Report: Java Programming

CD Library Management System (Java Language)

Luckhun Leckraj - M00677980

Middlesex University (Mauritius)

ITX 1300 Module – Introduction to Programming

Module Coordinator: Dr. Aditya Santokee

4th April 2019

Table of Contents

Abstract

This report is about my Java Project where I had to design and create a CD Library Management System using Java platform, and including an appealing experience for any user, by making use of GUI, and exciting features. This report lists the details about the project, its design and implementation, features, problems that arised, and solutions to overcome them.

1. Introduction

1.1 Overview

Java is a class-based, object-oriented programming language, designed to develop programs. Code developed in Java can be compiled and run on multiple platforms. It avoids re-compilation to run on other platforms. Java programs are saved as class file that can be run on any Java Virtual Machine (JVM). Most applications developed with Java contain classes, which are then used to define objects, including methods. (Christensson, 2012). Initially Java was developed by James Gosling at Sun Microsystems, which has now merged with Oracle Corporation, and later in 1995, this merger released the Java Language as a component of Sun Microsystems' Java platform. (Mercer, 2017).

1.2 Object – Oriented Programming

As mentioned above, Java is an object- oriented programming language, that encompasses a hierarchy of classes, and objects. (Austerlitz, 2003).

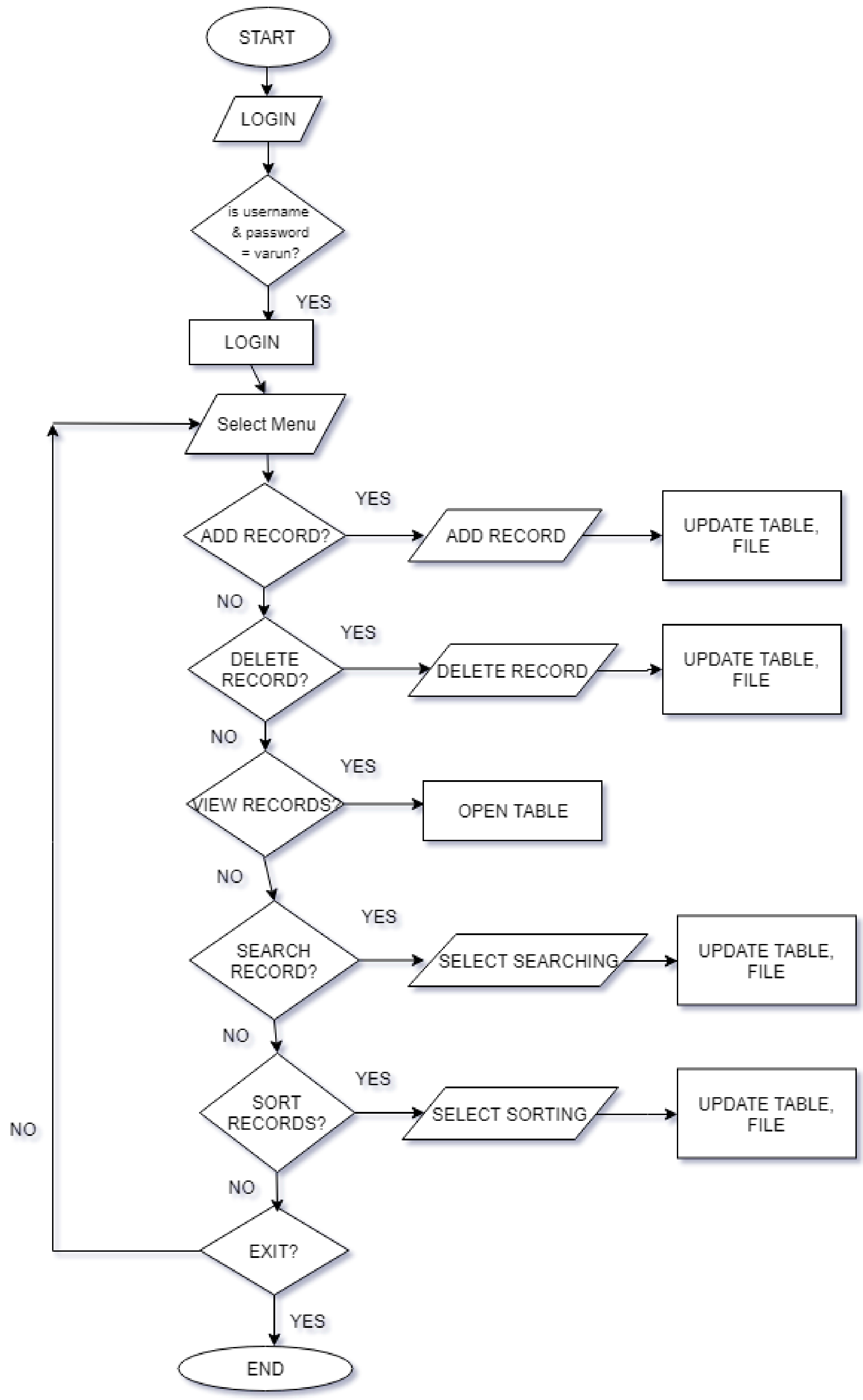
Classes: When you code in Java, the program data is stored in a class. They state the data and techniques to work on the program. Fields, methods and constructors are defined in classes. (Oracle, 2017).

Objects: An object is an instance of a class. Once instantiated, the `dot` operator can be used to access instance variables and methods. (Oracle, 2017).

1.3 The Main Project- CD Management Library System

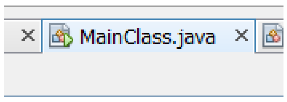
The aim of this report is to list the details of a Java project, namely, a CD Management Library System. The project has ten classes, mainly, AddMovieRecord, ConfirmBox, DeleteMovieRecord, FileManagement, MainClass, Movies, SearchMovieRecord, SortMovieRecord, Table, and login. This application lets a user login into the system, add movies, delete movies, view the list of movies, search for records, and sort records, based on some criterias found in the program. A text file is also used in the application, whereby all data added will be added to the text file, and when an entry is removed, the file is updated with new contents.

2.0 Flowchart



3.0 Implementation

The CD Library Application consists of ten classes, which were developed according to the flowchart. The flowchart helped in designing the application in such a way that it is convenient for the user. I started off by creating a main class, which is the central part of my project. Every other class are connected to this single Main Class.



The following classes were developed in this order:

1. MainClass.java – The Main Class linking all classes together
2. ConfirmBox.java – Exit application request
3. login.java – Login window
4. Movies.java - Constructors
5. FileManagement.java – To read and write to the file
6. AddMovieRecord.java – To add a record
7. Table.java – The table used in the application
8. DeleteMovieRecord.java – To remove a record
9. SearchMovieRecord.java – To search for a movie
10. SortMovieRecord.java – To sort records

3.1 Start of the Project

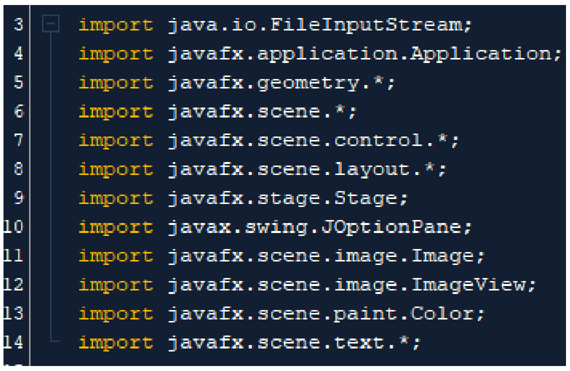
To start the project, I researched on how CD re-sellers manage their system. I have a friend who owns a studio, so he helped me in identifying the variables required for my project. I also checked on the Internet to find more ideas about the implementation, since every bit of this application needs to run smoothly.

3.2 User-Friendly Interface

I decided to design a Graphical User Interface (GUI) for user convenience. A GUI is an interface, which enables user interaction with devices, such as computers or any other appliances. GUI comprises of menus, icons, text, images, and drawings, which enhance user experience when using an application. (Techopedia.com, 2019). I made use of JavaFX to build a GUI. JavaFX is more advanced and render high-quality GUI, compared to previous versions such as AWT and Java Swing. Building the GUI was not an easy task, as I had to watch videos, read books, check on the Internet, and find samples to develop the system. I learn more about JavaFX from a YouTuber, who demonstrated the building and implementation of JavaFX in an application. (thenewboston, 2015).

"JavaFX is a Java library used to build Rich Internet Applications. The applications written using this library can run consistently across multiple platforms. The applications developed using JavaFX can run on various devices such as Desktop Computers, Mobile Phones, TVs, Tablets, etc..", (tutorialspoint, 2019).

The following libraries have been imported to build the JavaFX GUI:

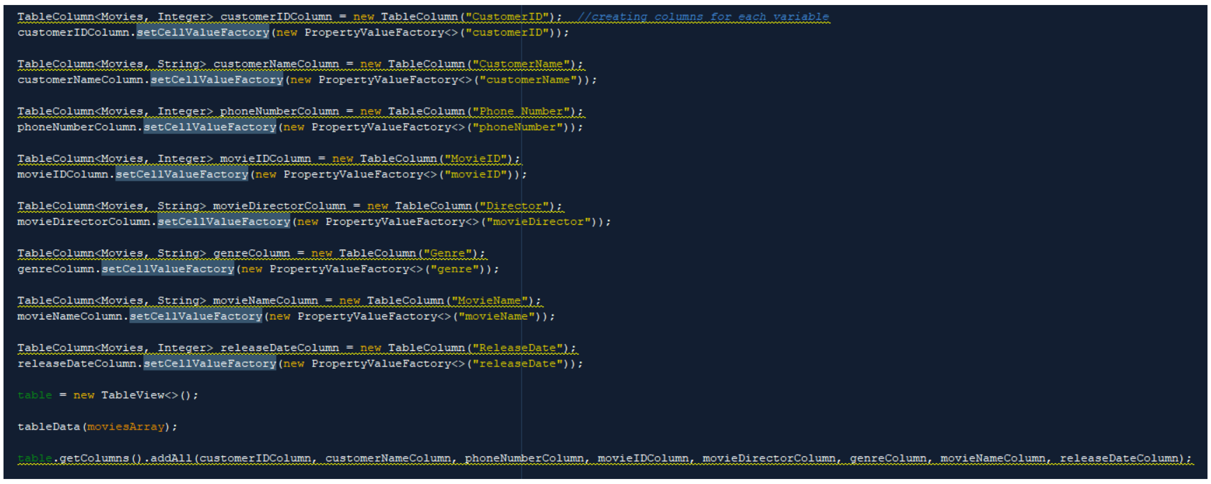


```
import java.io.FileInputStream;
import javafx.application.Application;
import javafx.geometry.*;
import javafx.scene.*;
import javafx.scene.control.*;
import javafx.scene.layout.*;
import javafx.stage.Stage;
import javax.swing.JOptionPane;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.scene.paint.Color;
import javafx.scene.text.*;
```

3.3 Implementation Problems & Solutions

3.3.1 Table

When conducting research about how to create a table, I was quite lost, as most of them were using ArrayList. The issue was that I am not allowed to use ArrayList in my project, as it does not meet the requirements of the project. I had to come up to a solution as soon as possible, and after thorough research, I decided to use an array of objects and return them into a table.

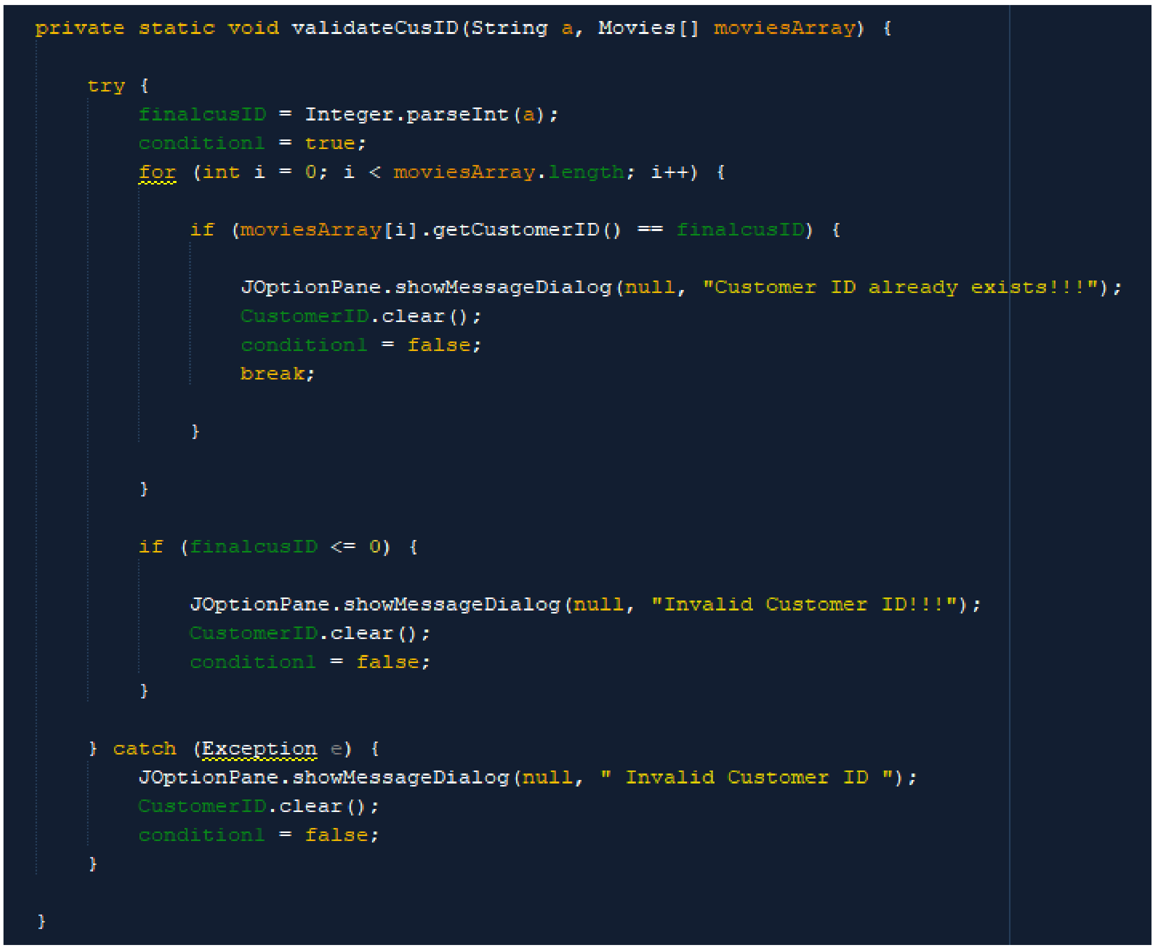


```java
TableColumn<Movies, Integer> customerIDColumn = new TableColumn("CustomerID");  //creating columns for each variable
customerIDColumn.setCellValueFactory(new PropertyValueFactory<>("customerID"));

TableColumn<Movies, String> customerNameColumn = new TableColumn("CustomerName");
customerNameColumn.setCellValueFactory(new PropertyValueFactory<>("customerName"));

TableColumn<Movies, Integer> phoneNumberColumn = new TableColumn("Phone Number");
phoneNumberColumn.setCellValueFactory(new PropertyValueFactory<>("phoneNumber"));

TableColumn<Movies, Integer> movieIDColumn = new TableColumn("MovieID");
movieIDColumn.setCellValueFactory(new PropertyValueFactory<>("movieID"));

TableColumn<Movies, String> movieDirectorColumn = new TableColumn("Director");
movieDirectorColumn.setCellValueFactory(new PropertyValueFactory<>("movieDirector"));

TableColumn<Movies, String> genreColumn = new TableColumn("Genre");
genreColumn.setCellValueFactory(new PropertyValueFactory<>("genre"));

TableColumn<Movies, String> movieNameColumn = new TableColumn("MovieName");
movieNameColumn.setCellValueFactory(new PropertyValueFactory<>("movieName"));

TableColumn<Movies, Integer> releaseDateColumn = new TableColumn("ReleaseDate");
releaseDateColumn.setCellValueFactory(new PropertyValueFactory<>("releaseDate"));

table = new TableView<>();

tableData(moviesArray);

table.getColumns().addAll(customerIDColumn, customerNameColumn, phoneNumberColumn, movieIDColumn, movieDirectorColumn, genreColumn, movieNameColumn, releaseDateColumn);
```

3.3.2 GUI

The problem with the GUI was that while the program was running, and the user made a choice, and at that exact moment when data were to be returned, the user decided to abort the operation. The program would crash by aborting this operation. To solve this issue, I had to implement multiple validations possible to make sure the program runs smoothly, without crashing.

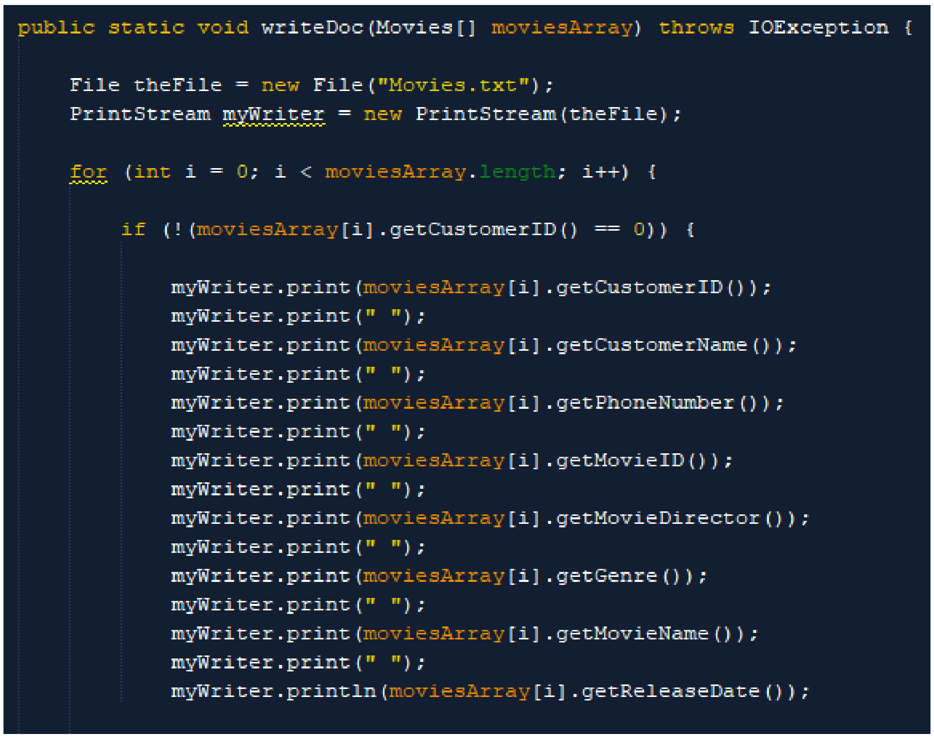
An example is shown below. In this picture, the program makes sure that there is no duplicate CustomerID being entered. Validations had to be implemented to solve these issues.



```
private static void validateCusID(String a, Movies[] moviesArray) {

    try {
        finalcusID = Integer.parseInt(a);
        condition1 = true;
        for (int i = 0; i < moviesArray.length; i++) {

            if (moviesArray[i].getCustomerID() == finalcusID) {

                JOptionPane.showMessageDialog(null, "Customer ID already exists!!!");
                CustomerID.clear();
                condition1 = false;
                break;

            }

        }

        if (finalcusID <= 0) {

            JOptionPane.showMessageDialog(null, "Invalid Customer ID!!!");
            CustomerID.clear();
            condition1 = false;
        }

    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, " Invalid Customer ID ");
        CustomerID.clear();
        condition1 = false;
    }

}
```

3.3.3 Files - Writing to Files

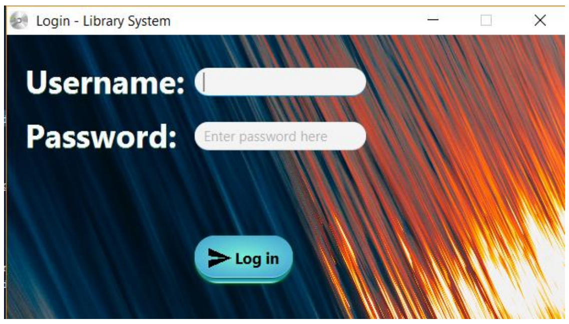
The problem associated with the file was writing to the file. I have not used the split method to read data from files, but instead, I used <space>. I left a <space> between each data in the file. To solve the issue, I applied myWriter.print method to write to the file. The program will write the data on the same line itself and shall remain on the current line itself until another data is written.



```java
public static void writeDoc(Movies[] moviesArray) throws IOException {

    File theFile = new File("Movies.txt");
    PrintStream myWriter = new PrintStream(theFile);

    for (int i = 0; i < moviesArray.length; i++) {

        if (!(moviesArray[i].getCustomerID() == 0)) {

            myWriter.print(moviesArray[i].getCustomerID());
            myWriter.print(" ");
            myWriter.print(moviesArray[i].getCustomerName());
            myWriter.print(" ");
            myWriter.print(moviesArray[i].getPhoneNumber());
            myWriter.print(" ");
            myWriter.print(moviesArray[i].getMovieID());
            myWriter.print(" ");
            myWriter.print(moviesArray[i].getMovieDirector());
            myWriter.print(" ");
            myWriter.print(moviesArray[i].getGenre());
            myWriter.print(" ");
            myWriter.print(moviesArray[i].getMovieName());
            myWriter.print(" ");
            myWriter.println(moviesArray[i].getReleaseDate());
```

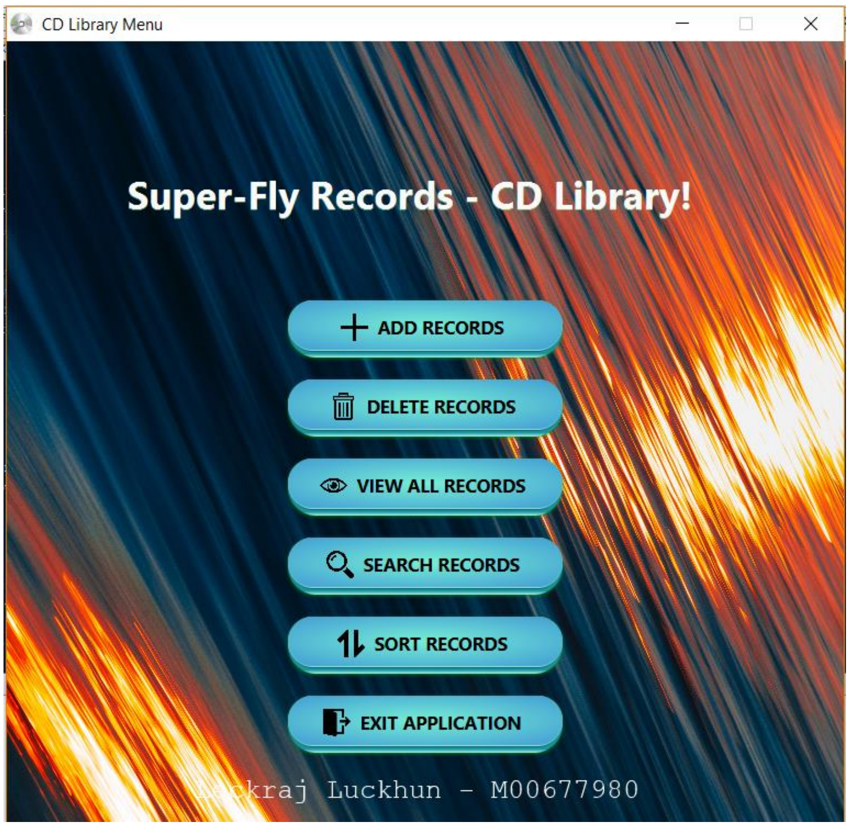
4.0 Application Interface

4.1 Login



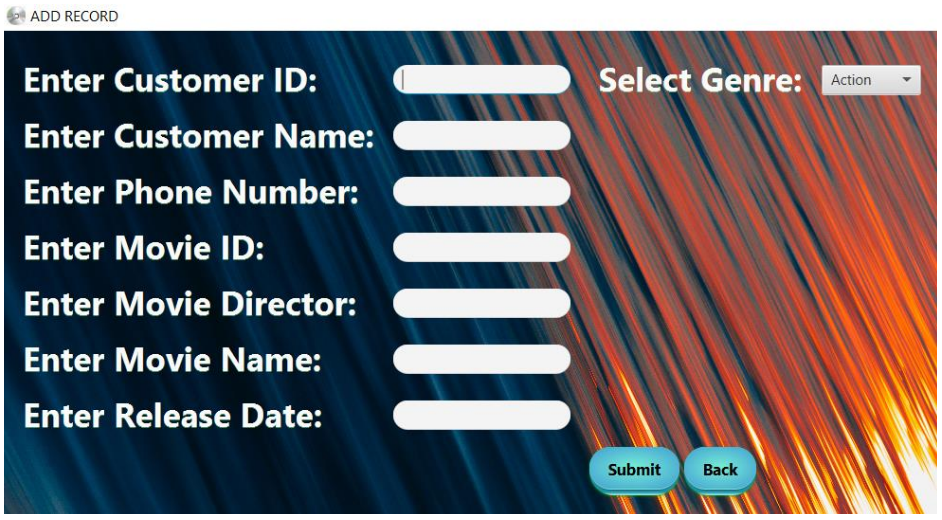
The Login interface is where the user inputs the appropriate username and password to get access to the main menu. In this scenario, the username is “varun”, and the password is “varun”. If the username or password is wrongly written, an error message “Invalid Credentials” will be displayed.

4.2 Menu



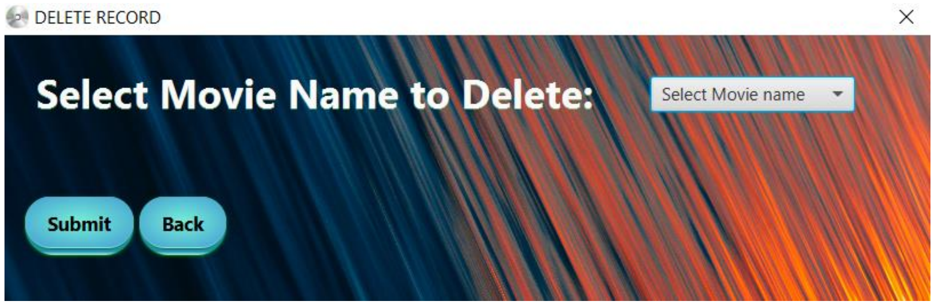
The Menu is where the user selects his/her desired operation, and proceed to the respective window to carry out operations.

4.3 Add Records



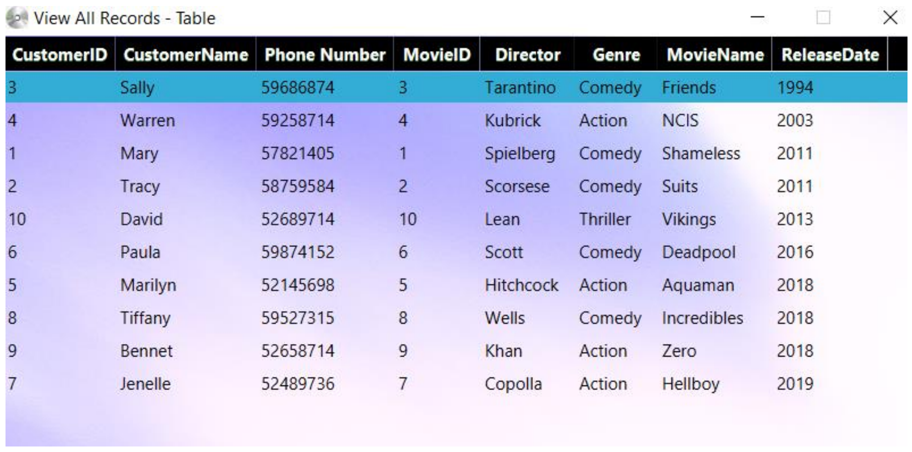
In the Add Records window, the user has to input the details of the customer and the movie. Once filled, the user clicks the submit button, and this triggers a validation check on all fields, before adding the record to the file. Validations include checking if the same CustomerID, as well as MovieID, have been entered twice. It will also check if the Customer Name contains numbers or invalid characters.

4.4 Delete Records



In the Delete Records window, the user has to select what record he/she wants to remove by selecting the Movie Name from the drop-down button. Once the user clicks on submit, the entry is removed, and the file, as well as the table, is updated.

4.5 View All Records

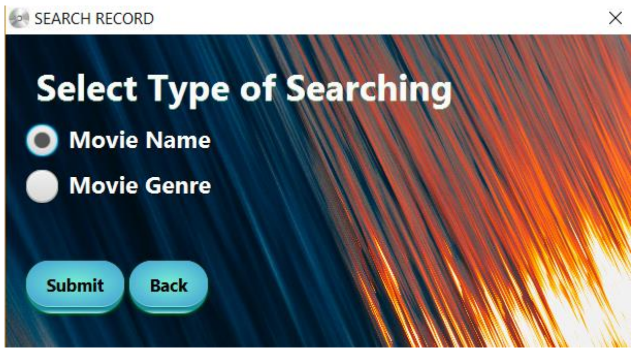


View All Records - Table

CustomerID	CustomerName	Phone Number	MovieID	Director	Genre	MovieName	ReleaseDate
3	Sally	59686874	3	Tarantino	Comedy	Friends	1994
4	Warren	59258714	4	Kubrick	Action	NCIS	2003
1	Mary	57821405	1	Spielberg	Comedy	Shameless	2011
2	Tracy	58759584	2	Scorsese	Comedy	Suits	2011
10	David	52689714	10	Lean	Thriller	Vikings	2013
6	Paula	59874152	6	Scott	Comedy	Deadpool	2016
5	Marilyn	52145698	5	Hitchcock	Action	Aquaman	2018
8	Tiffany	59527315	8	Wells	Comedy	Incredibles	2018
9	Bennet	52658714	9	Khan	Action	Zero	2018
7	Jenelle	52489736	7	Copolla	Action	Hellboy	2019

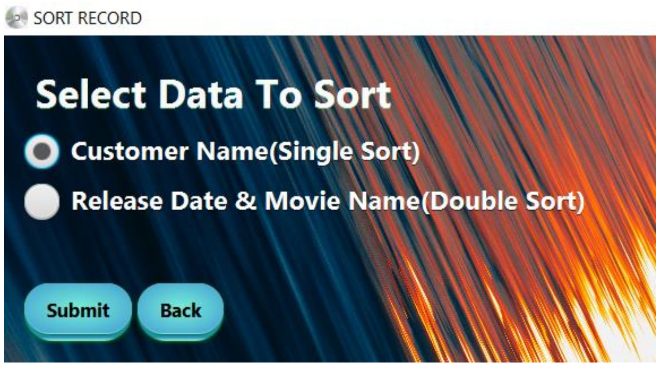
In the View All Records window, a table displays all records that are currently on the file.

4.6 Search Records



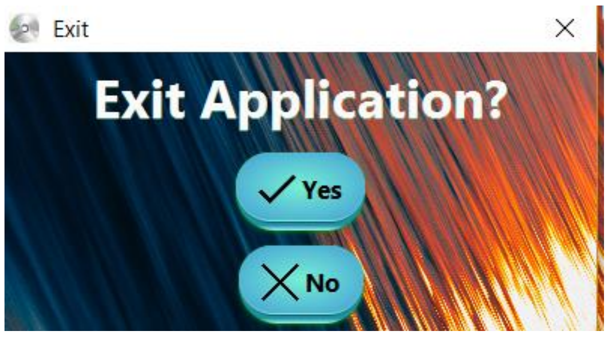
In the Search Record window, the user has to choose the type of searching, which is by Movie Name or by Movie Genre. If the user selects by Movie Name, a drop-down is provided to select the movie. If the user selects Movie Genre, a drop-down listing all available genre is provided. The user has to select the Movie Genre, and all records having that specific Movie Genre is displayed in a table.

4.7 Sort Records


SORT RECORD
Select Data To Sort
Customer Name(Single Sort)
Release Date & Movie Name(Double Sort)
Submit
Back

In the Sort Records window, the user has to select from two types of sorting, which includes, Single Sort and Double Sort. In this application, sorting by Customer Name is Single Sort, whereas sorting by Release Date & Movie Name is Double Sorting. Based on user selection, a table will appear with the appropriate result.

4.8 Exit Confirmation



The Exit Application pops out when the user wants to exit the Main Menu. It asks the user if he/she really want to exit the application.

5. References

1. Christensson, P. (2012). Java Definition. [online] TechTerms. Available at: https://techterms.com/definition/java [Accessed 4 Apr. 2019].

2. Mercer, J. (2017). A Short History of Java. [online] DZone. Available at: https://dzone.com/articles/a-short-history-of-java [Accessed 6 Apr. 2019].

3. Austerlitz, H. (2003). Java Programming Language- An Overview. 2nd ed. [ebook] ScienceDirect, pp.1-2. Available at: https://www.sciencedirect.com/topics/computer-science/java-programming-language [Accessed 6 Apr. 2019].

4. Oracle. (2017). Lesson: Classes and Objects (The Java™ Tutorials > Learning the Java Language). [online] Available at: https://docs.oracle.com/javase/tutorial/java/javaOO/index.html [Accessed 6 Apr. 2019].

5. Techopedia.com. (2019). What is a Graphical User Interface (GUI)? - Definition from Techopedia. [online] Available at: https://www.techopedia.com/definition/5435/graphical-user-interface-gui [Accessed 7 Apr. 2019].

6. thenewboston (2015). JavaFX Java GUI Tutorial - 1 - Creating a Basic Window. [video] Available at: https://youtu.be/FLkOX4Eez6o?list=PL6gx4Cwl9DGBzfXLWLSYVy8EbTdpGbUIG [Accessed 7 Apr. 2019].

7. tutorialspoint. (2019). JavaFX Tutorial. [online] Available at: https://www.tutorialspoint.com/javafx/ [Accessed 7 Apr. 2019].

8. Oracle. (2016). Client Technologies: Java Platform, Standard Edition (Java SE) 8 Release 8. [online] Available at: https://docs.oracle.com/javase/8/javase-clienttechnologies.htm [Accessed 8 Apr. 2019].