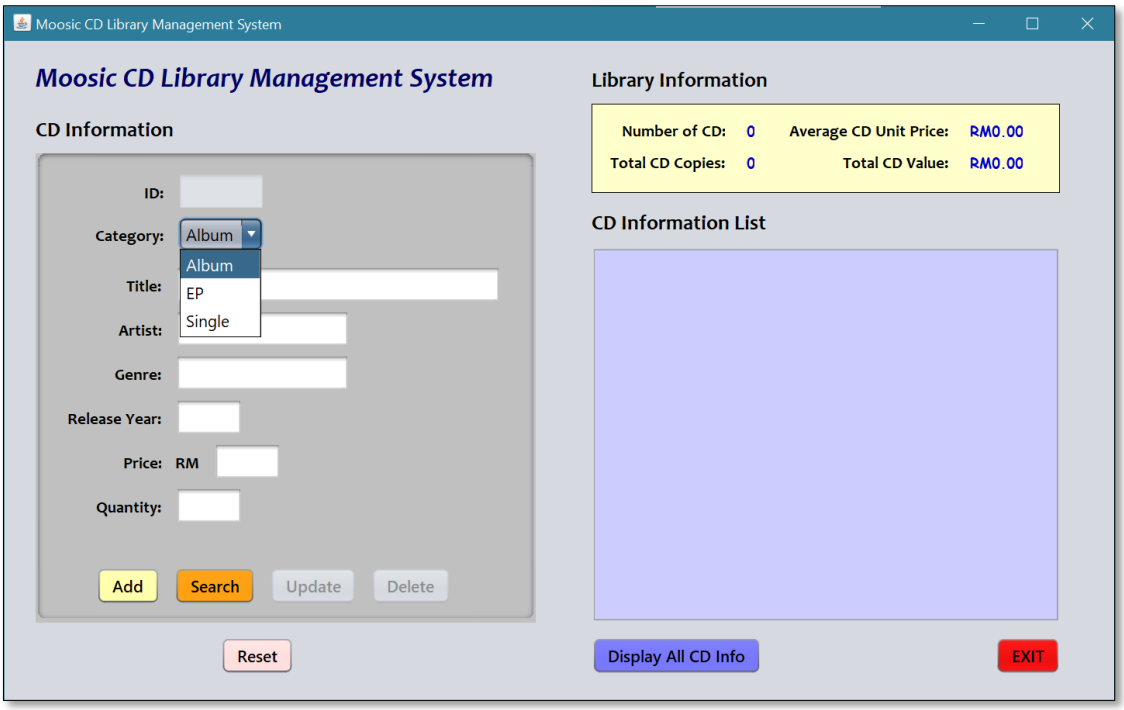


- Or, press “Shift” and “F6” together, the program will start running.

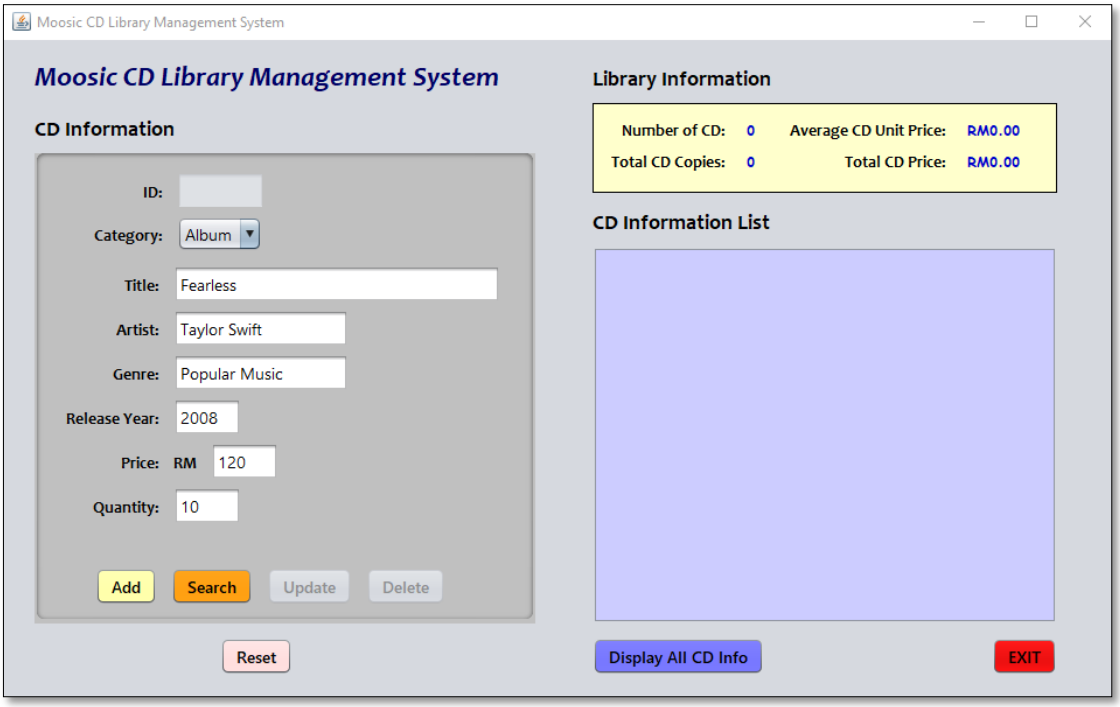
3.2 Using the Program



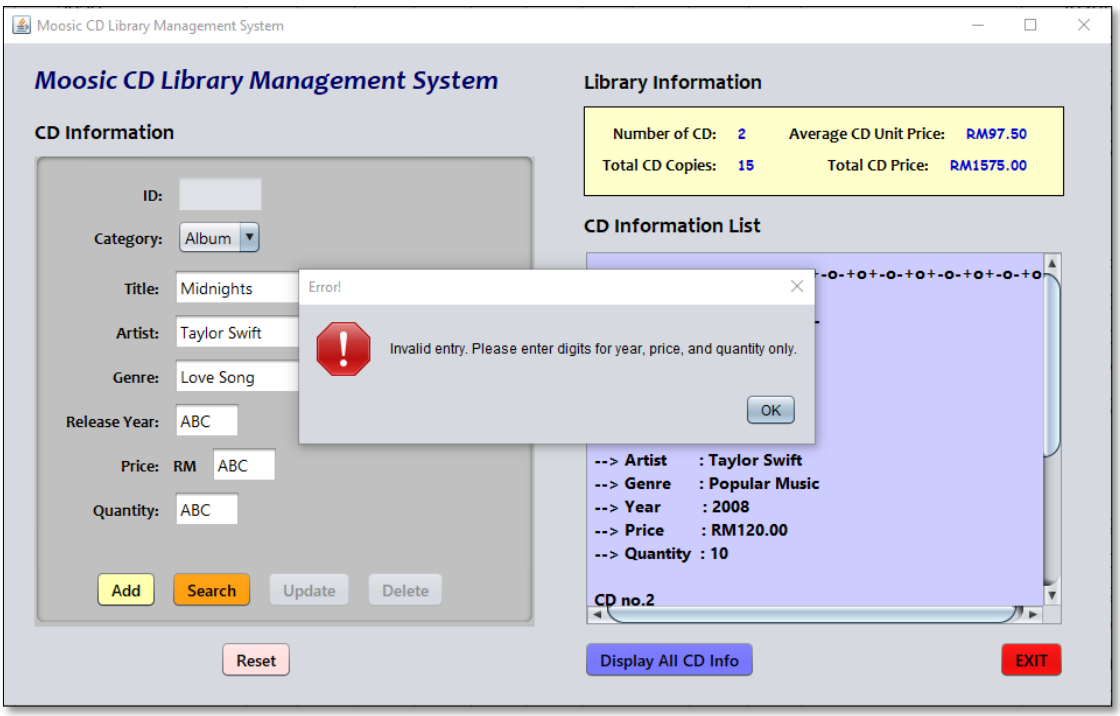
Above is the user interface of the Moosic CD Library Management System Program when it starts running.



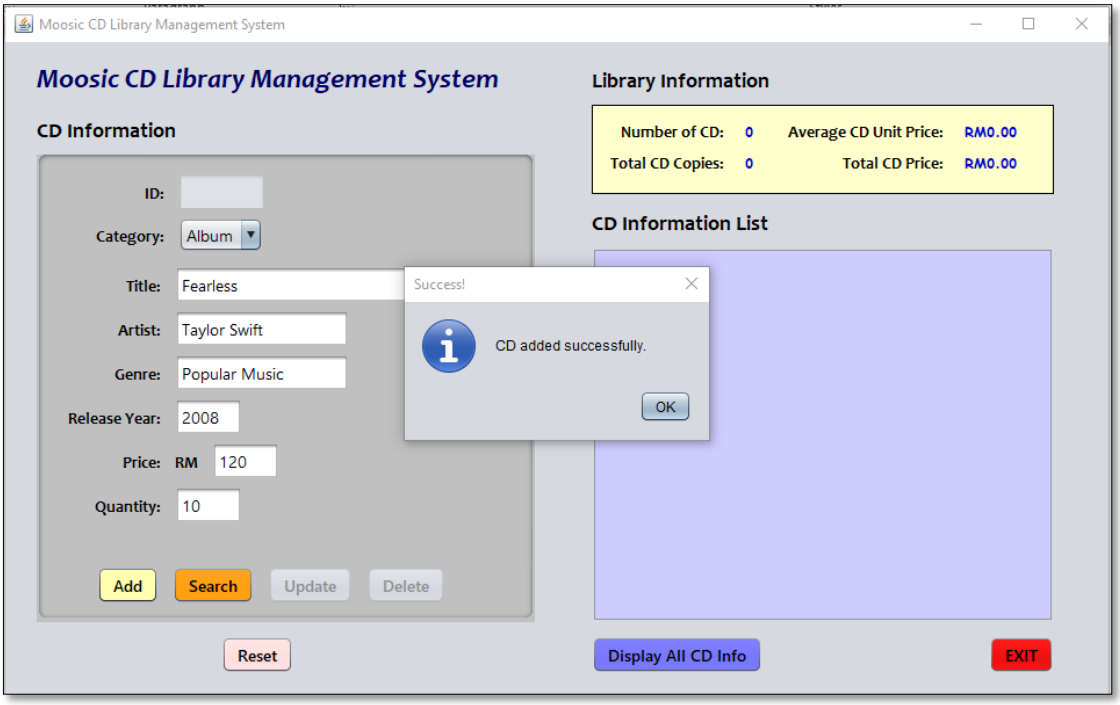
User can choose the category of CD they want to save through the combo box shown above.



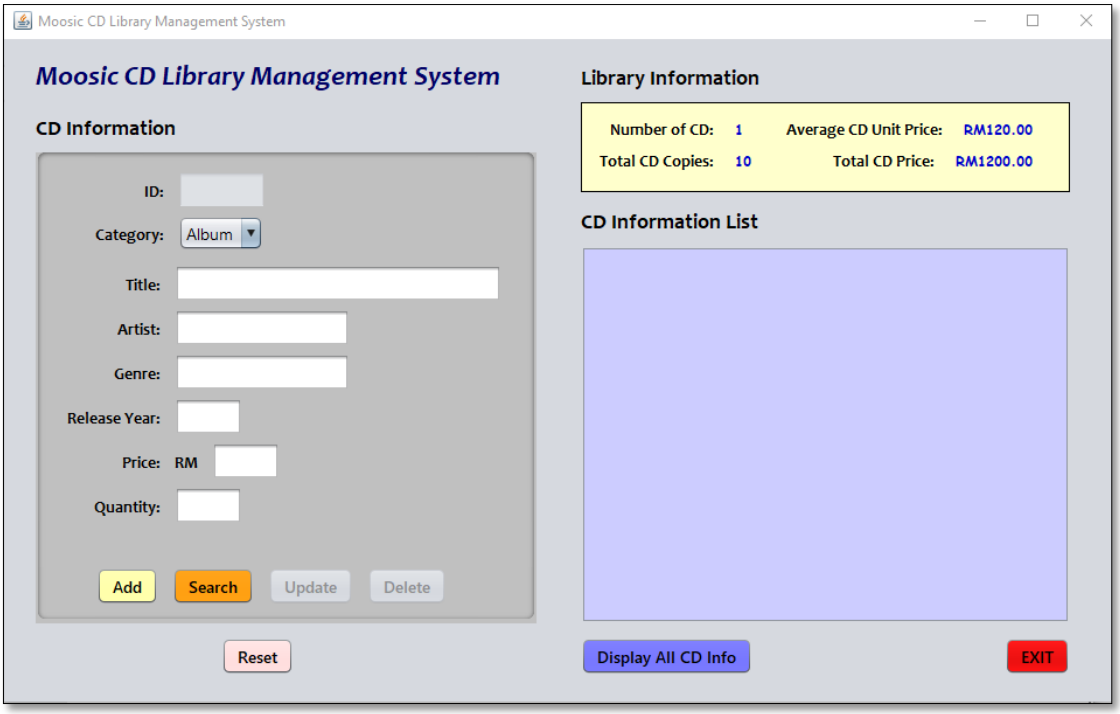
After user input the CD information, they need to click the “Add” button so the information can be saved in the system.



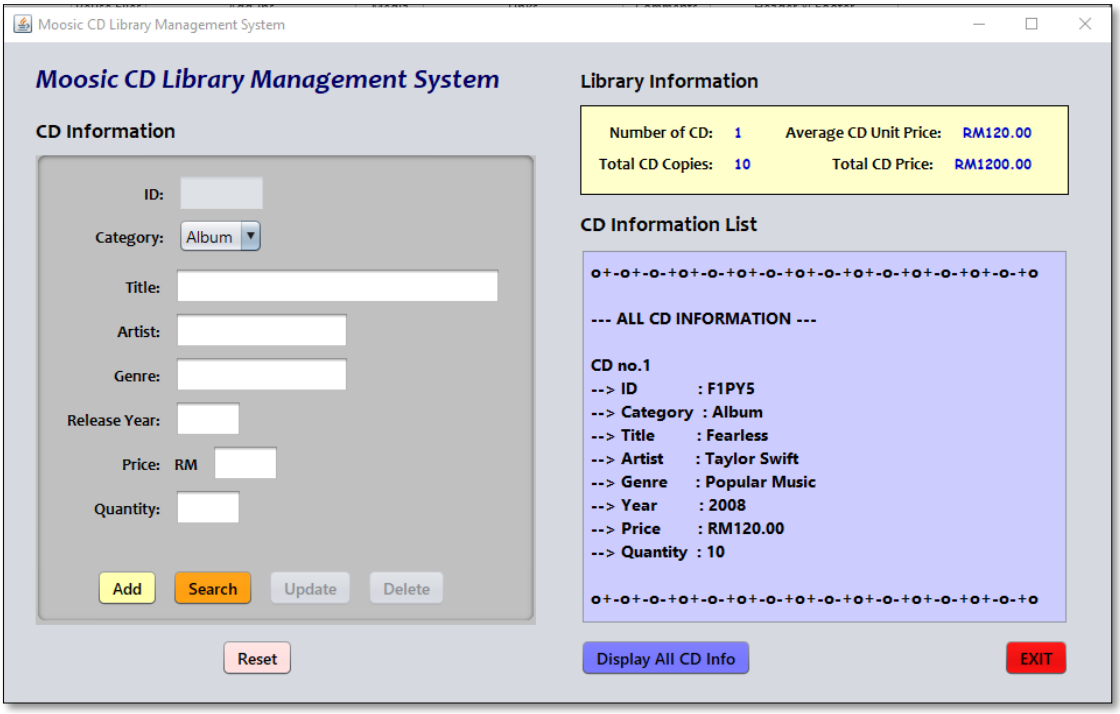
When user enters invalid input, the system will display the respective message dialog as shown above and user may try again.



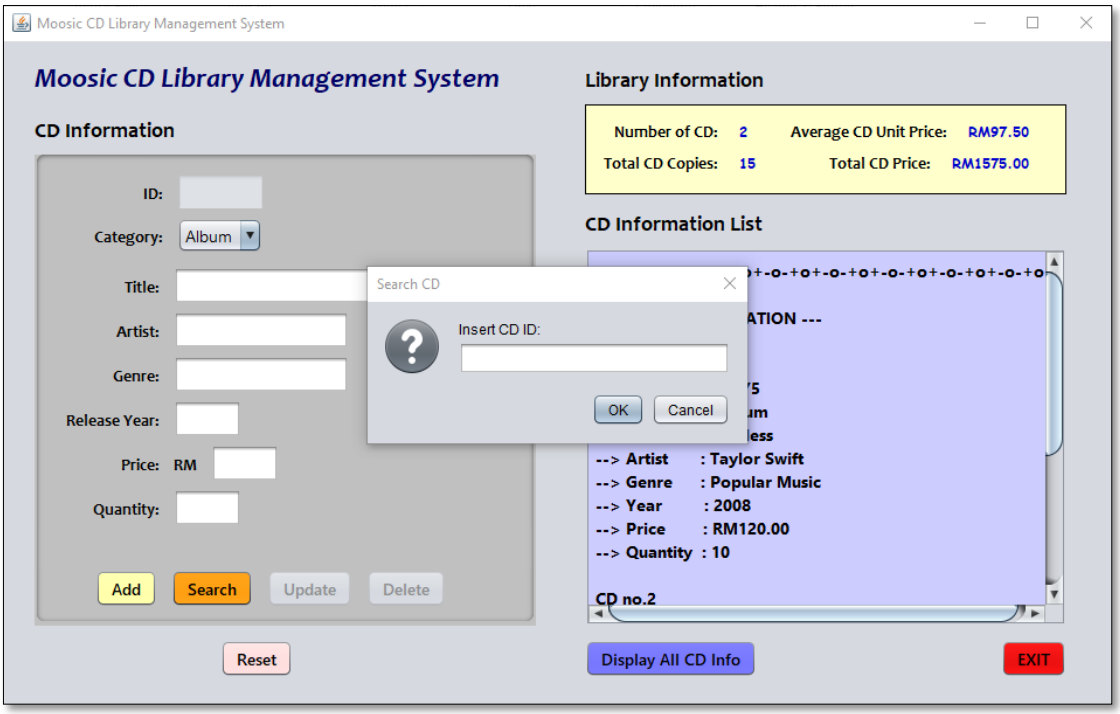
The program will display the above information dialog when the inserted CD information has been added successfully.



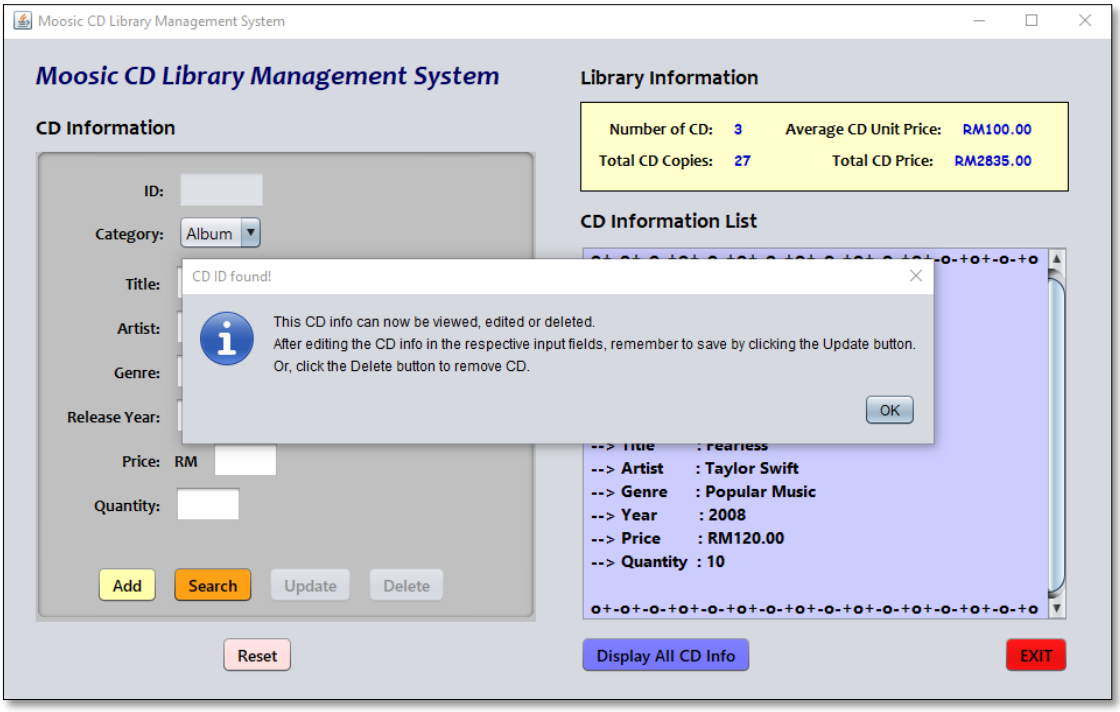
After a CD is added, the “Library Information” panel will be updated with the saved information.



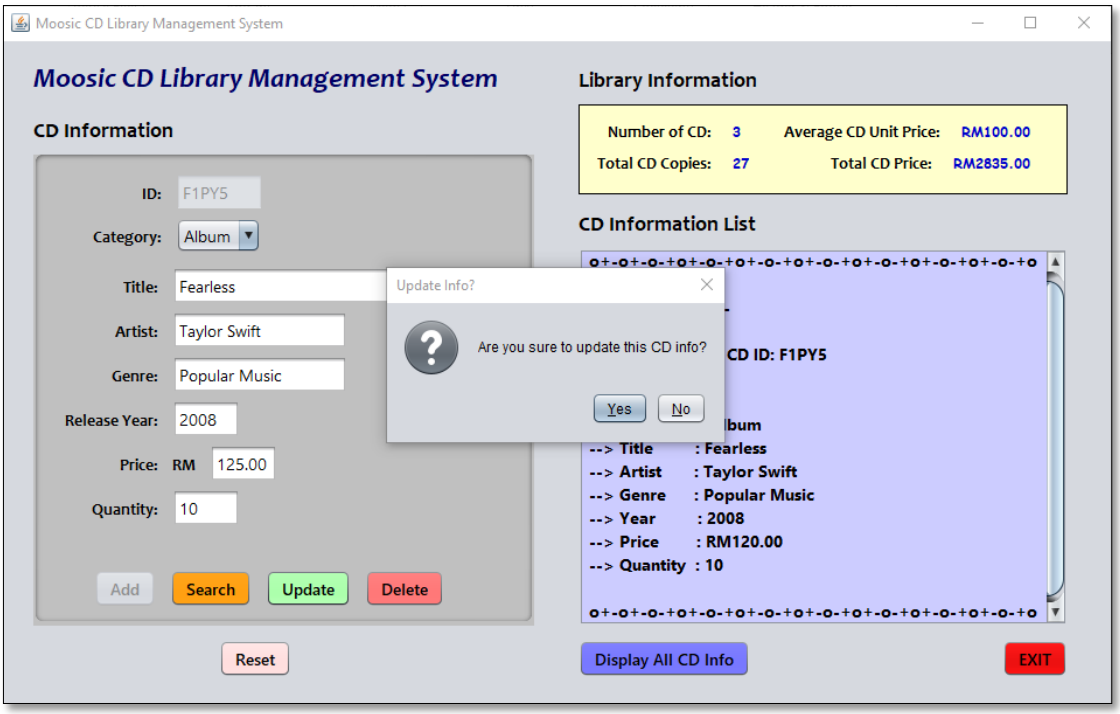
By clicking the “Display All CD Info” button, the “CD Information List” text area will display all saved CD information. The ID of the CD will be randomly generated by the program.



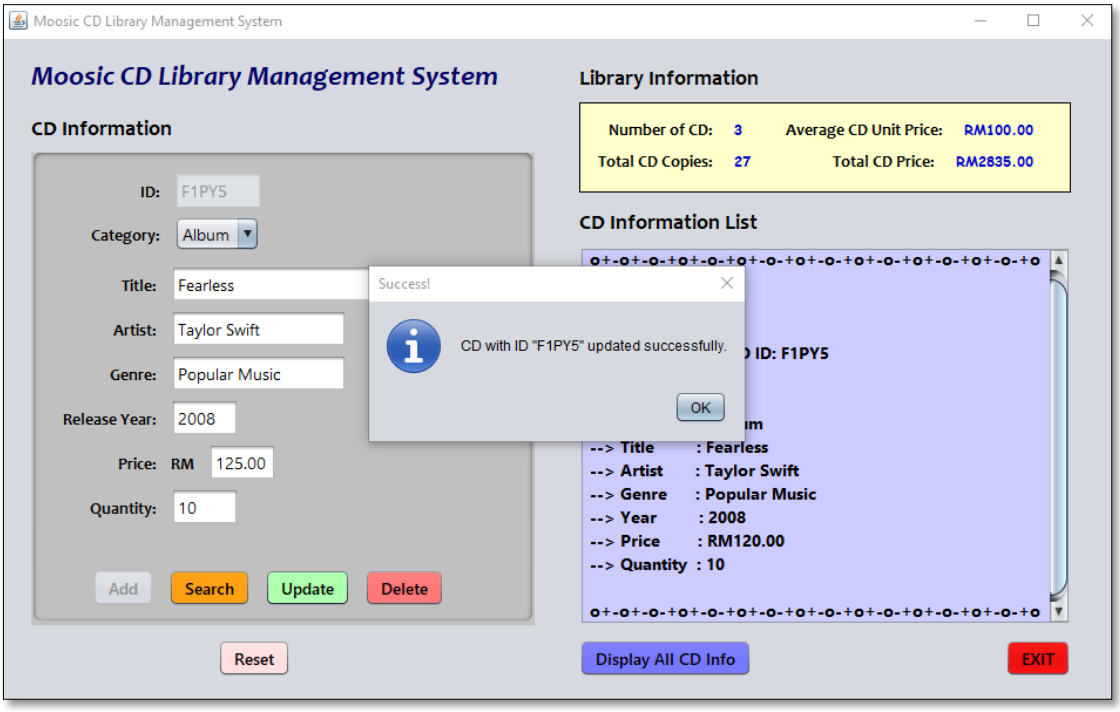
When user clicks on the “Search” button, program will require user to input the ID of the CD in order to display CD information of the inserted ID.



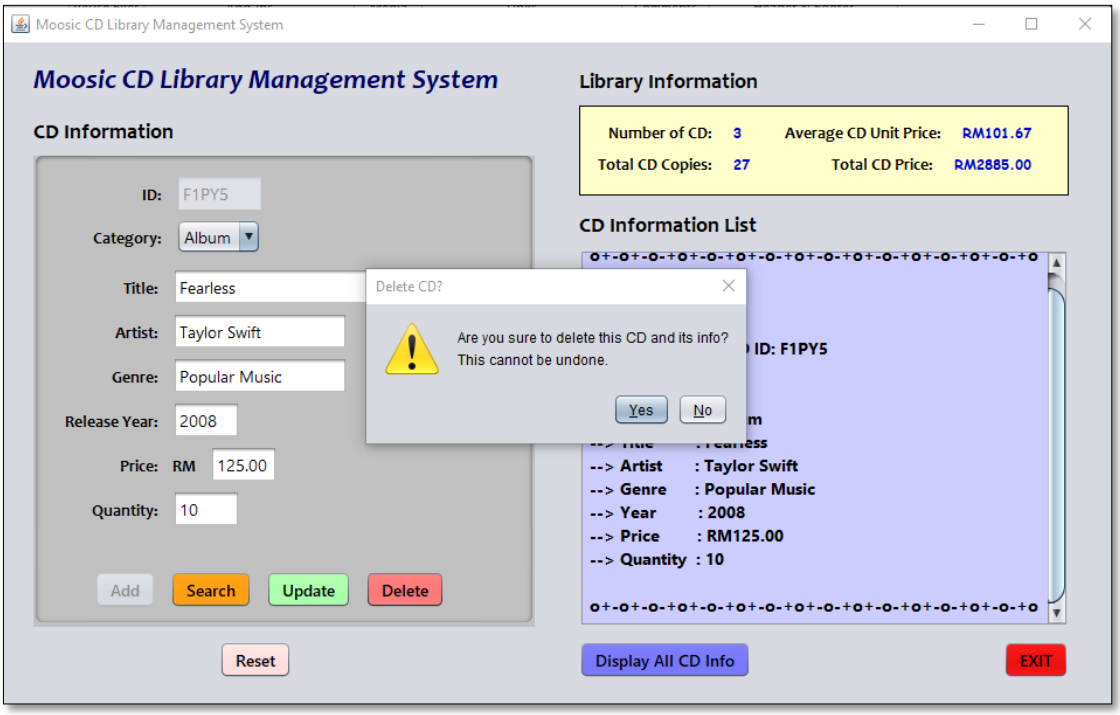
When the correct CD ID is entered, the system will display a message to inform users that they can now view, edit and delete the particular CD information.



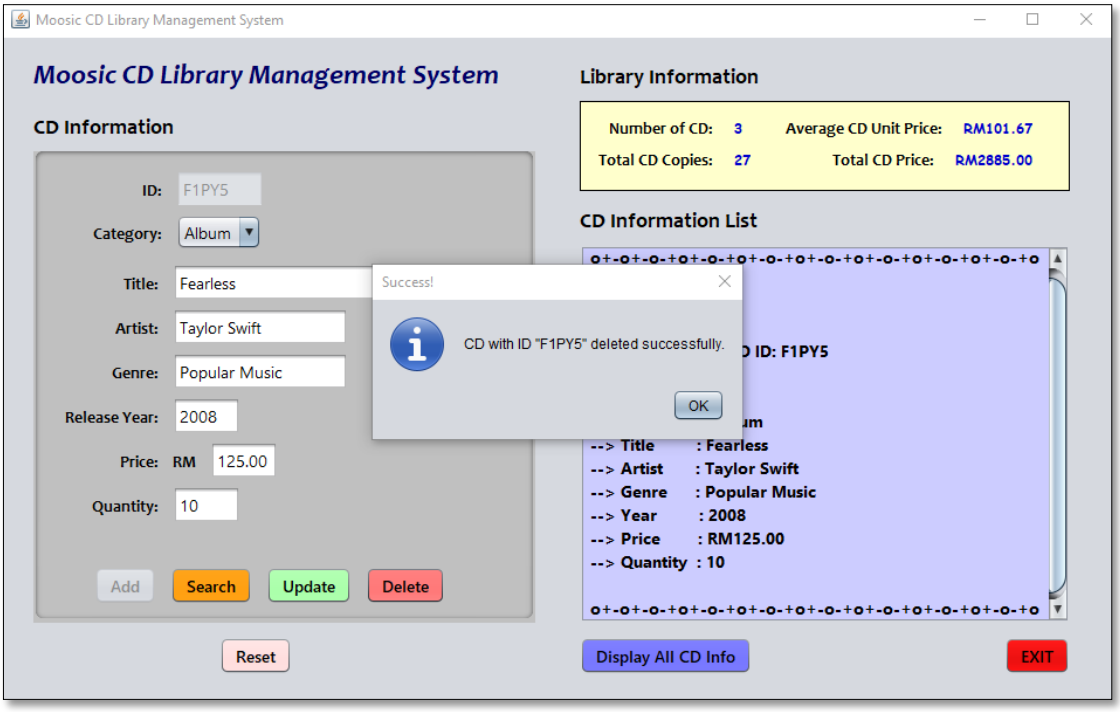
System will display a confirmation message if user wants to update the CD information.



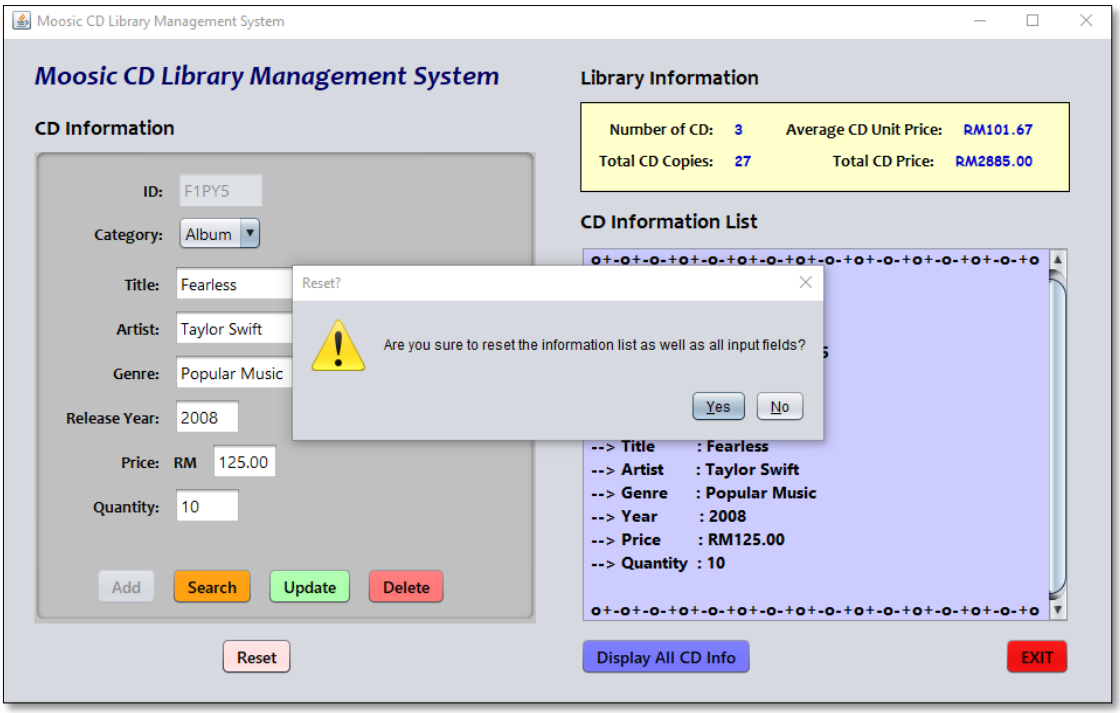
A “Success!” message will be displayed once the CD information has been updated.



If user clicks on the “Delete” button, program will pop up a confirmation message to ask user whether or not to delete the searched CD information.



If user clicks the "Yes" button, program will delete the CD information from the file and displays the above message.



The "Reset" button will reset the "CD Information List" text area and all the input fields.

To exit the program, click on the “Exit” button and select “Yes”.

4.0 SOURCE CODE

Attached below is the complete source code separated into three parts that represent the three Java classes used in this program:

- *MoosicGUI.java*
- *CDLibraryOperations.java*
- *CD.java*

--- MoosicGUI.java

```
/**

* STIA1123(E) Programming 2
* Group Project
*
* This program is an enhanced version of Lee Juan's Individual Assignment 1,
providing a user-friendly
* graphical user interface (GUI) to manage a collection of CDs in the Moosic CD
Library. The GUI is built
* using JFrame and offers various functionalities to interact with the CD Library.
*
* Features:
* - Display, Add, Edit, Delete Function
* - Search Function: Users can search for CDs by their unique ID and perform
editing or removal.
* - Unlimited CDs: Users can add an unlimited number of CDs to the library with
relevant information.
* - Local File Storage: CD information is saved in a local text file, allowing data
persistence between
*                       program runs, and automatically loading the library's
latest information upon startup.
* - Real Time Display: The GUI always displays the latest information from the CD
Library.
* - Exception Handling: The program provides helpful error messages in case of
invalid inputs or any file-related
*                       issues, ensuring a smooth and user-friendly experience.
*
* @author   Lee Juan (280027), Edwin Lim Wei Bin (281775)
* @version  1.0
* @since    2023-07-24
*/

package moosic.cd.library.management.system.project;

import java.time.Year;              //import Year class from .time package to
obtain current year
import javax.swing.JOptionPane;

public class MoosicGUI extends javax.swing.JFrame {
```

```
    private CD cd;
    private CDLibraryOperations op;


    public MoosicGUI() {
        initComponents();
        op = new CDLibraryOperations();

        op.loadCDDataFromFile();        //load data from file on startup

        //set labels in panel under "Library Information"
        numOfCDLabel.setText(String.valueOf(op.getNumOfCD()));
        totalCDCopiesLabel.setText(String.valueOf(op.calculateTotalCDCopies()));
        averageUnitPriceLabel.setText(String.format("RM%.2f",
op.calculateAverageUnitPrice()));
        totalValueLabel.setText(String.format("RM%.2f",
op.calculateTotalCDValue()));
    }


    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jLabel3 = new javax.swing.JLabel();
        titleTF = new javax.swing.JTextField();
        jLabel4 = new javax.swing.JLabel();
        artistTF = new javax.swing.JTextField();
        jLabel5 = new javax.swing.JLabel();
        priceTF = new javax.swing.JTextField();
        genreTF = new javax.swing.JTextField();
        jLabel7 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        yearTF = new javax.swing.JTextField();
        jLabel9 = new javax.swing.JLabel();
        categoryCB = new javax.swing.JComboBox<>();
        jLabel10 = new javax.swing.JLabel();
        quantityTF = new javax.swing.JTextField();
        jLabel6 = new javax.swing.JLabel();
        idTF = new javax.swing.JTextField();
        jLabel15 = new javax.swing.JLabel();
        addBTN = new javax.swing.JButton();
        searchBTN = new javax.swing.JButton();
        updateBTN = new javax.swing.JButton();
        deleteBTN = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        outputTA = new javax.swing.JTextArea();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        displayBTN = new javax.swing.JButton();
        resetBTN = new javax.swing.JButton();
        exitBTN = new javax.swing.JButton();
```

```
        jPanel2 = new javax.swing.JPanel();
        jLabel11 = new javax.swing.JLabel();
        jLabel12 = new javax.swing.JLabel();
        jLabel13 = new javax.swing.JLabel();
        jLabel14 = new javax.swing.JLabel();
        numOfCDLabel = new javax.swing.JLabel();
        totalCDCopiesLabel = new javax.swing.JLabel();
        totalValueLabel = new javax.swing.JLabel();
        averageUnitPriceLabel = new javax.swing.JLabel();
        jLabel16 = new javax.swing.JLabel();
        jLabel20 = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("Moosic CD Library Management System");

        jPanel1.setBackground(java.awt.Color.lightGray);
        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(""));
        jPanel1.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N

        jLabel3.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel3.setText("Title:");

        titleTF.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

        jLabel4.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel4.setText("Artist:");

        artistTF.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

        jLabel5.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel5.setText("Price:");

        priceTF.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

        genreTF.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

        jLabel7.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel7.setText("Genre:");

        jLabel8.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel8.setText("Release Year:");

        yearTF.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

        jLabel9.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel9.setText("Category:");

        categoryCB.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N
        categoryCB.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] {
"Album", "EP", "Single" }));

        jLabel10.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel10.setText("Quantity:");

        quantityTF.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N

        jLabel6.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel6.setText("ID:");
```

```
        idTF.setFont(new java.awt.Font("Segoe UI", 0, 14)); // NOI18N
        idTF.setEnabled(false);

        jLabel15.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel15.setText("RM");

        addBTN.setBackground(new java.awt.Color(255, 255, 153));
        addBTN.setFont(new java.awt.Font("Segoe UI Semibold", 1, 14)); // NOI18N
        addBTN.setText("Add");
        addBTN.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                addBTNActionPerformed(evt);
            }
        });

        searchBTN.setBackground(new java.awt.Color(255, 153, 0));
        searchBTN.setFont(new java.awt.Font("Segoe UI Semibold", 1, 14)); // NOI18N
        searchBTN.setText("Search");
        searchBTN.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                searchBTNActionPerformed(evt);
            }
        });

        updateBTN.setBackground(new java.awt.Color(153, 255, 153));
        updateBTN.setFont(new java.awt.Font("Segoe UI Semibold", 1, 14)); // NOI18N
        updateBTN.setText("Update");
        updateBTN.setEnabled(false);
        updateBTN.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                updateBTNActionPerformed(evt);
            }
        });

        deleteBTN.setBackground(new java.awt.Color(255, 102, 102));
        deleteBTN.setFont(new java.awt.Font("Segoe UI Semibold", 1, 14)); // NOI18N
        deleteBTN.setText("Delete");
        deleteBTN.setEnabled(false);
        deleteBTN.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                deleteBTNActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(17, 17, 17)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addGap(24, 24, 24)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAIL
```

```
ING)
                                .addComponent(jLabel9)
                                .addComponent(jLabel6))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
                                .addComponent(idTF,
javax.swing.GroupLayout.PREFERRED_SIZE, 75, javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(categoryCB,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                        .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAIL
ING)
                                .addComponent(jLabel8)
                                .addComponent(jLabel7)
                                .addComponent(jLabel4)
                                .addComponent(jLabel3)
                                .addComponent(jLabel5)
                                .addComponent(jLabel10))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
                                .addComponent(quantityTF,
javax.swing.GroupLayout.PREFERRED_SIZE, 58, javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(artistTF,
javax.swing.GroupLayout.PREFERRED_SIZE, 150,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(genreTF,
javax.swing.GroupLayout.PREFERRED_SIZE, 150,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addComponent(yearTF,
javax.swing.GroupLayout.PREFERRED_SIZE, 58, javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGroup(jPanel1Layout.createSequentialGroup()
                                    .addComponent(jLabel15)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                                    .addComponent(priceTF,
javax.swing.GroupLayout.PREFERRED_SIZE, 58,
javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addComponent(titleTF,
javax.swing.GroupLayout.PREFERRED_SIZE, 280,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                        .addGroup(jPanel1Layout.createSequentialGroup()
                            .addGap(25, 25, 25)
                            .addComponent(addBTN)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                            .addComponent(searchBTN)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                            .addComponent(updateBTN)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
```

```
                        .addComponent(deleteBTN)))
                .addContainerGap(20, Short.MAX_VALUE))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)
                    .addComponent(jLabel6)
                    .addComponent(idTF, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)
                    .addComponent(jLabel9)
                    .addComponent(categoryCB,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)
                    .addComponent(jLabel3)
                    .addComponent(titleTF, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)
                    .addComponent(jLabel4)
                    .addComponent(artistTF, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)
                    .addComponent(jLabel7)
                    .addComponent(genreTF, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)
                    .addComponent(jLabel8)
                    .addComponent(yearTF, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
```

```
INE)
                        .addComponent(jLabel5)
                        .addComponent(priceTF, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(jLabel15))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)
                        .addComponent(jLabel10)
                        .addComponent(quantityTF,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 37,
Short.MAX_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)
                        .addComponent(addBTN)
                        .addComponent(searchBTN)
                        .addComponent(updateBTN)
                        .addComponent(deleteBTN))
                    .addContainerGap())
        );

        outputTA.setEditable(false);
        outputTA.setBackground(new java.awt.Color(204, 204, 255));
        outputTA.setColumns(20);
        outputTA.setFont(new java.awt.Font("Segoe UI", 1, 14)); // NOI18N
        outputTA.setRows(5);
        jScrollPane1.setViewportView(outputTA);

        jLabel1.setBackground(new java.awt.Color(0, 0, 51));
        jLabel1.setFont(new java.awt.Font("Candara", 3, 24)); // NOI18N
        jLabel1.setForeground(new java.awt.Color(0, 0, 102));
        jLabel1.setText("Moosic CD Library Management System");

        jLabel2.setFont(new java.awt.Font("Candara", 1, 18)); // NOI18N
        jLabel2.setText("CD Information List");

        displayBTN.setBackground(new java.awt.Color(102, 102, 255));
        displayBTN.setFont(new java.awt.Font("Segoe UI Semibold", 1, 14)); //
NOI18N
        displayBTN.setText("Display All CD Info");
        displayBTN.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                displayBTNActionPerformed(evt);
            }
        });

        resetBTN.setBackground(new java.awt.Color(255, 204, 204));
        resetBTN.setFont(new java.awt.Font("Segoe UI Semibold", 1, 14)); // NOI18N
        resetBTN.setText("Reset");
        resetBTN.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                resetBTNActionPerformed(evt);
            }
```

```
        });

        exitBTN.setBackground(new java.awt.Color(204, 0, 0));
        exitBTN.setFont(new java.awt.Font("Segoe UI Semibold", 1, 14)); // NOI18N
        exitBTN.setText("EXIT");
        exitBTN.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                exitBTNActionPerformed(evt);
            }
        });

        jPanel2.setBackground(new java.awt.Color(255, 255, 204));
        jPanel2.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(0, 0, 0)));

        jLabel11.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel11.setText("Number of CD:");

        jLabel12.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel12.setText("Total CD Copies:");

        jLabel13.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel13.setText("Total CD Value:");

        jLabel14.setFont(new java.awt.Font("Candara", 1, 14)); // NOI18N
        jLabel14.setText("Average CD Unit Price:");

        numOfCDLabel.setFont(new java.awt.Font("Comic Sans MS", 1, 12)); // NOI18N
        numOfCDLabel.setForeground(new java.awt.Color(0, 0, 204));
        numOfCDLabel.setText("jLabel16");

        totalCDCopiesLabel.setFont(new java.awt.Font("Comic Sans MS", 1, 12)); //
NOI18N
        totalCDCopiesLabel.setForeground(new java.awt.Color(0, 0, 204));
        totalCDCopiesLabel.setText("jLabel16");

        totalValueLabel.setFont(new java.awt.Font("Comic Sans MS", 1, 12)); //
NOI18N
        totalValueLabel.setForeground(new java.awt.Color(0, 0, 204));
        totalValueLabel.setText("jLabel16");

        averageUnitPriceLabel.setFont(new java.awt.Font("Comic Sans MS", 1, 12));
// NOI18N
        averageUnitPriceLabel.setForeground(new java.awt.Color(0, 0, 204));
        averageUnitPriceLabel.setText("jLabel16");

        javax.swing.GroupLayout jPanel2Layout = new
javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addGap(15, 15, 15)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAIL
ING)
                    .addComponent(jLabel12)
                    .addComponent(jLabel11))
```

```
                .addGap(18, 18, 18)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
                    .addComponent(totalCDCopiesLabel)
                    .addComponent(numOfCDLabel))
                .addGap(30, 30, 30)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addComponent(jLabel14)
                        .addGap(18, 18, 18)
                        .addComponent(averageUnitPriceLabel))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel2Layout.createSequentialGroup()
                        .addComponent(jLabel13)
                        .addGap(18, 18, 18)
                        .addComponent(totalValueLabel)))
                .addContainerGap(31, Short.MAX_VALUE))
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addGap(13, 13, 13)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)
                    .addComponent(jLabel11)
                    .addComponent(numOfCDLabel)
                    .addComponent(jLabel14)
                    .addComponent(averageUnitPriceLabel))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASEL
INE)
                    .addComponent(jLabel12)
                    .addComponent(totalCDCopiesLabel)
                    .addComponent(jLabel13)
                    .addComponent(totalValueLabel))
                .addContainerGap(14, Short.MAX_VALUE))
        );

        jLabel16.setFont(new java.awt.Font("Candara", 1, 18)); // NOI18N
        jLabel16.setText("Library Information");

        jLabel20.setFont(new java.awt.Font("Candara", 1, 18)); // NOI18N
        jLabel20.setText("CD Information");

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
                    .addGroup(layout.createSequentialGroup()
                        .addGap(27, 27, 27)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jLabel20)
                            .addComponent(jLabel1)))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(186, 186, 186)
                        .addComponent(resetBTN)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 25,
Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel16)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
false)
                        .addComponent(jLabel2)
                        .addComponent(jPanel2,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                        .addComponent(jScrollPane1)
                        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                            .addComponent(displayBTN)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(exitBTN))))
                .addContainerGap(23, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addGap(21, 21, 21)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel16)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 323,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel1)
                        .addGap(18, 18, 18)
```

```
                        .addComponent(jLabel20)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(resetBTN)
                    .addComponent(displayBTN)
                    .addComponent(exitBTN))
                .addContainerGap(23, Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>


    private void addBTNActionPerformed(java.awt.event.ActionEvent evt) {

        try {

            String category = String.valueOf(categoryCB.getSelectedItem());
            String title = titleTF.getText();
            String artist = artistTF.getText();
            String genre = genreTF.getText();
            String yearText = yearTF.getText();
            String priceText = priceTF.getText();
            String quantityText = quantityTF.getText();

            //throw new exception for empty text fields
            if ((title.equals("")) || (artist.equals("")) || (genre.equals(""))
                    || (yearText.equals("")) || (priceText.equals("")) ||
(quantityText.equals("")))
                throw new Exception("Please enter all fields.");

            //potential statement with NumberFormatException
            int year = Integer.parseInt(yearText);
            double price = Double.parseDouble(priceText);
            int quantity = Integer.parseInt(quantityText);

            //throw new exception for invalid year, price, or quantity
            if ((year < 0) || (year > Year.now().getValue()))
                throw new Exception("Invalid year!");
            if (price <= 0.0)
                throw new Exception("Invalid price!");
            if (quantity <= 0)
                throw new Exception("Invalid quantity!");


            String id = op.generateRandomID();

            CD cd = new CD(id, category, title, artist, genre, year, price,
quantity);   //instantiate new cd object with its attributes
            op.addCD(cd);

            JOptionPane.showMessageDialog(this, "CD added successfully.",
```

```
"Success!", JOptionPane.INFORMATION_MESSAGE);

            categoryCB.setSelectedIndex(0);
            titleTF.setText("");
            artistTF.setText("");
            genreTF.setText("");
            yearTF.setText("");
            priceTF.setText("");
            quantityTF.setText("");
            outputTA.setText("");

            numOfCDLabel.setText(String.valueOf(op.getNumOfCD()));

totalCDCopiesLabel.setText(String.valueOf(op.calculateTotalCDCopies()));
            averageUnitPriceLabel.setText(String.format("RM%.2f",
op.calculateAverageUnitPrice()));
            totalValueLabel.setText(String.format("RM%.2f",
op.calculateTotalCDValue()));

        }
        catch (NumberFormatException nfx) {
            JOptionPane.showMessageDialog(this,
                    "Invalid entry. Please enter digits for year, price, and
quantity only.",
                   "Error!", JOptionPane.ERROR_MESSAGE);
        }
        catch (Exception x) {
            JOptionPane.showMessageDialog(this, x.getMessage(), "Error!",
JOptionPane.ERROR_MESSAGE);
        }

    }


    private void searchBTNActionPerformed(java.awt.event.ActionEvent evt) {

        if (op.isEmpty() == true) {
            JOptionPane.showMessageDialog(this, "Unable to search for any
CD.\nPlease add CD information.",
                    "CD Library is empty!", JOptionPane.INFORMATION_MESSAGE);
        }
        else {

            String id = JOptionPane.showInputDialog(this, "Insert CD ID:", "Search
CD", JOptionPane.QUESTION_MESSAGE);

            String cdInfo = op.displayCD(id);

            if (cdInfo.equals("ID not found"))
                JOptionPane.showMessageDialog(this,
                        "Unable to search for CD ID: " + id + "\nPlease insert a
valid CD ID.", "CD ID not found!", JOptionPane.ERROR_MESSAGE);

            else {
                outputTA.setText(cdInfo);
                JOptionPane.showMessageDialog(this,
                        "This CD info can now be viewed, edited or deleted.\n" +
                        "After editing the CD info in the respective input fields,
remember to save by clicking the Update button.\n" +
```

```
                        "Or, click the Delete button to remove CD.",
                        "CD ID found!", JOptionPane.INFORMATION_MESSAGE);

                //extract the attributes from cdInfo
                String category = cdInfo.substring(cdInfo.indexOf("Category  : ") +
12, cdInfo.indexOf("\n--> Title"));
                String title = cdInfo.substring(cdInfo.indexOf("Title         : ")
+ 16, cdInfo.indexOf("\n--> Artist"));
                String artist = cdInfo.substring(cdInfo.indexOf("Artist       : ")
+ 15, cdInfo.indexOf("\n--> Genre"));
                String genre = cdInfo.substring(cdInfo.indexOf("Genre      : ") +
13, cdInfo.indexOf("\n--> Year"));
                String year = cdInfo.substring(cdInfo.indexOf("Year         : ") +
15, cdInfo.indexOf("\n--> Price"));
                String price = cdInfo.substring(cdInfo.indexOf("Price        : RM")
+ 17, cdInfo.indexOf("\n--> Quantity"));
                String quantity = cdInfo.substring(cdInfo.indexOf("Quantity  : ") +
12, cdInfo.indexOf("\n\no"));

                //set the extracted attributes to the respective combo box and text
fields
                idTF.setText(id);
                categoryCB.setSelectedItem(category);
                titleTF.setText(title);
                artistTF.setText(artist);
                genreTF.setText(genre);
                yearTF.setText(year);
                priceTF.setText(price);
                quantityTF.setText(quantity);

                addBTN.setEnabled(false);
                updateBTN.setEnabled(true);
                deleteBTN.setEnabled(true);
            }

        } //end outer else

    }


    private void updateBTNActionPerformed(java.awt.event.ActionEvent evt) {

        //handle all the possible exceptions for user input
        try {

            int respond = JOptionPane.showConfirmDialog(this, "Are you sure to
update this CD info?",
                "Update Info?", JOptionPane.YES_NO_OPTION,
JOptionPane.QUESTION_MESSAGE);

            if (respond == JOptionPane.YES_OPTION) {

                String id = idTF.getText();
                String category = String.valueOf(categoryCB.getSelectedItem());
                String title = titleTF.getText();
                String artist = artistTF.getText();
                String genre = genreTF.getText();
                String yearText = yearTF.getText();
                String priceText = priceTF.getText();
```

```
                String quantityText = quantityTF.getText();

                //throw new exception for empty text fields
                if ((title.equals("")) || (artist.equals("")) || (genre.equals(""))
                        || (yearText.equals("")) || (priceText.equals("")) ||
(quantityText.equals("")))
                    throw new Exception("Please enter all fields.");

                //potential statement with NumberFormatException
                int year = Integer.parseInt(yearText);
                double price = Double.parseDouble(priceText);
                int quantity = Integer.parseInt(quantityText);

                //throw new exception for invalid year, price, or quantity
                if ((year <= 0) || (year > Year.now().getValue()))
                    throw new Exception("Invalid year!");
                if (price <= 0.0)
                    throw new Exception("Invalid price!");
                if (quantity <= 0)
                    throw new Exception("Invalid quantity!");

                CD updatedCD = new CD(id, category, title, artist, genre, year,
price, quantity);
                op.updateCD(updatedCD);

                JOptionPane.showMessageDialog(this, "CD with ID \"" + id + "\"
updated successfully.",
                        "Success!", JOptionPane.INFORMATION_MESSAGE);

                idTF.setText("");
                categoryCB.setSelectedIndex(0);
                titleTF.setText("");
                artistTF.setText("");
                genreTF.setText("");
                yearTF.setText("");
                priceTF.setText("");
                quantityTF.setText("");
                outputTA.setText("");

                addBTN.setEnabled(true);
                updateBTN.setEnabled(false);
                deleteBTN.setEnabled(false);

                numOfCDLabel.setText(String.valueOf(op.getNumOfCD()));

totalCDCopiesLabel.setText(String.valueOf(op.calculateTotalCDCopies()));
                averageUnitPriceLabel.setText(String.format("RM%.2f",
op.calculateAverageUnitPrice()));
                totalValueLabel.setText(String.format("RM%.2f",
op.calculateTotalCDValue()));

            }//end if

        }
        catch (NumberFormatException nfx) {
            JOptionPane.showMessageDialog(this,
                    "Invalid entry. Please enter digits for year, price, and
quantity only.",
                    "Error!", JOptionPane.ERROR_MESSAGE);
```

```
        }
        catch (Exception x) {
            JOptionPane.showMessageDialog(this, x.getMessage(), "Error!",
JOptionPane.ERROR_MESSAGE);
        }

    }


    private void deleteBTNActionPerformed(java.awt.event.ActionEvent evt) {

        int respond = JOptionPane.showConfirmDialog(this, "Are you sure to delete
this CD and its info?\nThis cannot be undone.",
                "Delete CD?", JOptionPane.YES_NO_OPTION,
JOptionPane.WARNING_MESSAGE);

        if (respond == JOptionPane.YES_OPTION) {

            String deleteID = idTF.getText();
            op.deleteCD(deleteID);

            JOptionPane.showMessageDialog(this, "CD with ID \"" + deleteID + "\"
deleted successfully.",
                        "Success!", JOptionPane.INFORMATION_MESSAGE);

            idTF.setText("");
            categoryCB.setSelectedIndex(0);
            titleTF.setText("");
            artistTF.setText("");
            genreTF.setText("");
            yearTF.setText("");
            priceTF.setText("");
            quantityTF.setText("");
            outputTA.setText("");

            addBTN.setEnabled(true);
            updateBTN.setEnabled(false);
            deleteBTN.setEnabled(false);

            numOfCDLabel.setText(String.valueOf(op.getNumOfCD()));

totalCDCopiesLabel.setText(String.valueOf(op.calculateTotalCDCopies()));
            averageUnitPriceLabel.setText(String.format("RM%.2f",
op.calculateAverageUnitPrice()));
            totalValueLabel.setText(String.format("RM%.2f",
op.calculateTotalCDValue()));

        }

    }


    private void displayBTNActionPerformed(java.awt.event.ActionEvent evt) {

        if (op.isEmpty() == true) {
            JOptionPane.showMessageDialog(this, "There is nothing to
display.\nPlease add CD information.",
                    "CD Library is empty!", JOptionPane.INFORMATION_MESSAGE);
        }
```

```
        else {
            String output = op.displayCD();
            outputTA.setText(output);

            idTF.setText("");
            categoryCB.setSelectedIndex(0);
            titleTF.setText("");
            artistTF.setText("");
            genreTF.setText("");
            yearTF.setText("");
            priceTF.setText("");
            quantityTF.setText("");

            addBTN.setEnabled(true);
            updateBTN.setEnabled(false);
            deleteBTN.setEnabled(false);
        }

    }


    private void resetBTNActionPerformed(java.awt.event.ActionEvent evt) {

        int respond = JOptionPane.showConfirmDialog(this, "Are you sure to reset
the information list as well as all input fields?",
                "Reset?", JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);

        if (respond == JOptionPane.YES_OPTION) {
            idTF.setText("");
            categoryCB.setSelectedIndex(0);
            titleTF.setText("");
            artistTF.setText("");
            genreTF.setText("");
            yearTF.setText("");
            priceTF.setText("");
            quantityTF.setText("");
            outputTA.setText("");

            addBTN.setEnabled(true);
            updateBTN.setEnabled(false);
            deleteBTN.setEnabled(false);
        }

    }


    private void exitBTNActionPerformed(java.awt.event.ActionEvent evt) {

        int respond = JOptionPane.showConfirmDialog(this, "Are you sure to exit?",
                "Exit?", JOptionPane.YES_NO_OPTION, JOptionPane.WARNING_MESSAGE);

        if (respond == JOptionPane.YES_OPTION)
            System.exit(0);

    }


    /**
     * @param args the command line arguments
```

```
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code
(optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the
default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(MoosicGUI.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(MoosicGUI.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(MoosicGUI.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(MoosicGUI.class.getName()).log(java.util.logging
.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new MoosicGUI().setVisible(true);
            }
        });
    }


    // Variables declaration - do not modify
    private javax.swing.JButton addBTN;
    private javax.swing.JTextField artistTF;
    private javax.swing.JLabel averageUnitPriceLabel;
    private javax.swing.JComboBox<String> categoryCB;
    private javax.swing.JButton deleteBTN;
    private javax.swing.JButton displayBTN;
    private javax.swing.JButton exitBTN;
    private javax.swing.JTextField genreTF;
    private javax.swing.JTextField idTF;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
```

```
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel13;
    private javax.swing.JLabel jLabel14;
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel16;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel20;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel numOfCDLabel;
    private javax.swing.JTextArea outputTA;
    private javax.swing.JTextField priceTF;
    private javax.swing.JTextField quantityTF;
    private javax.swing.JButton resetBTN;
    private javax.swing.JButton searchBTN;
    private javax.swing.JTextField titleTF;
    private javax.swing.JLabel totalCDCopiesLabel;
    private javax.swing.JLabel totalValueLabel;
    private javax.swing.JButton updateBTN;
    private javax.swing.JTextField yearTF;
    // End of variables declaration
}
```

--- CDLibraryOperations.java

```
package moosic.cd.library.management.system.project;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.Random;
import javax.swing.JOptionPane;

public class CDLibraryOperations {

    private ArrayList<CD> cds;

    //constructor
    public CDLibraryOperations() {
        cds = new ArrayList<>();          //ArrayList of CDs
    }
```

```
    //method to read CD data from the text file and populate the cds ArrayList
    public void loadCDDataFromFile() {

        cds.clear();

        try {

            File cdFile = new File("CDLibrary.txt");
            FileReader fr = new FileReader(cdFile);
            BufferedReader br = new BufferedReader(fr);

            String line = br.readLine();

            while (line != null) {

                //extract all eight CD attributes of a line when each ";" is
found in the line
                String[] parts = line.split(";");

                if (parts.length == 8) {

                    String id = parts[0];
                    String category = parts[1];
                    String title = parts[2];
                    String artist = parts[3];
                    String genre = parts[4];
                    int year = Integer.parseInt(parts[5]);
                    double price = Double.parseDouble(parts[6]);
                    int quantity = Integer.parseInt(parts[7]);

                    //instantiate cd object with data from the file and add into
cds ArrayList
                    CD cd = new CD(id, category, title, artist, genre, year,
price, quantity);
                    cds.add(cd);

                }

                line = br.readLine();

            } //end while

            br.close();

        }
        catch (IOException iox) {
            //if text file not found, continue to run program
            //program will create the file when a new CD info is added
        }

    } //end of loadCDDataFromFile()


    //return number of CDs in CD library when method is called
    public int getNumOfCD() {
        return cds.size();
    }
```

```
    //method to calculate total CD copies available in the CD library
    public int calculateTotalCDCopies() {
        if (cds.isEmpty() == true)
            return 0;
        else {
            int totalCopies = 0;

            for (CD cd : cds) {
                totalCopies += cd.getQuantity();
            }

            return totalCopies;
        }
    }


    //method to calculate average price per CD
    public double calculateAverageUnitPrice() {
        if (cds.isEmpty() == true)
            return 0.0;
        else {
            double total = 0;

            for (CD cd : cds) {
                total += cd.getPrice();
            }

            double average = total / cds.size();
            return average;
        }
    }


    //method to calculate total value of all CDs in the library
    public double calculateTotalCDValue() {
        if (cds.isEmpty() == true)
            return 0.0;
        else {
            double grandTotal = 0;

            for (CD cd : cds) {
                double total = cd.getPrice() * cd.getQuantity();
                grandTotal += total;
            }

            return grandTotal;
        }
    }


    //method to display all CD info onto output text area
    public String displayCD() {

        String text1, cdInfo;
        StringBuilder sb = new StringBuilder();

        text1 =
                "o+-o+-o-+o+-o-+o+-o-+o+-o-+o+-o-+o+-o-+o+-o-+o\n\n" +
                "--- ALL CD INFORMATION ---\n\n"
```

```
        ;

        sb.append(text1);

        int i = 1;

        //using enhanced for-each loop to display all CD info until end of
ArrayList
        for (CD cd : cds) {

            cdInfo =
                    "CD no." + i + "\n" +
                    "--> ID              : " + cd.getId() + "\n" +
                    "--> Category  : " + cd.getCategory() + "\n" +
                    "--> Title           : " + cd.getTitle() + "\n" +
                    "--> Artist        : " + cd.getArtist() + "\n" +
                    "--> Genre       : " + cd.getGenre() + "\n" +
                    "--> Year          : " + cd.getYear() + "\n" +
                    String.format("--> Price         : RM%.2f%n", + cd.getPrice())
+
                    "--> Quantity  : " + cd.getQuantity() + "\n\n"
            ;

            sb.append(cdInfo);
            i++;

            if (i > cds.size())
                break;

        } //end for-each loop

        sb.append("o+-o+-o-+o+-o-+o+-o-+o+-o-+o+-o-+o+-o-+o+-o-+o");

        return sb.toString();

    } //end of displayCD()


    //overridden method of displayCD(), to display a particular CD info after
searching for a CD ID
    public String displayCD(String id) {

        String cdInfo;
        int i=1;

        for (CD cd : cds) {
            if (cd.getId().equalsIgnoreCase(id)) {
                cdInfo =
                        "o+-o+-o-+o+-o-+o+-o-+o+-o-+o+-o-+o+-o-+o+-o-+o\n\n" +
                        "--- SEARCH CD ---\n\n" +
                        "You searched for CD ID: " + cd.getId() + "\n\n" +
                        "CD no." + i + "\n" +
                        "--> Category  : " + cd.getCategory() + "\n" +
                        "--> Title           : " + cd.getTitle() + "\n" +
                        "--> Artist        : " + cd.getArtist() + "\n" +
                        "--> Genre       : " + cd.getGenre() + "\n" +
                        "--> Year          : " + cd.getYear() + "\n" +
                        String.format("--> Price         : RM%.2f%n", +
cd.getPrice()) +
```

```
                        "--> Quantity  : " + cd.getQuantity() + "\n\n" +
                        "o+-o+-o-+o+-o-+o+-o-+o+-o-+o+-o-+o+-o-+o+-o-+o"
                ;

                return cdInfo;
            }

            i++;
        }

        cdInfo = "ID not found";
        return cdInfo;

    } //end of displayCD(String)


    //method to add cd object into cds ArrayList and save into file
    public void addCD(CD cd) {

        try {

            cds.add(cd);

            File cdFile = new File("CDLibrary.txt");
            FileWriter fw = new FileWriter(cdFile, true);
            PrintWriter pw = new PrintWriter(fw);

            pw.println(cd.getId() + ";" + cd.getCategory() + ";" + cd.getTitle()
+ ";" + cd.getArtist() + ";"
                        + cd.getGenre() + ";" + cd.getYear() + ";" +
cd.getPrice() + ";" + cd.getQuantity());

            pw.close();

        }
        catch (IOException iox) {
            JOptionPane.showMessageDialog(null, "Error adding CD data to file: "
+ iox.getMessage(),
                    "Error!", JOptionPane.ERROR_MESSAGE);
        }

    } //end of addCD(CD)


    //method to update CD info in ArrayList and save into file
    public void updateCD(CD updatedCD) {

        try {

            String searchID = updatedCD.getId();

            //locate the respective CD info in the ArrayList
            for (int i=0; i<cds.size(); i++) {

                //if found, update the existing CD object in ArrayList and save
into file
                //.equalsIgnoreCase() as user input of ID could be in no caps
                if (cds.get(i).getId().equalsIgnoreCase(searchID)) {
```

```
                    cds.get(i).setCategory(updatedCD.getCategory());
                    cds.get(i).setTitle(updatedCD.getTitle());
                    cds.get(i).setArtist(updatedCD.getArtist());
                    cds.get(i).setGenre(updatedCD.getGenre());
                    cds.get(i).setYear(updatedCD.getYear());
                    cds.get(i).setPrice(updatedCD.getPrice());
                    cds.get(i).setQuantity(updatedCD.getQuantity());

                    File cdFile = new File("CDLibrary.txt");
                    FileWriter fw = new FileWriter(cdFile);
                    PrintWriter pw = new PrintWriter(fw);

                    for (CD cd : cds) {
                        pw.println(cd.getId() + ";" + cd.getCategory() + ";" +
cd.getTitle() + ";" + cd.getArtist() + ";"
                                + cd.getGenre() + ";" + cd.getYear() + ";" +
cd.getPrice() + ";" + cd.getQuantity());
                    }

                    pw.close();
                    break;      //exit loop once CD is updated

                }

            } //end for loop

        }
        catch (IOException iox) {
            JOptionPane.showMessageDialog(null, "Error updating CD data to file:
" + iox.getMessage(),
                    "Error!", JOptionPane.ERROR_MESSAGE);
        }

    } //end of updateCD(CD)


    //method to delete a cd object in cds ArrayList and remove from file
    public void deleteCD(String deleteID) {

        try {

            //traverse the ArrayList to find the matching CD to remove
            for (int i=0; i<cds.size(); i++) {

                if (cds.get(i).getId().equalsIgnoreCase(deleteID)) {

                    cds.remove(i);

                    File cdFile = new File("CDLibrary.txt");
                    FileWriter fw = new FileWriter(cdFile);
                    PrintWriter pw = new PrintWriter(fw);

                    for (CD cd : cds) {
                        pw.println(cd.getId() + ";" + cd.getCategory() + ";" +
cd.getTitle() + ";" + cd.getArtist() + ";"
                                + cd.getGenre() + ";" + cd.getYear() + ";" +
cd.getPrice() + ";" + cd.getQuantity());
                    }
```

```
                    pw.close();
                    break;      //exit loop once CD is removed
                }

            } //end for loop

        }
        catch (IOException iox) {
            JOptionPane.showMessageDialog(null, "Error deleting CD data from
file: " + iox.getMessage(),
                    "Error!", JOptionPane.ERROR_MESSAGE);
        }

    } //end of deleteCD(String)


    //method to return a randomly generated CD ID
    String generateRandomID() {

        String characters = "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"; //ID will
randomly generate from these characters
        StringBuilder sb = new StringBuilder();                    //sb object
from StringBuilder class to build the random ID string character by character
        Random random = new Random();                              //declare and
initialise random object

        final int indexNum = 5;                                    //fix the
number of random characters in ID to 5

        for (int i=0; i<indexNum; i++) {
            int index = random.nextInt(characters.length());   //generate index
value by random
            sb.append(characters.charAt(index));                  //from the
random generated index, append the character at the index value to the end of sb
        }

        return sb.toString();                                      //convert sb
to String and return the completed String ID

    } //end of generateRandomID()


    //method that returns boolean if cds ArrayList is empty
    public boolean isEmpty() {
        if (cds.isEmpty() == true)
            return true;
        else
            return false;
    }


} //end CDLibraryOperations.java class
```

--- CD.java

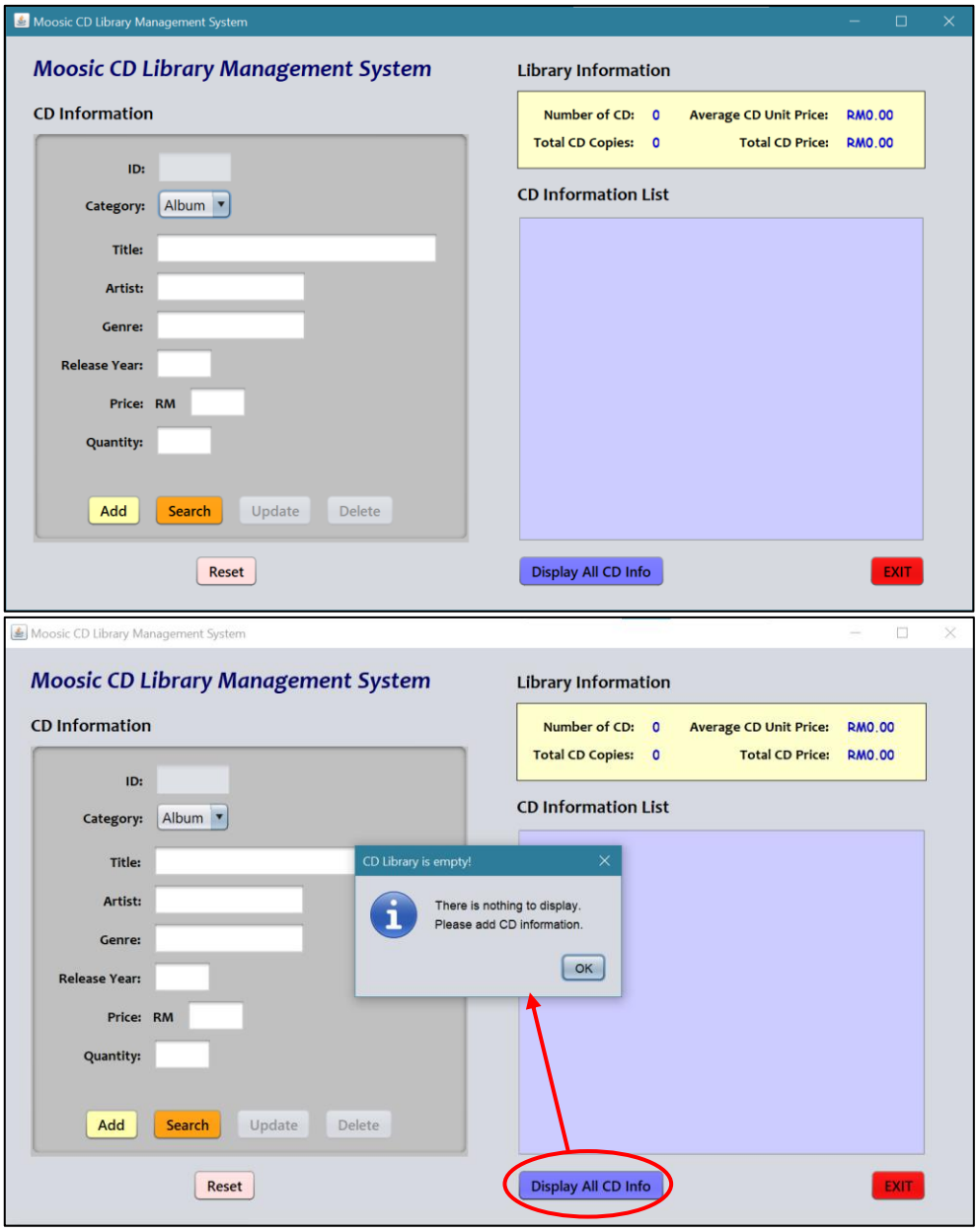
```
package moosic.cd.library.management.system.project;

public class CD {

    private final String id;
    private String category;
    private String title;
    private String artist;
    private String genre;
    private int year;
    private double price;
    private int quantity;


    //constructor to instantiate cd object and initialise its attributes
    public CD(String id, String category, String title, String artist, String
genre, int year, double price, int quantity) {
        this.id = id;
        this.category = category;
        this.title = title;
        this.artist = artist;
        this.genre = genre;
        this.year = year;
        this.price = price;
        this.quantity = quantity;
    }


    //mutators
    public void setCategory(String c) {
        category = c;
    }

    public void setTitle(String t) {
        title = t;
    }

    public void setArtist(String a) {
        artist = a;
    }

    public void setGenre(String g) {
        genre = g;
    }

    public void setYear(int y) {
        year = y;
    }

    public void setPrice(double p) {
        price = p;
    }

    public void setQuantity(int q) {
        quantity = q;
    }
```

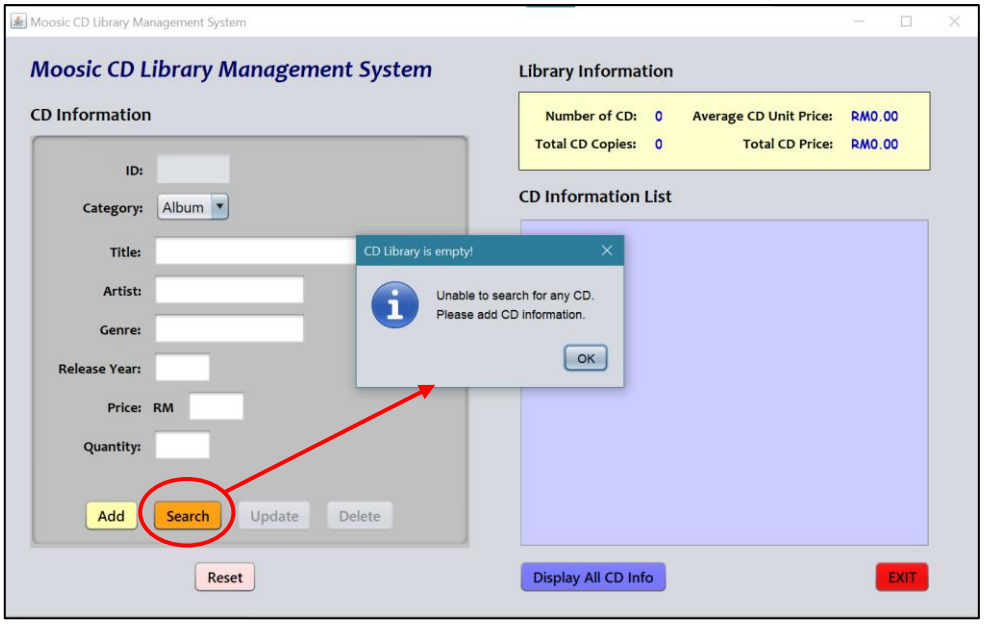
```
    //accessors
    public String getId() {
        return id;
    }

    public String getCategory() {
        return category;
    }

    public String getTitle() {
        return title;
    }

    public String getArtist() {
        return artist;
    }

    public String getGenre() {
        return genre;
    }

    public int getYear() {
        return year;
    }

    public double getPrice() {
        return price;
    }

    public int getQuantity() {
        return quantity;
    }


} //end CD.java class
```

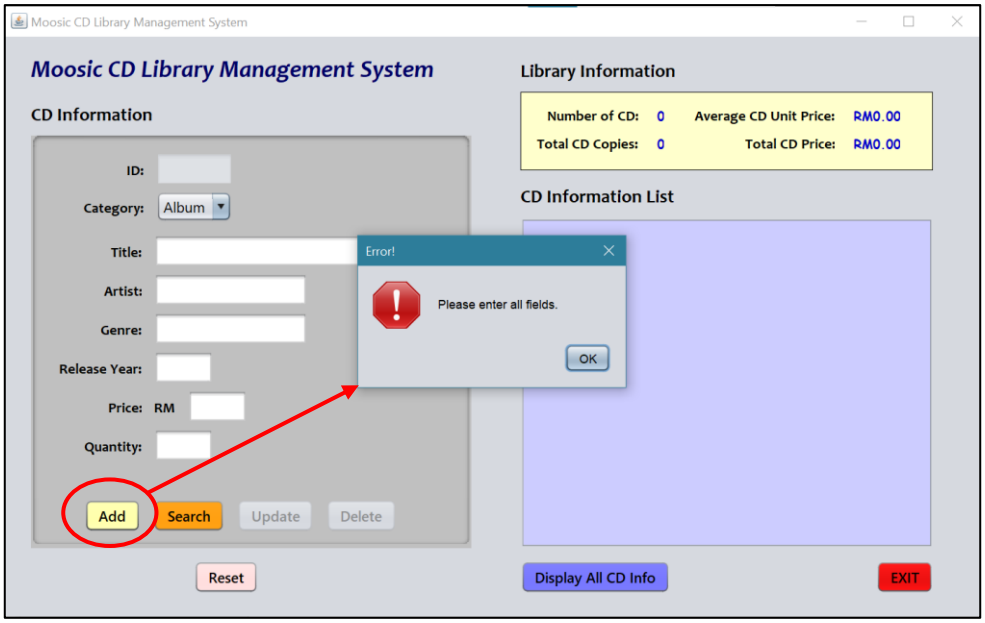
5.0 SAMPLE RUN

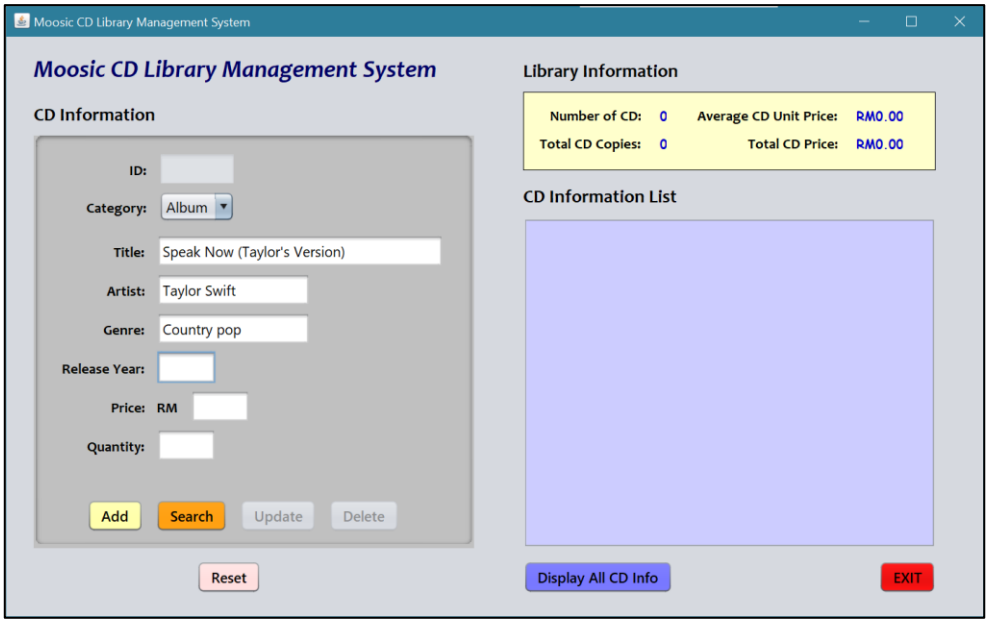
Below are screenshots of two sample runs when the GUI form, *MoosicGUI.java* is being executed in Apache NetBeans IDE 17.

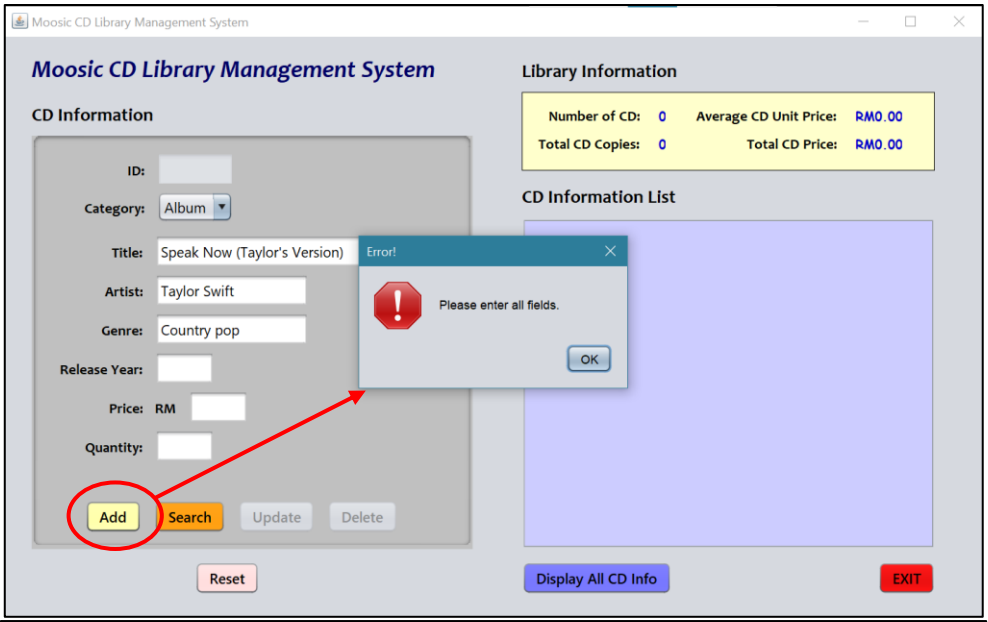
This is the sample run when the program is being run **for the first time** (red arrows and circles to show user actions being performed): -

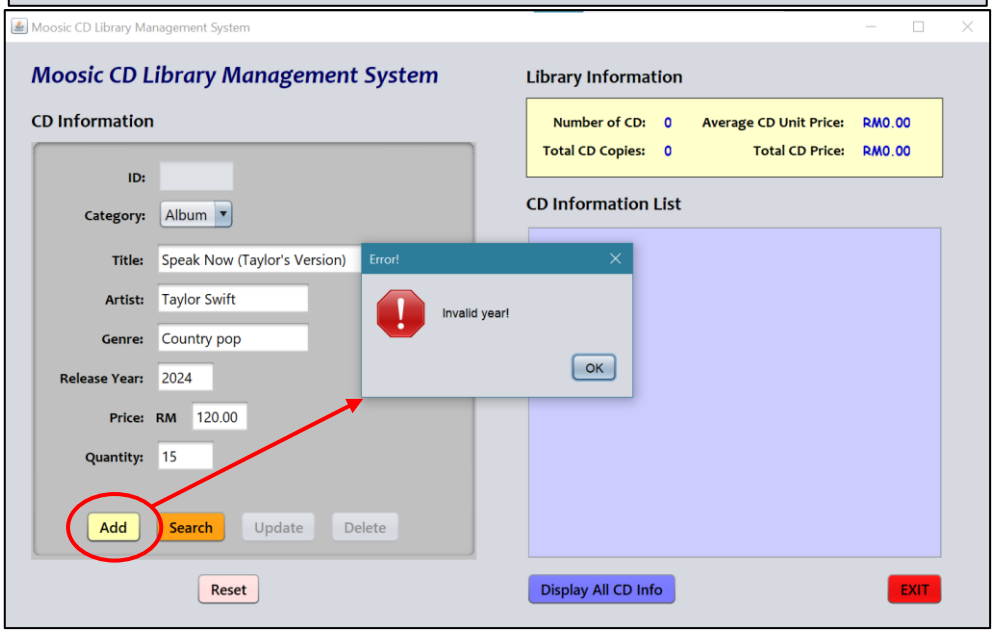


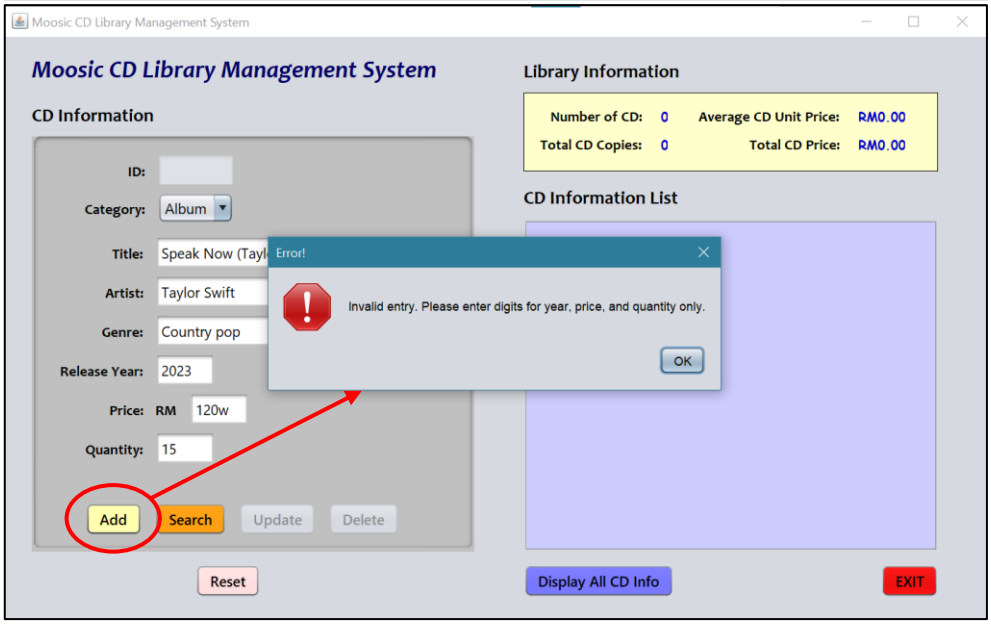

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title:
Artist:
Genre:
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 0
Average CD Unit Price: RM0.00
Total CD Copies: 0
Total CD Price: RM0.00
CD Information List
Display All CD Info
EXIT
CD Library is empty!
Unable to search for any CD.
Please add CD information.
OK

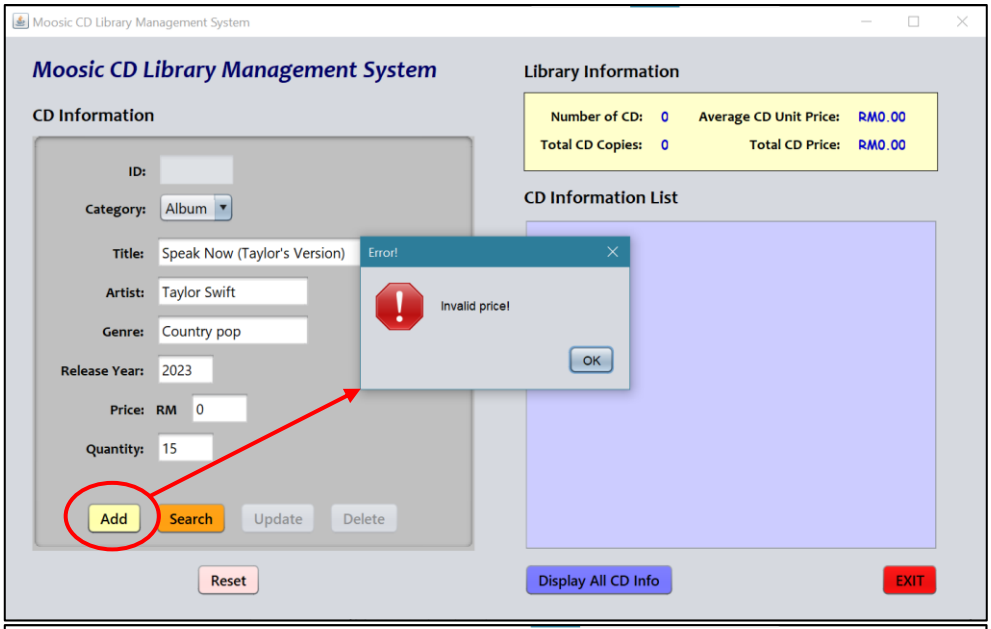

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title:
Artist:
Genre:
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 0
Average CD Unit Price: RM0.00
Total CD Copies: 0
Total CD Price: RM0.00
CD Information List
Display All CD Info
EXIT
Error!
Please enter all fields.
OK

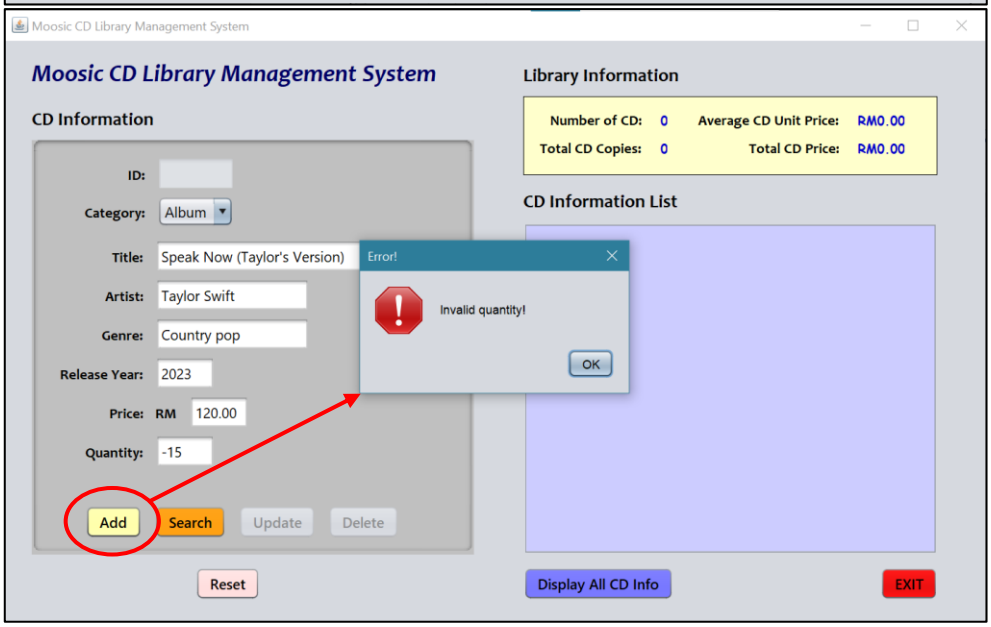

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title: Speak Now (Taylor's Version)
Artist: Taylor Swift
Genre: Country pop
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 0
Average CD Unit Price: RM0.00
Total CD Copies: 0
Total CD Price: RM0.00
CD Information List
Display All CD Info
EXIT

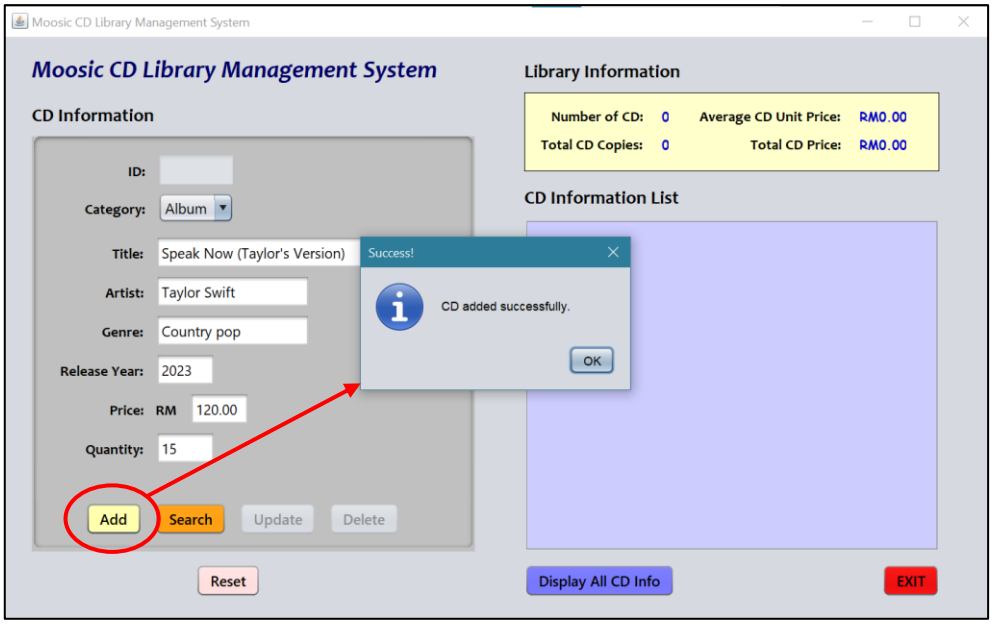

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title: Speak Now (Taylor's Version)
Artist: Taylor Swift
Genre: Country pop
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 0
Average CD Unit Price: RM0.00
Total CD Copies: 0
Total CD Price: RM0.00
CD Information List
Display All CD Info
EXIT
Error!
Please enter all fields.
OK

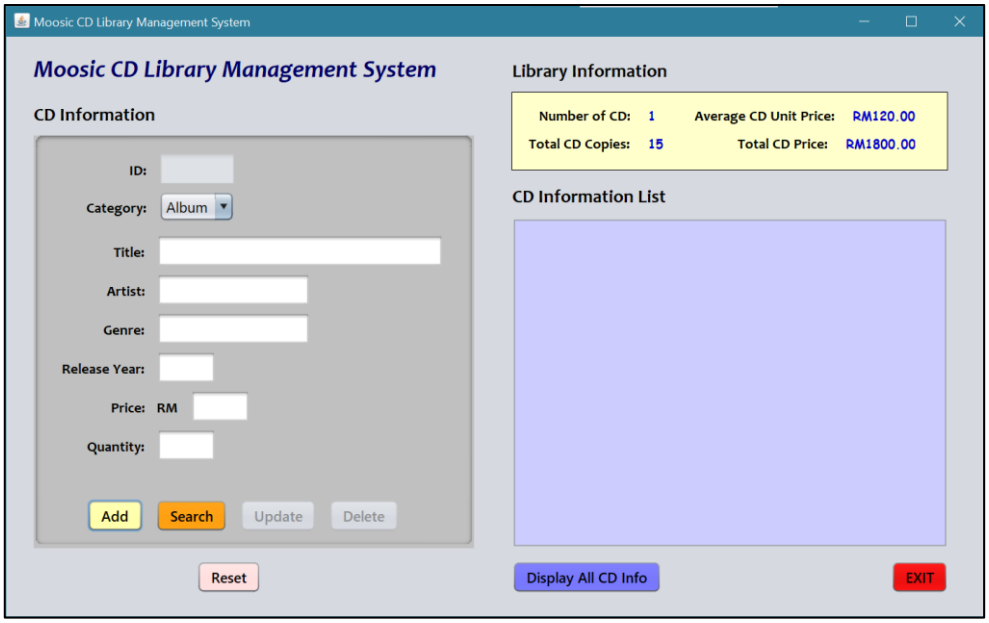

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title: Speak Now (Taylor's Version)
Artist: Taylor Swift
Genre: Country pop
Release Year: 2024
Price: RM 120.00
Quantity: 15
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 0
Average CD Unit Price: RM0.00
Total CD Copies: 0
Total CD Price: RM0.00
CD Information List
Display All CD Info
EXIT
Error!
Invalid year!
OK

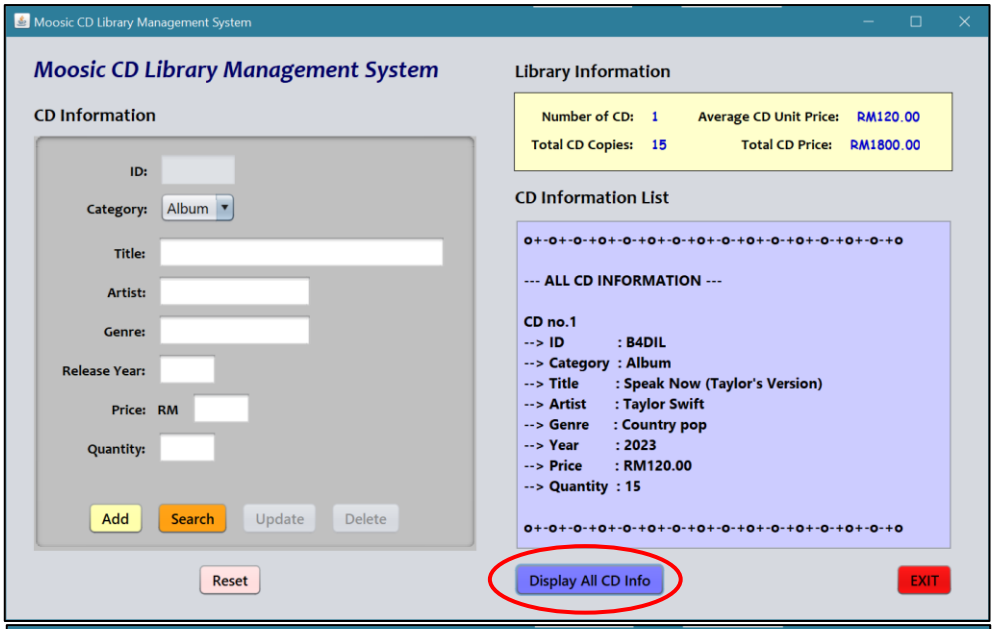

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title: Speak Now (Tayl
Artist: Taylor Swift
Genre: Country pop
Release Year: 2023
Price: RM 120w
Quantity: 15
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 0
Average CD Unit Price: RM0.00
Total CD Copies: 0
Total CD Price: RM0.00
CD Information List
Display All CD Info
EXIT
Error!
Invalid entry. Please enter digits for year, price, and quantity only.
OK

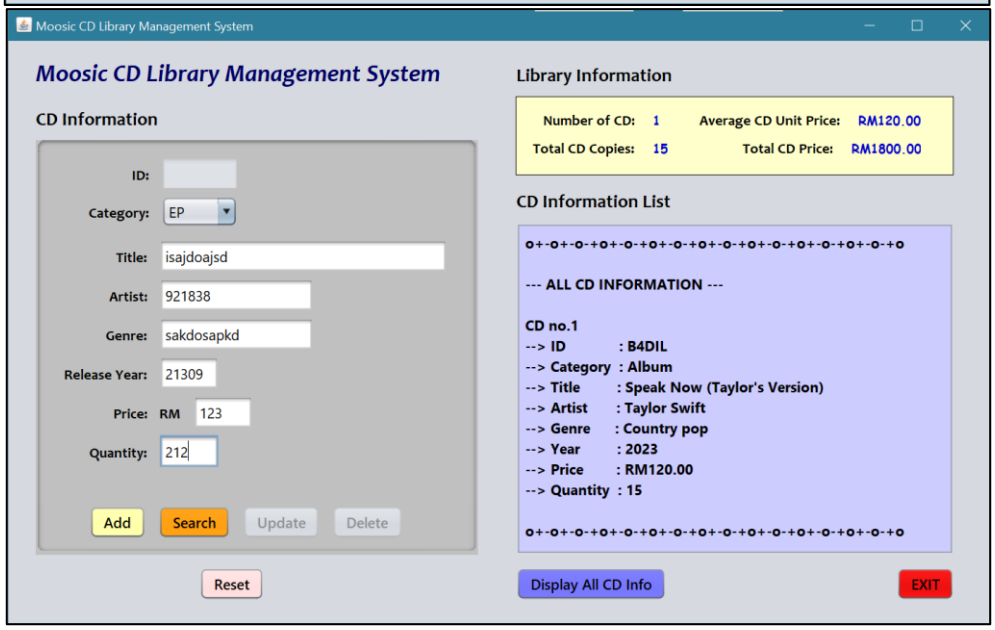

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title: Speak Now (Taylor's Version)
Artist: Taylor Swift
Genre: Country pop
Release Year: 2023
Price: RM 0
Quantity: 15
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 0
Average CD Unit Price: RM0.00
Total CD Copies: 0
Total CD Price: RM0.00
CD Information List
Display All CD Info
EXIT
Error!
Invalid price!
OK

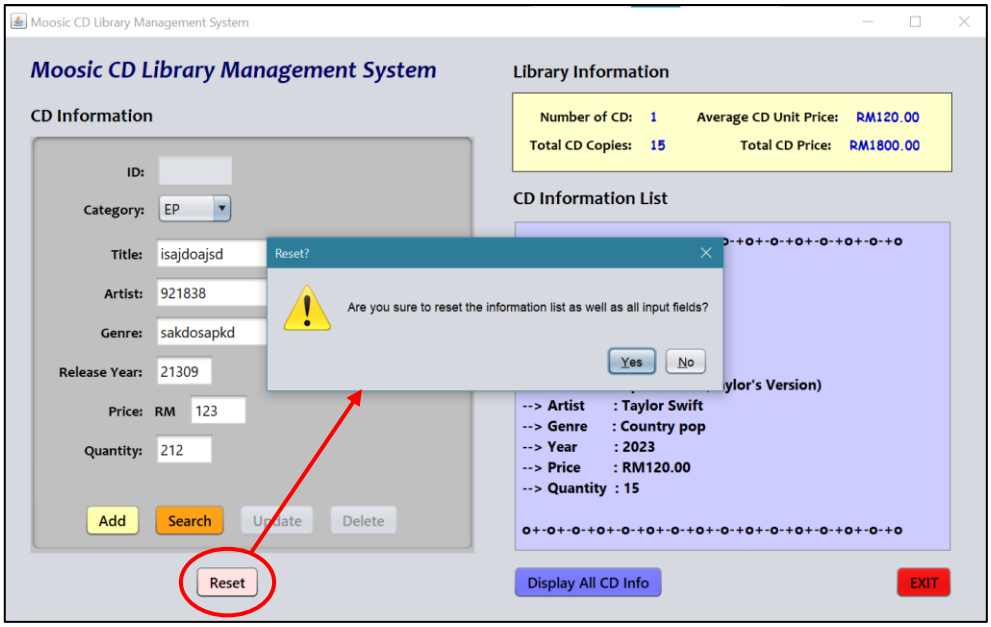

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title: Speak Now (Taylor's Version)
Artist: Taylor Swift
Genre: Country pop
Release Year: 2023
Price: RM 120.00
Quantity: -15
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 0
Average CD Unit Price: RM0.00
Total CD Copies: 0
Total CD Price: RM0.00
CD Information List
Display All CD Info
EXIT
Error!
Invalid quantity!
OK

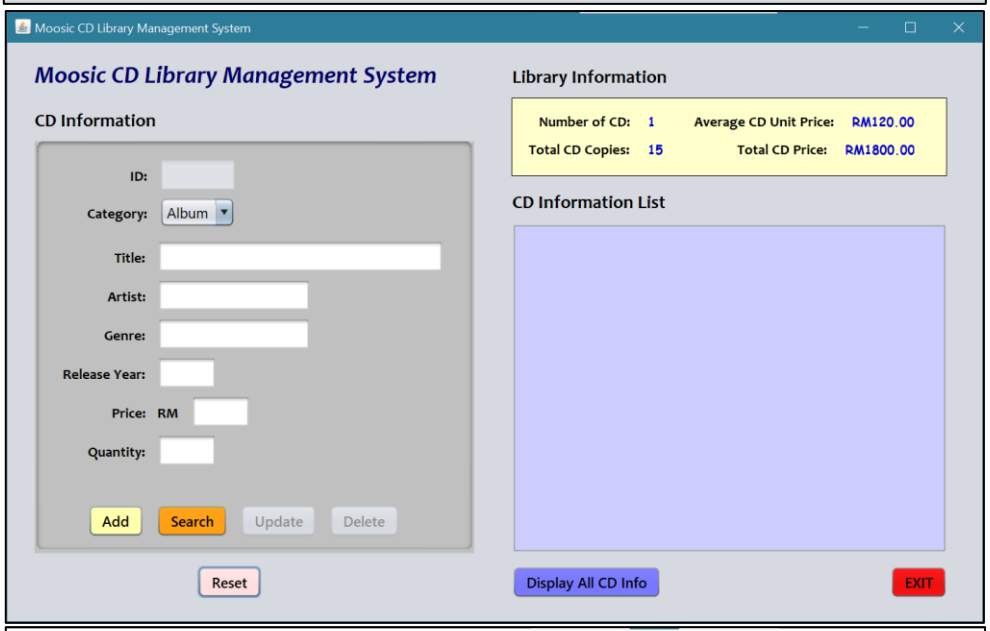

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title: Speak Now (Taylor's Version)
Artist: Taylor Swift
Genre: Country pop
Release Year: 2023
Price: RM 120.00
Quantity: 15
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 0
Average CD Unit Price: RM0.00
Total CD Copies: 0
Total CD Price: RM0.00
CD Information List
Display All CD Info
EXIT
Success!
CD added successfully.
OK

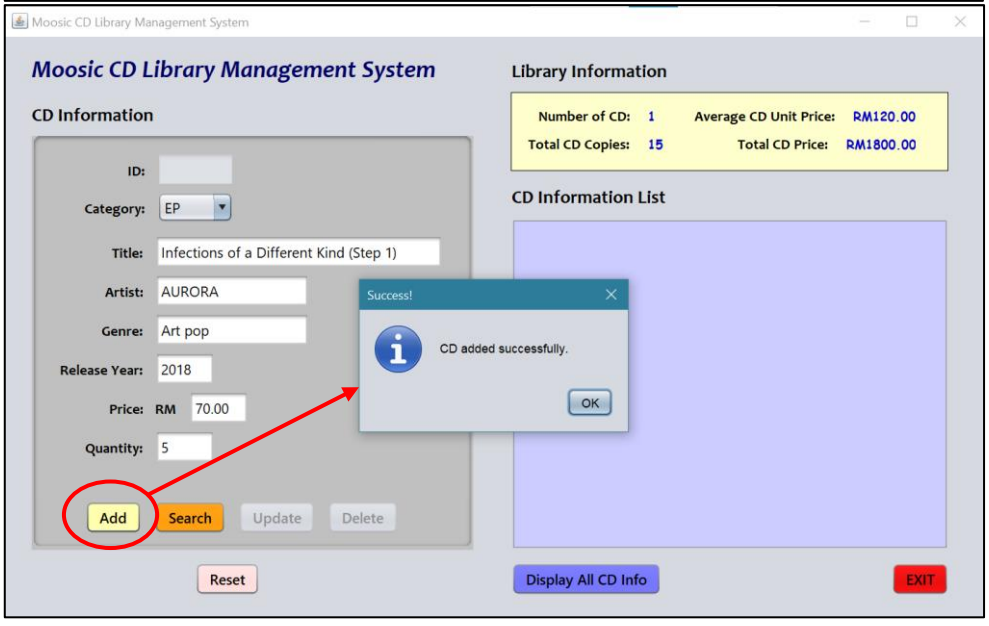

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title:
Artist:
Genre:
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 1
Average CD Unit Price: RM120.00
Total CD Copies: 15
Total CD Price: RM1800.00
CD Information List
Display All CD Info
EXIT

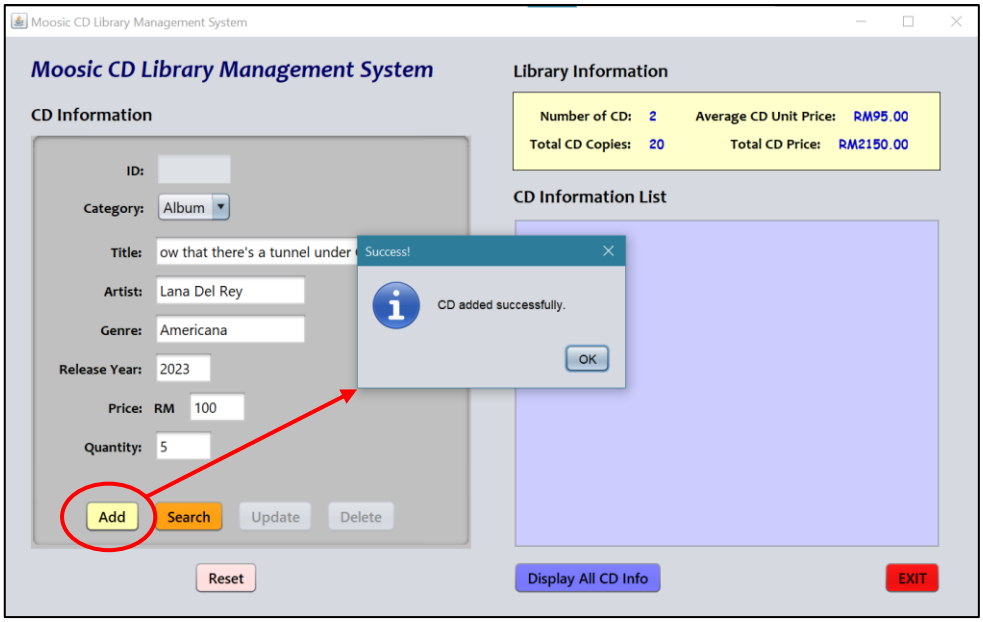

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title:
Artist:
Genre:
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 1
Average CD Unit Price: RM120.00
Total CD Copies: 15
Total CD Price: RM1800.00
CD Information List
--- ALL CD INFORMATION ---
CD no.1
--> ID : B4DIL
--> Category : Album
--> Title : Speak Now (Taylor's Version)
--> Artist : Taylor Swift
--> Genre : Country pop
--> Year : 2023
--> Price : RM120.00
--> Quantity : 15
Display All CD Info
EXIT

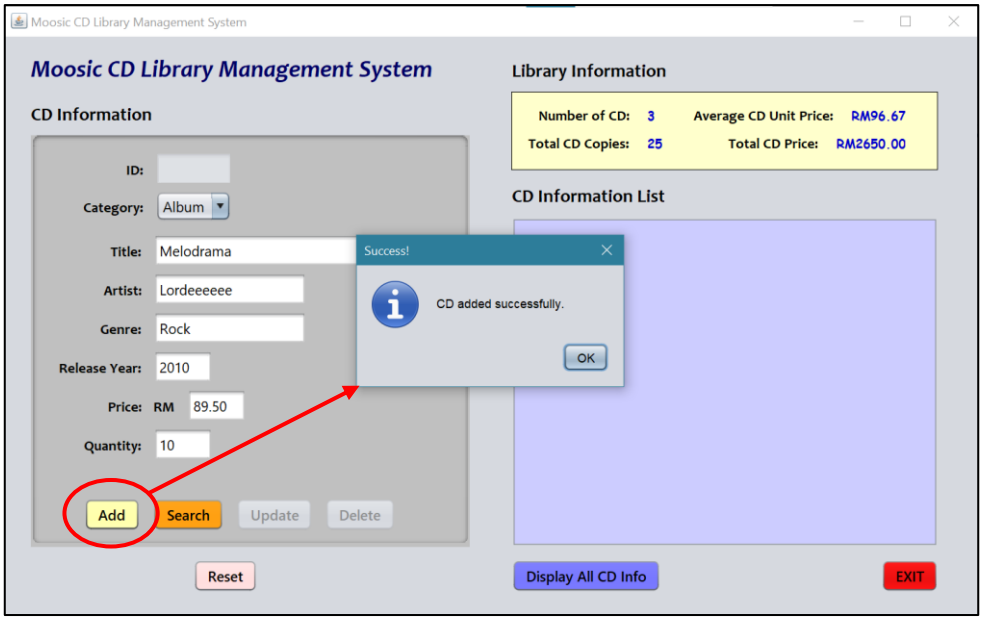

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: EP
Title: isajdoajsd
Artist: 921838
Genre: sakdosapkd
Release Year: 21309
Price: RM 123
Quantity: 212
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 1
Average CD Unit Price: RM120.00
Total CD Copies: 15
Total CD Price: RM1800.00
CD Information List
--- ALL CD INFORMATION ---
CD no.1
--> ID : B4DIL
--> Category : Album
--> Title : Speak Now (Taylor's Version)
--> Artist : Taylor Swift
--> Genre : Country pop
--> Year : 2023
--> Price : RM120.00
--> Quantity : 15
Display All CD Info
EXIT

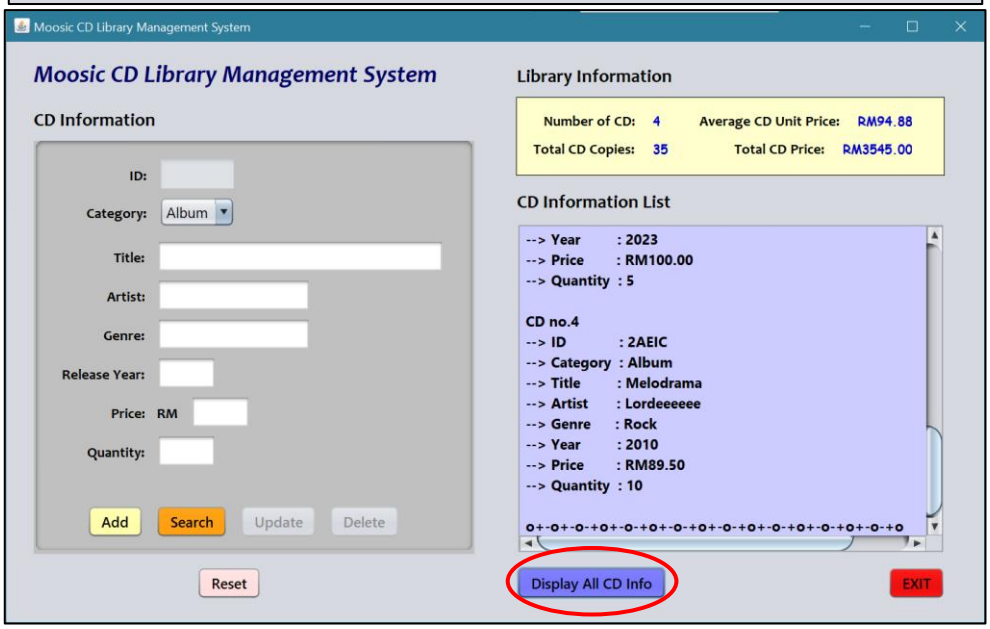

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: EP
Title: isajdoajsd
Artist: 921838
Genre: sakdosapkd
Release Year: 21309
Price: RM 123
Quantity: 212
Add
Search
Update
Delete
Reset
Reset?
Are you sure to reset the information list as well as all input fields?
Yes
No
Library Information
Number of CD: 1
Average CD Unit Price: RM120.00
Total CD Copies: 15
Total CD Price: RM1800.00
CD Information List
--> Artist : Taylor Swift
--> Genre : Country pop
--> Year : 2023
--> Price : RM120.00
--> Quantity : 15
Display All CD Info
EXIT

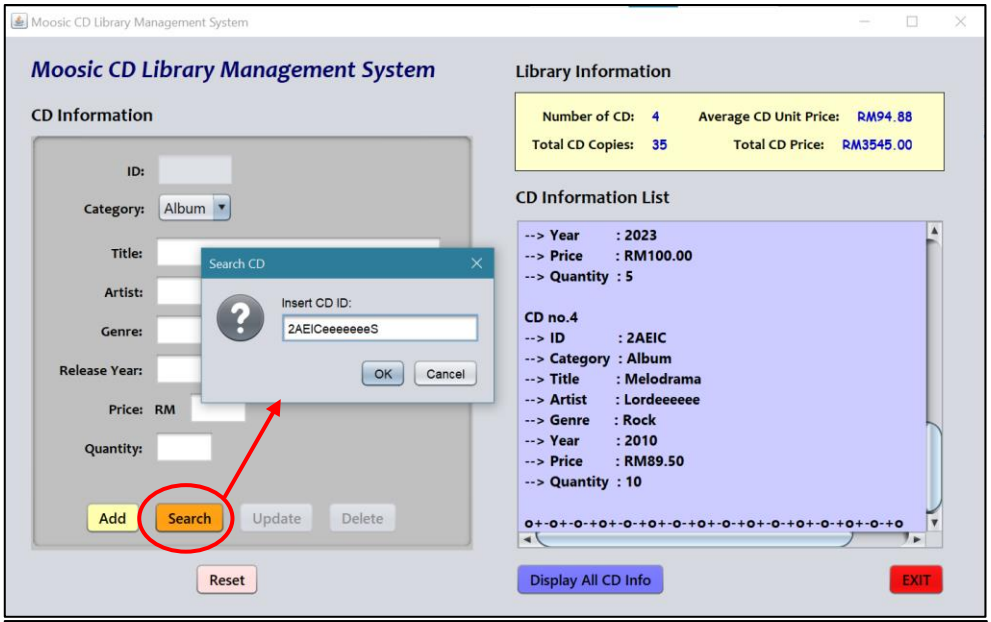

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title:
Artist:
Genre:
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 1
Average CD Unit Price: RM120.00
Total CD Copies: 15
Total CD Price: RM1800.00
CD Information List
Display All CD Info
EXIT

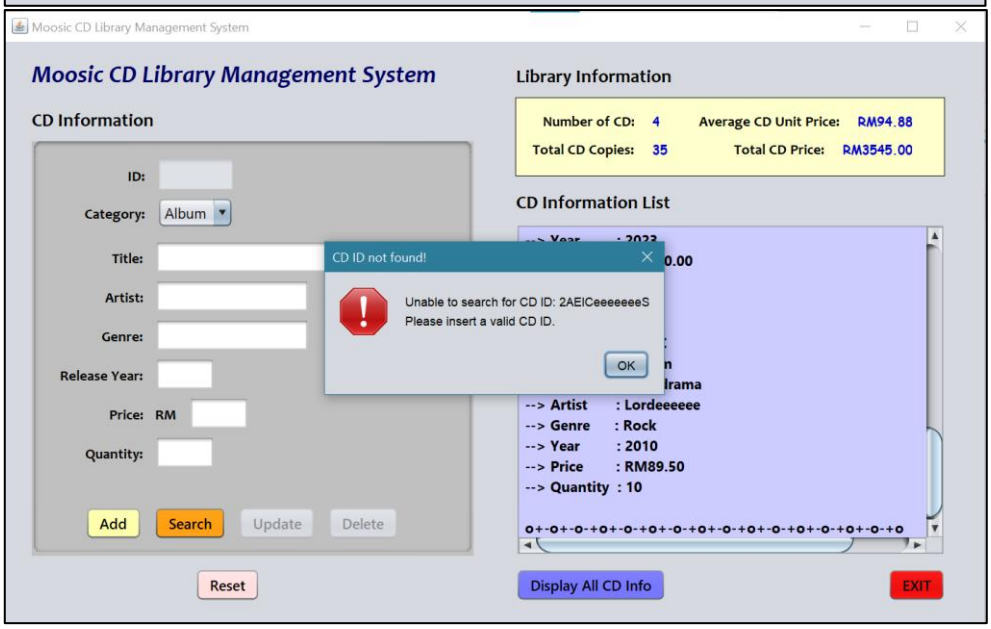

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: EP
Title: Infections of a Different Kind (Step 1)
Artist: AURORA
Genre: Art pop
Release Year: 2018
Price: RM 70.00
Quantity: 5
Add
Search
Update
Delete
Reset
Success!
CD added successfully.
OK
Library Information
Number of CD: 1
Average CD Unit Price: RM120.00
Total CD Copies: 15
Total CD Price: RM1800.00
CD Information List
Display All CD Info
EXIT

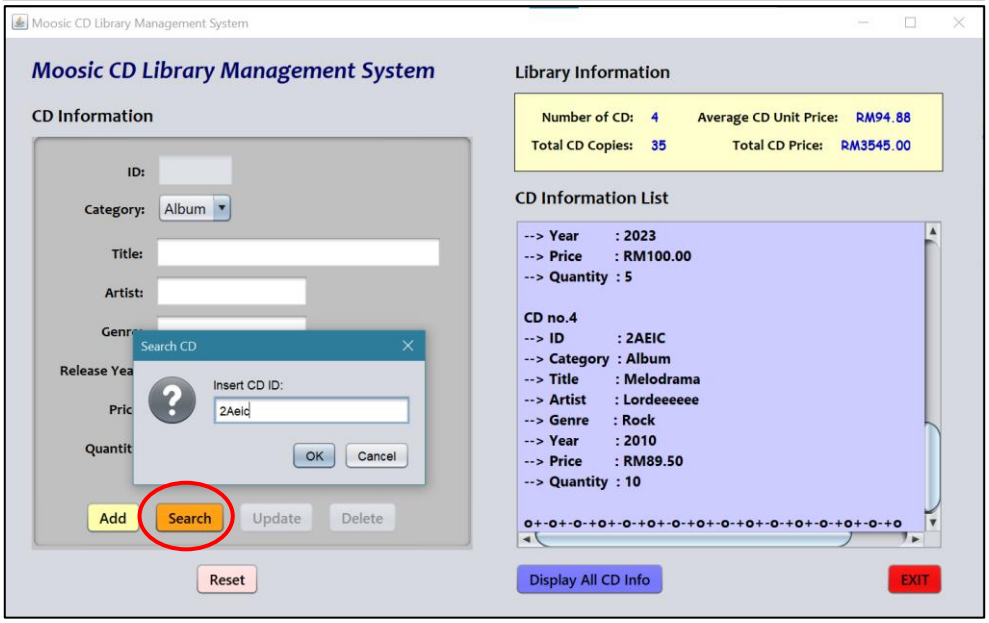

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title: ow that there's a tunnel under
Artist: Lana Del Rey
Genre: Americana
Release Year: 2023
Price: RM 100
Quantity: 5
Add
Search
Update
Delete
Reset
Success!
CD added successfully.
OK
Library Information
Number of CD: 2
Average CD Unit Price: RM95.00
Total CD Copies: 20
Total CD Price: RM2150.00
CD Information List
Display All CD Info
EXIT

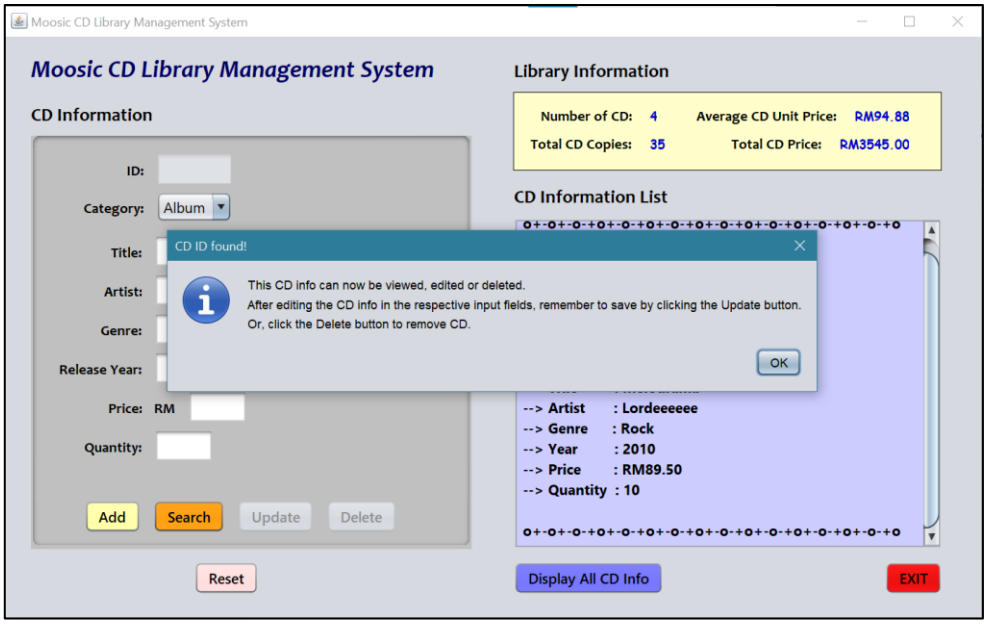

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title: Melodrama
Artist: Lordeeeeee
Genre: Rock
Release Year: 2010
Price: RM 89.50
Quantity: 10
Add
Search
Update
Delete
Reset
Success!
CD added successfully.
OK
Library Information
Number of CD: 3
Average CD Unit Price: RM96.67
Total CD Copies: 25
Total CD Price: RM2650.00
CD Information List
Display All CD Info
EXIT

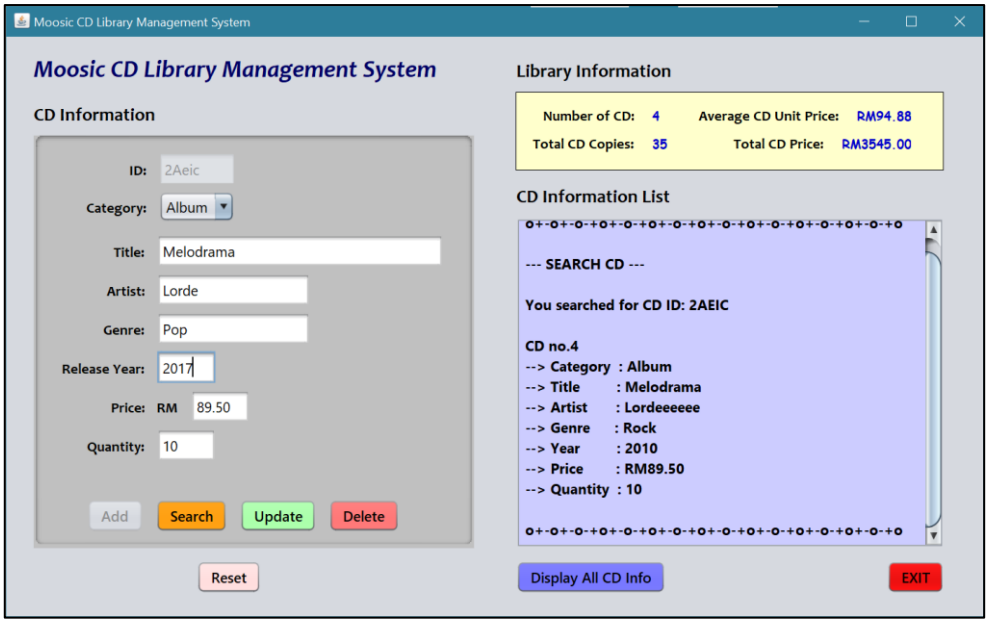

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title:
Artist:
Genre:
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 4
Average CD Unit Price: RM94.88
Total CD Copies: 35
Total CD Price: RM3545.00
CD Information List
--> Year : 2023
--> Price : RM100.00
--> Quantity : 5
CD no.4
--> ID : 2AEIC
--> Category : Album
--> Title : Melodrama
--> Artist : Lordeeeeee
--> Genre : Rock
--> Year : 2010
--> Price : RM89.50
--> Quantity : 10
Display All CD Info
EXIT

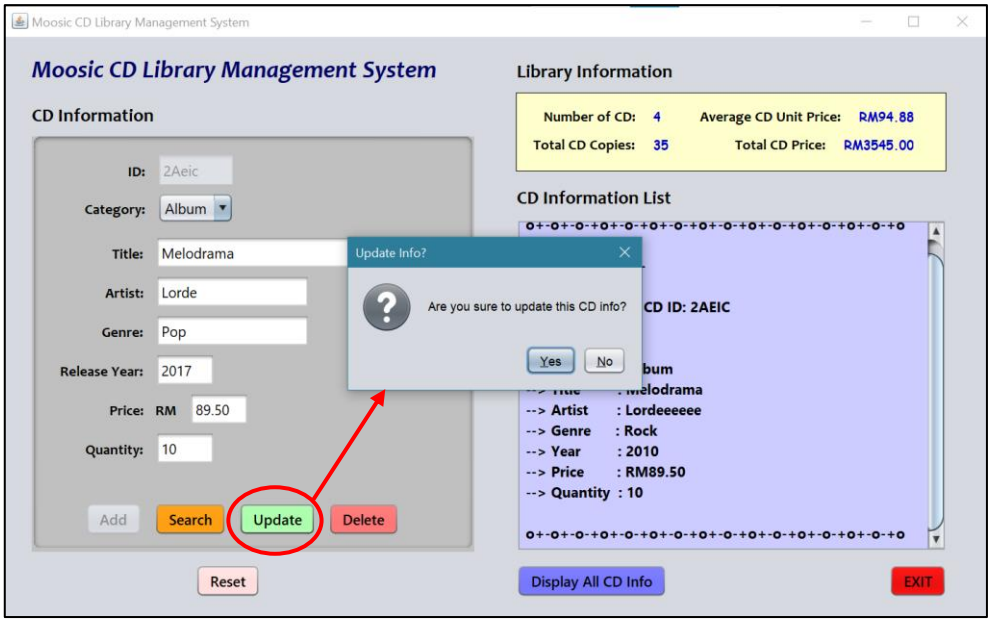

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title:
Artist:
Genre:
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
Search CD
Insert CD ID:
2AEICeeeeeeeS
OK
Cancel
Library Information
Number of CD: 4
Average CD Unit Price: RM94.88
Total CD Copies: 35
Total CD Price: RM3545.00
CD Information List
--> Year : 2023
--> Price : RM100.00
--> Quantity : 5
CD no.4
--> ID : 2AEIC
--> Category : Album
--> Title : Melodrama
--> Artist : Lordeeeeee
--> Genre : Rock
--> Year : 2010
--> Price : RM89.50
--> Quantity : 10
Display All CD Info
EXIT

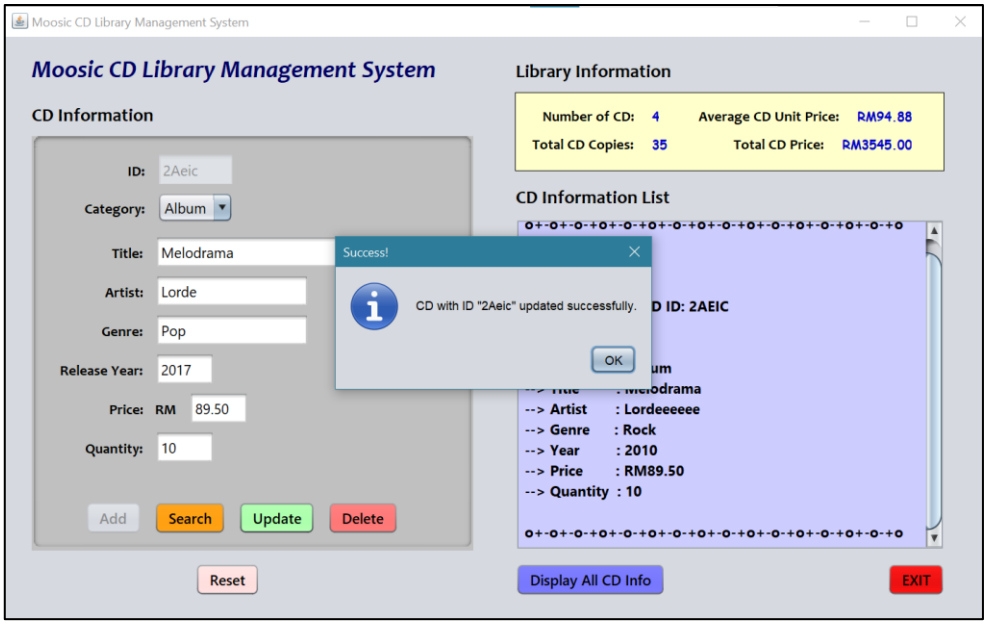

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title:
Artist:
Genre:
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
CD ID not found!
Unable to search for CD ID: 2AEICeeeeeeeS
Please insert a valid CD ID.
OK
Library Information
Number of CD: 4
Average CD Unit Price: RM94.88
Total CD Copies: 35
Total CD Price: RM3545.00
CD Information List
--> Artist : Lordeeeeee
--> Genre : Rock
--> Year : 2010
--> Price : RM89.50
--> Quantity : 10
Display All CD Info
EXIT

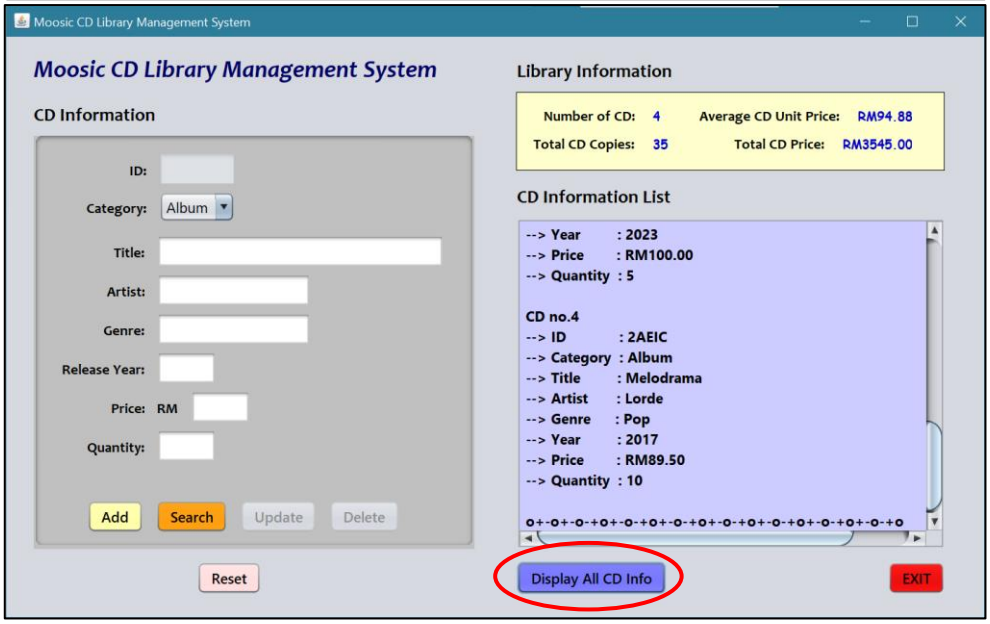

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title:
Artist:
Add
Search
Update
Delete
Reset
Search CD
Insert CD ID:
2Aeic
OK
Cancel
Library Information
Number of CD: 4
Average CD Unit Price: RM94.88
Total CD Copies: 35
Total CD Price: RM3545.00
CD Information List
--> Year : 2023
--> Price : RM100.00
--> Quantity : 5
CD no.4
--> ID : 2AEIC
--> Category : Album
--> Title : Melodrama
--> Artist : Lordeeeeee
--> Genre : Rock
--> Year : 2010
--> Price : RM89.50
--> Quantity : 10
Display All CD Info
EXIT

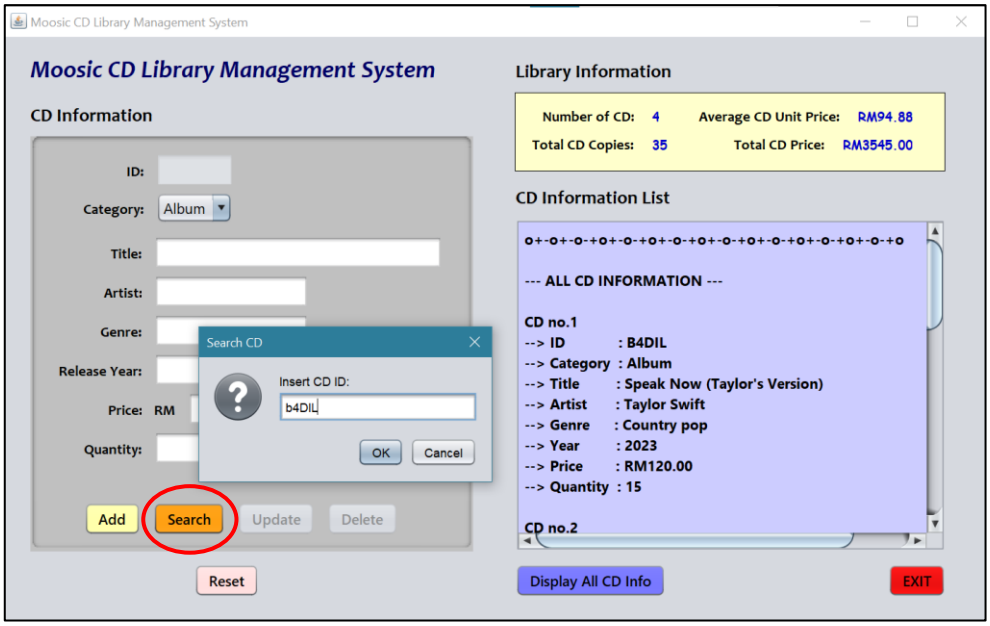

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title:
Artist:
Genre:
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
CD ID found!
This CD info can now be viewed, edited or deleted.
After editing the CD info in the respective input fields, remember to save by clicking the Update button.
Or, click the Delete button to remove CD.
OK
Library Information
Number of CD: 4
Average CD Unit Price: RM94.88
Total CD Copies: 35
Total CD Price: RM3545.00
CD Information List
--> Artist : Lordeeeeee
--> Genre : Rock
--> Year : 2010
--> Price : RM89.50
--> Quantity : 10
Display All CD Info
EXIT

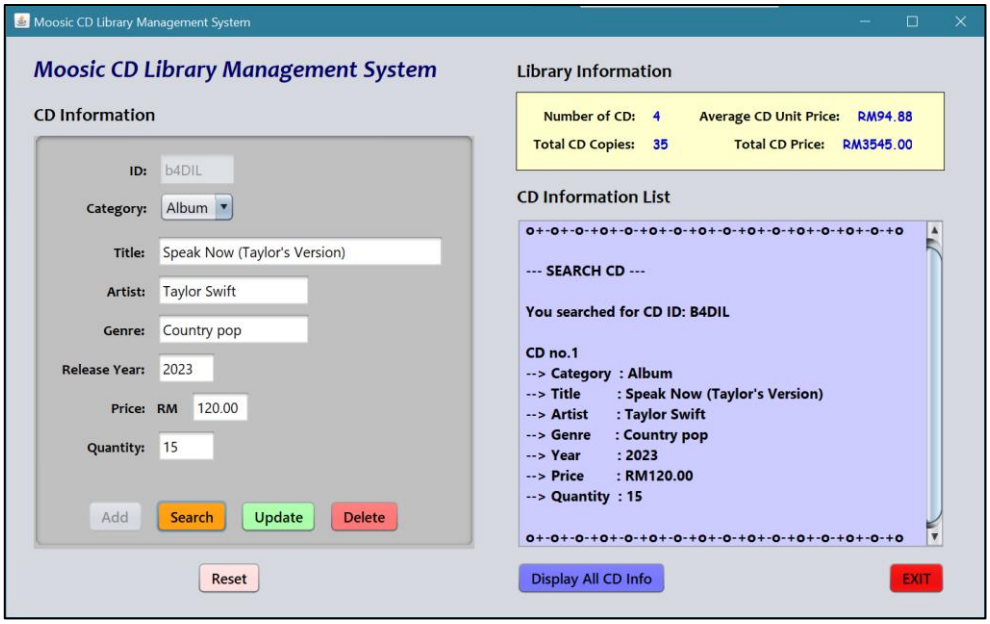

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID: 2Aeic
Category: Album
Title: Melodrama
Artist: Lorde
Genre: Pop
Release Year: 2017
Price: RM 89.50
Quantity: 10
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 4
Average CD Unit Price: RM94.88
Total CD Copies: 35
Total CD Price: RM3545.00
CD Information List
--- SEARCH CD ---
You searched for CD ID: 2AEIC
CD no.4
--> Category : Album
--> Title : Melodrama
--> Artist : Lordeeeeee
--> Genre : Rock
--> Year : 2010
--> Price : RM89.50
--> Quantity : 10
Display All CD Info
EXIT

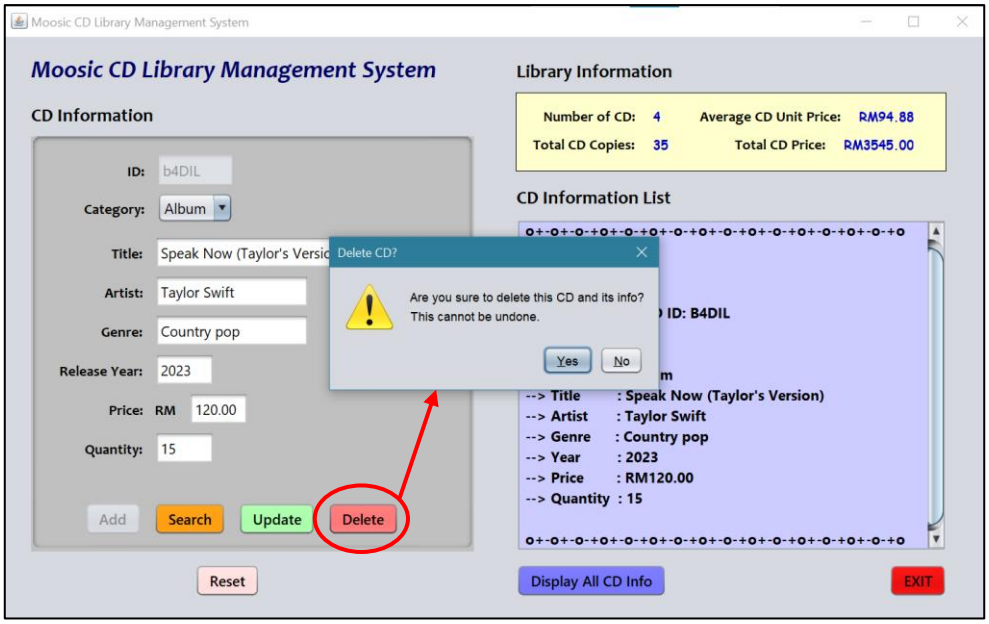

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID: 2Aeic
Category: Album
Title: Melodrama
Artist: Lorde
Genre: Pop
Release Year: 2017
Price: RM 89.50
Quantity: 10
Add
Search
Update
Delete
Reset
Update Info?
Are you sure to update this CD info?
Yes
No
Library Information
Number of CD: 4
Average CD Unit Price: RM94.88
Total CD Copies: 35
Total CD Price: RM3545.00
CD Information List
CD ID: 2AEIC
--> Artist : Lordeeeeee
--> Genre : Rock
--> Year : 2010
--> Price : RM89.50
--> Quantity : 10
Display All CD Info
EXIT

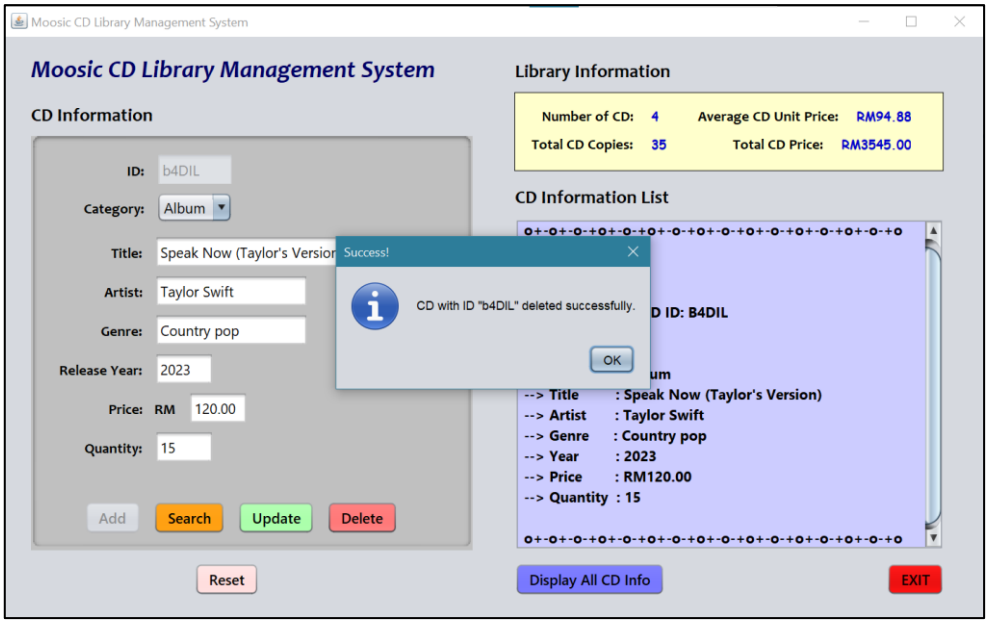

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID: 2Aeic
Category: Album
Title: Melodrama
Artist: Lorde
Genre: Pop
Release Year: 2017
Price: RM 89.50
Quantity: 10
Add
Search
Update
Delete
Reset
Success!
CD with ID "2Aeic" updated successfully.
OK
Library Information
Number of CD: 4
Average CD Unit Price: RM94.88
Total CD Copies: 35
Total CD Price: RM3545.00
CD Information List
--> Artist : Lordeeeeee
--> Genre : Rock
--> Year : 2010
--> Price : RM89.50
--> Quantity : 10
Display All CD Info
EXIT

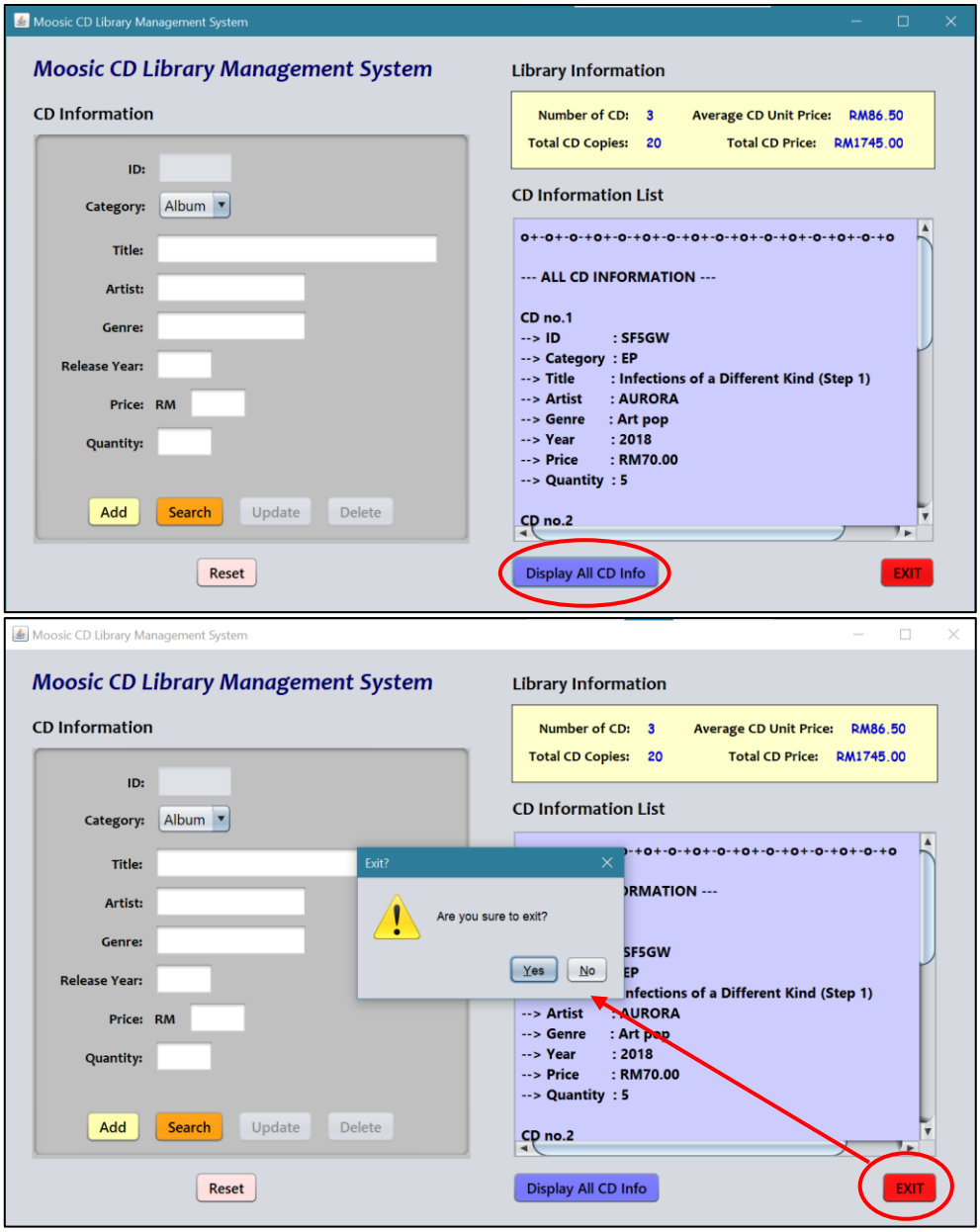

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
Category: Album
Title:
Artist:
Genre:
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 4
Average CD Unit Price: RM94.88
Total CD Copies: 35
Total CD Price: RM3545.00
CD Information List
--> Year : 2023
--> Price : RM100.00
--> Quantity : 5
CD no.4
--> ID : 2AEIC
--> Category : Album
--> Title : Melodrama
--> Artist : Lorde
--> Genre : Pop
--> Year : 2017
--> Price : RM89.50
--> Quantity : 10
Display All CD Info
EXIT


Moosic CD Library Management System
Moosic CD Library Management System
CD Information
Category: Album
Title:
Artist:
Genre:
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
Search CD
Insert CD ID:
b4DIL
OK
Cancel
Library Information
Number of CD: 4
Average CD Unit Price: RM94.88
Total CD Copies: 35
Total CD Price: RM3545.00
CD Information List
--- ALL CD INFORMATION ---
CD no.1
--> ID : B4DIL
--> Category : Album
--> Title : Speak Now (Taylor's Version)
--> Artist : Taylor Swift
--> Genre : Country pop
--> Year : 2023
--> Price : RM120.00
--> Quantity : 15
CD no.2
Display All CD Info
EXIT

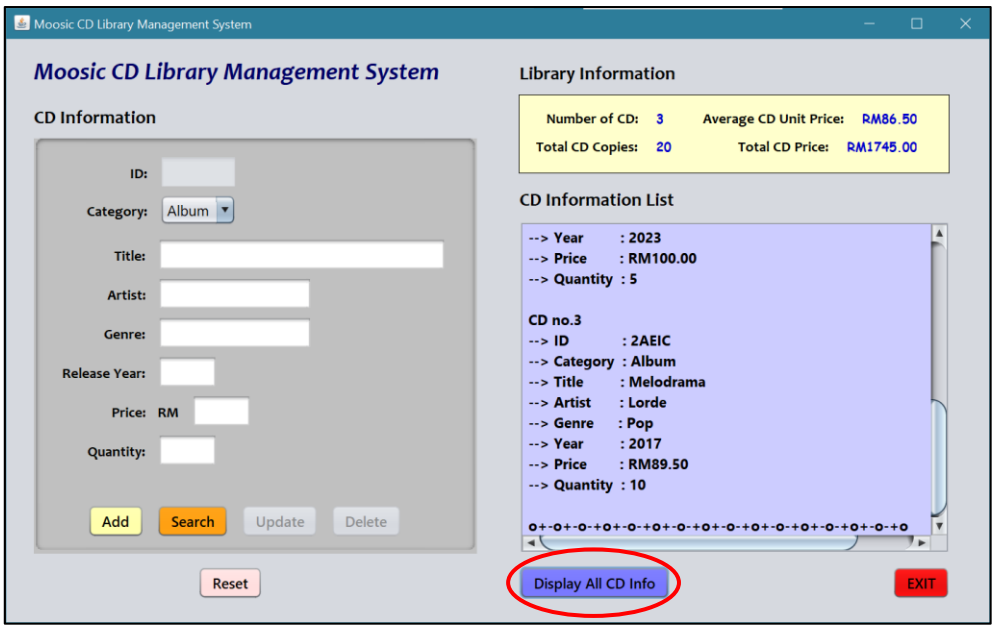

Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID: b4DIL
Category: Album
Title: Speak Now (Taylor's Version)
Artist: Taylor Swift
Genre: Country pop
Release Year: 2023
Price: RM 120.00
Quantity: 15
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 4
Average CD Unit Price: RM94.88
Total CD Copies: 35
Total CD Price: RM3545.00
CD Information List
--- SEARCH CD ---
You searched for CD ID: B4DIL
CD no.1
--> Category : Album
--> Title : Speak Now (Taylor's Version)
--> Artist : Taylor Swift
--> Genre : Country pop
--> Year : 2023
--> Price : RM120.00
--> Quantity : 15
Display All CD Info
EXIT


Moosic CD Library Management System
Moosic CD Library Management System
CD Information
Delete CD?
Are you sure to delete this CD and its info?
This cannot be undone.
Yes
No
Reset
Display All CD Info
EXIT


Moosic CD Library Management System
Moosic CD Library Management System
CD Information
Success!
CD with ID "b4DIL" deleted successfully.
OK
Reset
Display All CD Info
EXIT


Moosic CD Library Management System
Moosic CD Library Management System
CD Information
ID:
Category: Album
Title:
Artist:
Genre:
Release Year:
Price: RM
Quantity:
Add
Search
Update
Delete
Reset
Library Information
Number of CD: 3
Average CD Unit Price: RM86.50
Total CD Copies: 20
Total CD Price: RM1745.00
CD Information List
--- ALL CD INFORMATION ---
CD no.1
--> ID : SF5GW
--> Category : EP
--> Title : Infections of a Different Kind (Step 1)
--> Artist : AURORA
--> Genre : Art pop
--> Year : 2018
--> Price : RM70.00
--> Quantity : 5
CD no.2
Display All CD Info
EXIT
Exit?
Are you sure to exit?
Yes
No

If user runs the program **for the second time**, below is the screenshot of the GUI where the “Library Information” panel would still remain, and the “CD Information List” will display where the user has left off in the previous program run when user clicks on the “Display All CD Info” button: -

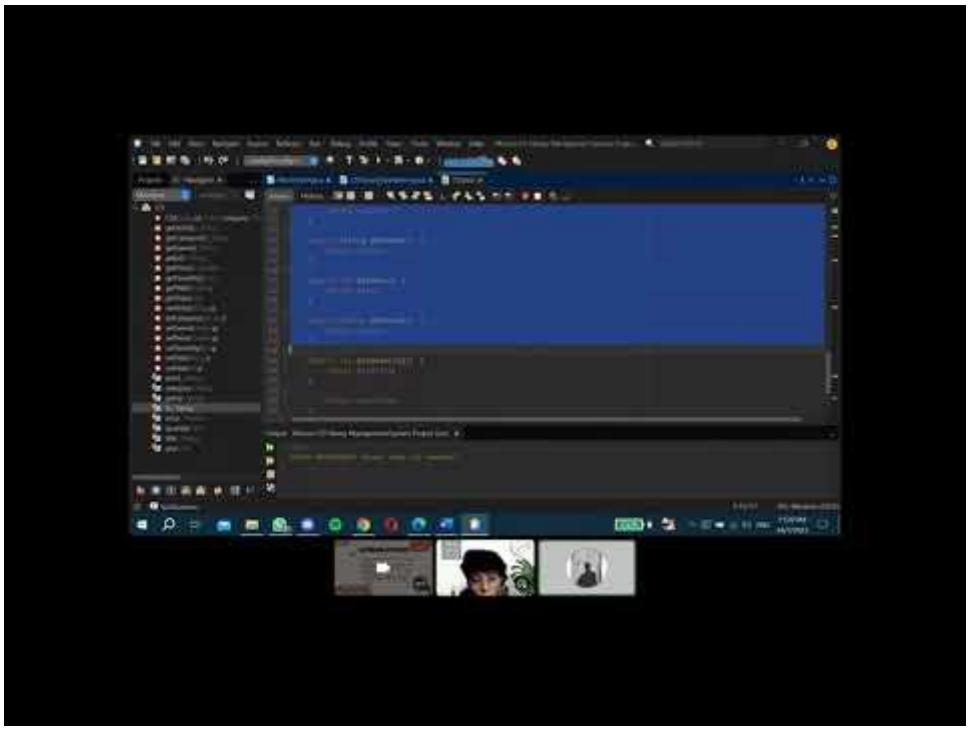


6.0 VIDEO PRESENTATION

Below is the YouTube link to our video presentation:

https://youtu.be/DAC3IbU5bAs

Embed of the video, this will direct you to the YouTube link as well:



STIA1123 PROJECT RUBRICS

Group Members : **LEE JUAN (280027)**
EDWIN LIM WEI BIN (281775)
-

0-None	**1-Very Dissatisfied**	**2 -Dissatisfied**	**3-Average**	**4-Satisfied**	**5-Very Satisfied**

Bil	**Item**	**Marks**					
PROBLEM SOLVING (12%)							
1.	**GUI Interface Layout/Design (P3)** *The system provides simple, well-organized and readable interface for input & output*	0	1	2	3	4	5
2.	**Functionality: (P4)**						
	i) Add	0	1	2	3	4	5
	ii) Delete	0	1	2	3	4	5
	iii) Search	0	1	2	3	4	5
.	**iv) Display**	0	1	2	3	4	5
	v) Update	0	1	2	3	4	5
3.	**User-defined Class (P4)**	0	1	2	3	4	5
4.	**Usage of list (P5)**	0	1	2	3	4	5
5.	**Usage of Text File (P5)**	0	1	2	3	4	5
6.	**Exception (P5)**	0	1	2	3	4	5
7.	**OOP concept (abstraction, encapsulation) (P3)**	0	1	2	3	4	5
8.	**Calculation (P3)**	0	1	2	3	4	5
	TOTAL (60)						

Verbal Communication (5%)							
	i) Clarity	0	1 Unable to explain	2 Able to explain **limited** program design	3 Able to explain **some** program design	4 Able to explain **most** program design	5 Able to explain ALL program design
	ii) Question/Answer	0	1 Unable to answer	2 Able to answer limited questions	3 Able to answer some questions	4 Able to answer most questions	5 Able to answer ALL questions
	Total (10)						
Written Communication (3%)							
	i) Well Written/Clarity – the way that they explain user manual, background	0	1 Very sloppy	2 Poorly written	3 Fairly written	4 Appropriately written	5 Perfectly written
	ii) Content Coverage – background, UML, user manual, source code, sample run	0	1 Have at least 1 required contents.	2 Have at least 2 required contents.	3 Have at least 3 required contents.	4 Have at least 4 required contents.	5 Have ALL required content
	ii) Organization and Structure -	0	1 No organization	2 Disorganized, difficult to locate information	3 Organized but presentation is confusing	4 Mostly organized and easy to find info	5 Appropriately organized and easy to find info
	Total (15)						