Module Code & Module Title

CS5004NI Emerging Programming Platforms and Technologies

Assessment Weightage & Type

30% Group Coursework

Year and Semester

2019-20 Autumn

Group Name:			
S.N.	**Student Name**	**College ID**	**University ID**
1.	Meru Sangroula	NP01CP4A180038	18029653
2.	Sodip Bikram Thapa	NP01CP4A180059	18029759
3.	Aaryan Shrestha	NP01CP4A180030	18029599

Table of Contents

Table of Figures

Table of Tables

1. Proposal

1.1. Introduction

This coursework requires us to develop an information system for a company or an organization of our choice and we have decided to develop such a system specialized for bookstores. The system will allow its users to add new books, search for specific books, and group books by genres, authors, and publications. This system will allow bookstore owners to manage and access their data and information in a quick and convenient manner. In order to develop this system, we will be using the concepts and tools that we have learned during this semester.

1.2. List of Data

- **Book Id**: Unique Id for each book (*Integer*)
- **Book Name**: The Name of the book (*String*)
- **Author**: Name of the book's author (*String*)
- **Book Type**: Radio button to choose between hardcover or paperback (*Stored as String*)
- **Genre**: Genres of available books selectable from Combo Box (*Stored as String*)
- **Publication**: Name of the publications of the available books (*String*)
- **Price**: The selling price of the book (*Double*)

1.3. List of Features

- Add new books based on the input provided by the user.
- Display stored data in a table that is updated when a new book is added.
- Books can be searched based on price or genre.
- Delete a row from the table.
- Clear the entire table.
- A user manual containing instructions for using the program.
- Add more data to the table from a text file.

1.4. Technology Used

1.4.1. Apache NetBeans IDE

For developing the whole system, the Apache NetBeans IDE (Version 11.2) will be used. The IDE provides one with easy to use drag and drop features for developing GUIs for desktop applications. This helps them to waste less time in designing the user interface. The GUI will be developed using Java Swing which is GUI widget tool kit for developing GUIs with Java.

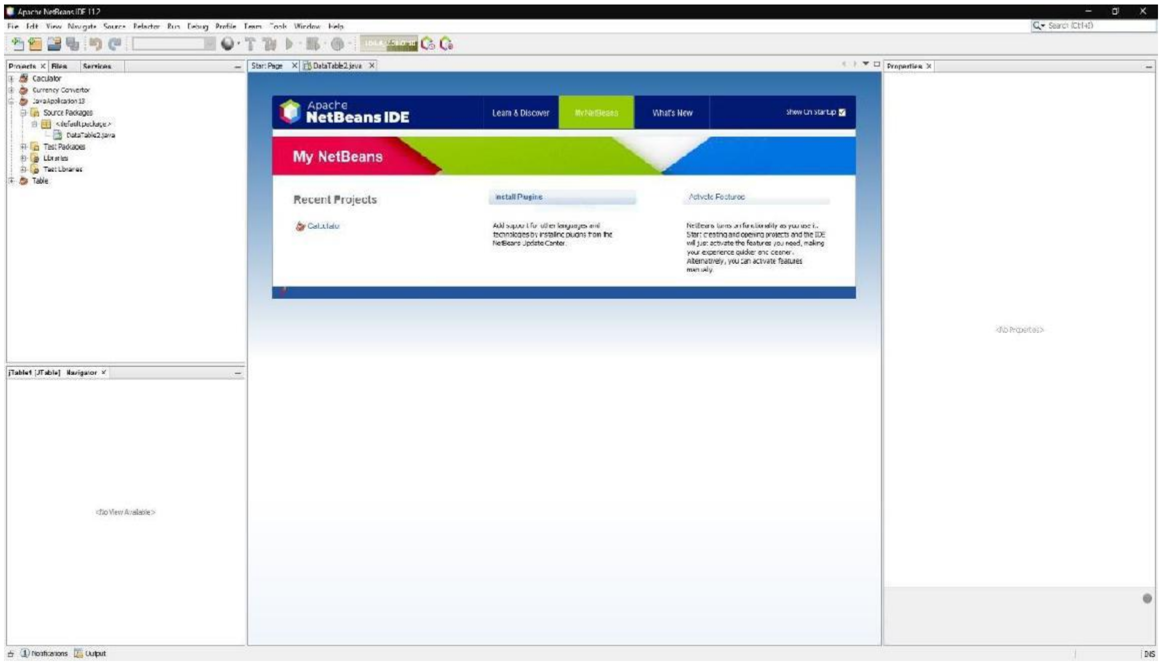


Figure 1: Apache NetBeans

1.4.2. Java Programming Language

The Java programming language will be used to develop the application. It is one of the most widely used general purpose programming language in the world so, learning the language is one important reason for choosing it for this project. Also, Java is an object-oriented programming language and by using it for this project we will also be able to apply our knowledge of OOP (Object-Oriented Programming) concepts in practice.

1.4.3. Balsamiq Wireframes

For developing the wireframes for the system's user interface, Balsamiq will be used. It is an easy to desktop application for developing UI wireframes. The main reason for choosing Balsamiq is that, we had previously learned to use it for one of our coursework during our first year so all of us are already familiar with it and can use it properly.

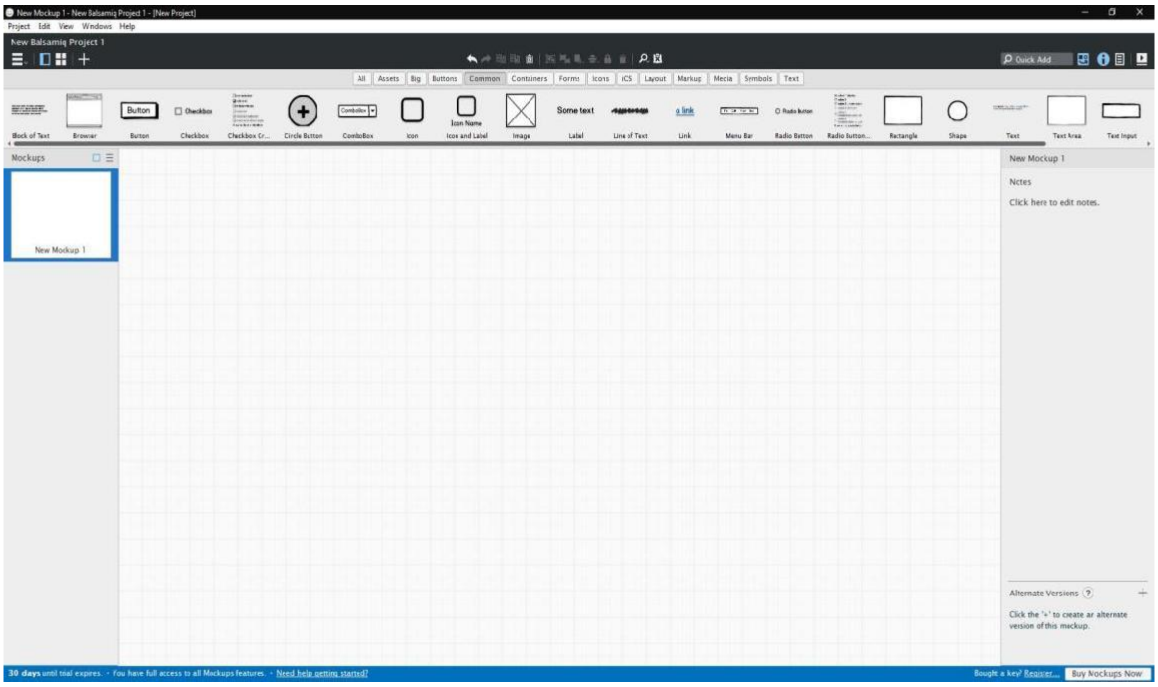

New Mockup 1 - New Balsamiq Project 1 - [New Project]
Project Edit View Windows Help
New Balsamiq Project 1
Quick Add
All Assets Big Buttons Common Containers Forms Icons iOS Layout Markup Media Symbols Text
Block of Text Browser Button Checkbox Checkbox Gr... Circle Button ComboBox Icon Icon and Label Image Label Line of Text Link Menu Bar Radio Button Radio button... Rectangle Shape Text Text Area Text Input
Mockups
New Mockup 1
Notes
Click here to edit notes.
Alternate Versions
Click the '+' to create an alternate version of this mockup.
30 days until trial expires. · You have full access to all Mockups features. · Need help getting started?
Bought a key? Register... Buy Mockups Now

Figure 2: Balsamiq Wireframes

1.5. Wireframe

The wireframe for the program is given below:

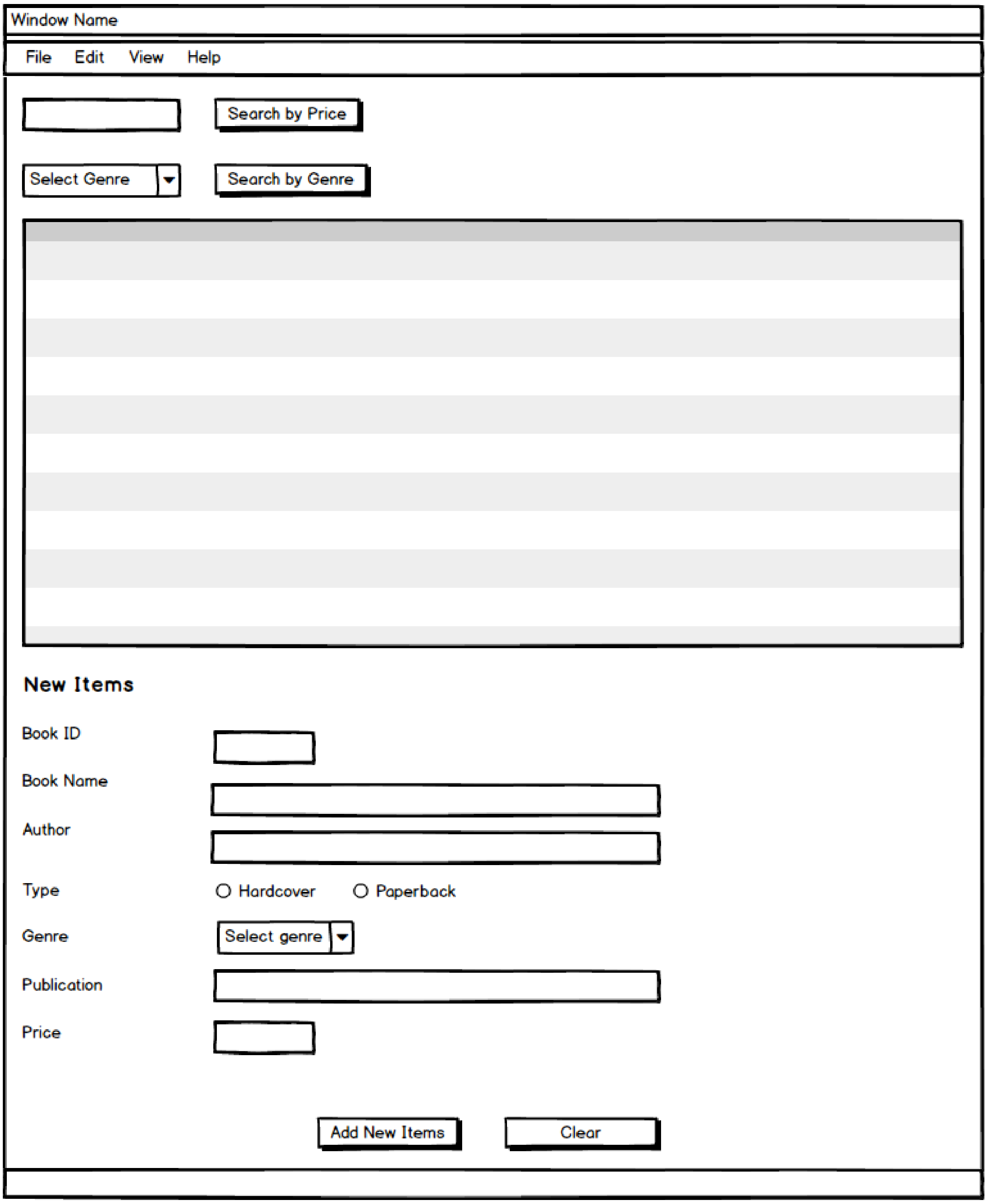


Figure 1: Program Wireframe

2. Report

2.1. Introduction

The coursework given in this module requires us to create a Java based Menu Information System. For this, a project named bookmark was created containing a class called App. The project created allowed us to store information about a specific book. The information of the books that can be stored in the program are book id, book name, author, book type, genre, publication and price. As required by the coursework text field, radio button and combo box were used. The table contains information of twelve books by default. We can add the information stored in a text file with the help of open option in the file menu of the menu bar. We can also delete information of specific books with the help of delete selected row option in the edit menu. There is also an option to delete all the information stored in the table called clear table in the edit menu. A user manual can be opened in pdf format from the manual option in the help menu. The project also has a function to search books by its price and by its genre. The option to search books by its price was created using binary search algorithm. Similarly, the option to search books by its genre was created using linear search algorithm. This sums up the features that are in the project created.

This project was created using Netbeans. Netbeans is an open source integrated development environment (IDE) for developing with Java, PHP, C++, and other programming language. Netbeans is also a platform of modular components used for developing Java desktop applications. The Graphical User Interface (GUI) for the project was created using the GUI tool in Netbeans and the programming was done using Netbeans JavaScript Editor.

2.2. Individual Tasks

The tasks carried out by each of the group members for this groupwork:

SN	Name	Task
1.	Meru Sangroula	Report, Validation, UI design
2.	Sodip Bikram Thapa	Binary Search, Linear Search, Selection Sort
3.	Aaryan Shrestha	Update Table from file, Testing

Table 1: Individual Tasks

2.3. Binary Search Algorithm

Binary Search is an algorithm for efficiently searching for an item within a sorted list of items. It is one of the most popular search algorithms. The search is performed in steps, with each step reducing the search space by half (with linear or sequential search the search space reduces by only one item). At each step, the midpoint is selected and compared against the search value. If the midpoint is lesser than the search value, only the upper half is retained for the next step (Jeet Singh, 2017).

For example, let us consider the following sequence of integers sorted in ascending order and the number which is to be searched is 23. At first, the median value is chosen which is 16 in this example. In the above example, 16 is smaller than 23 so all the values in the left side are neglected. Now 23 is searched only in right side following the same process.

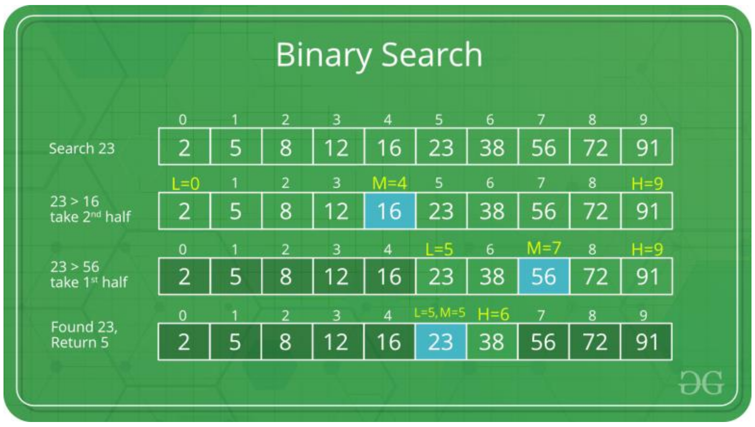
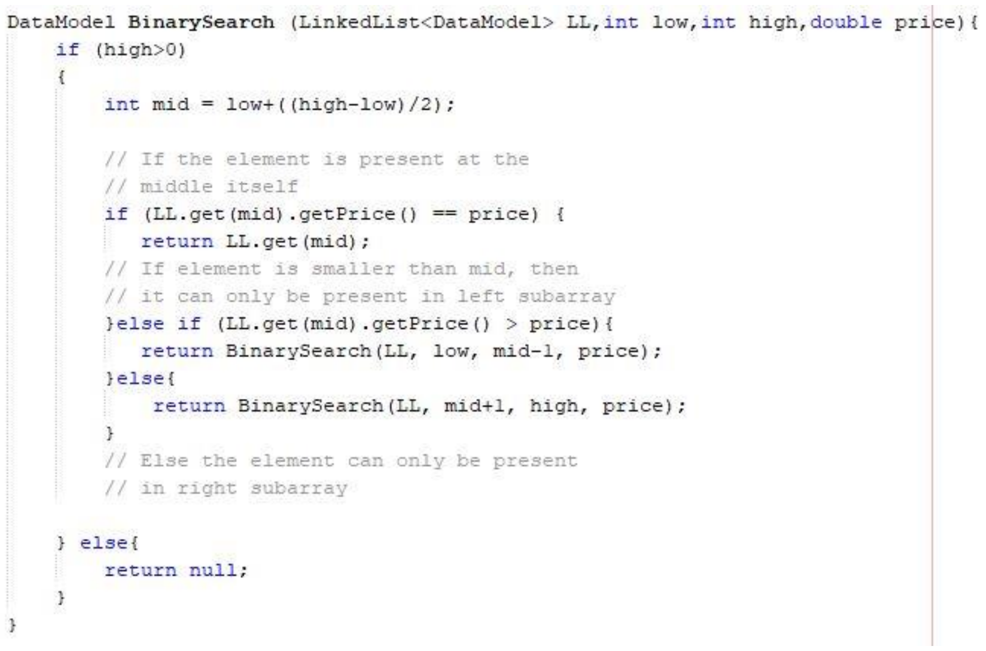


Figure 2: Binary Search Example (geeksforgeeks, 2020)

The Binary Search algorithm has been implemented in the Algorithms class of this project. The name of the method where it has been implemented is BinarySearch. The method takes a Linked List containing instances of the data model class, two integer values for the lowest and highest indexes, and a double value for searching through the Linked List as arguments. The two integer values are used for accessing the elements of the linked list as well as calculating the middle index. The formula used for calculating the middle index is:

mid=low+((high-low)/2)

When the method is called it returns the matched element from the linked list. If no match is found in the list, then a null value is returned.



```
DataModel BinarySearch (LinkedList<DataModel> LL,int low,int high,double price){
    if (high>0)
    {
        int mid = low+((high-low)/2);

        // If the element is present at the
        // middle itself
        if (LL.get(mid).getPrice() == price) {
           return LL.get(mid);
        // If element is smaller than mid, then
        // it can only be present in left subarray
        }else if (LL.get(mid).getPrice() > price){
           return BinarySearch(LL, low, mid-1, price);
        }else{
            return BinarySearch(LL, mid+1, high, price);
        }
        // Else the element can only be present
        // in right subarray

    } else{
        return null;
    }
}
```

Figure 3: Binary Search Implementation

2.4. Linear Search Algorithm

Linear search is an algorithm which searches target value within a list. It checks each element of the list until the match is found or reaches end. It is the most basic type of searching algorithm.

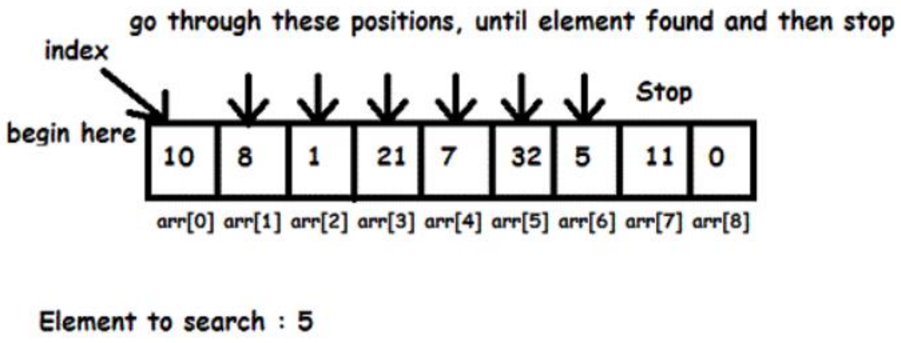
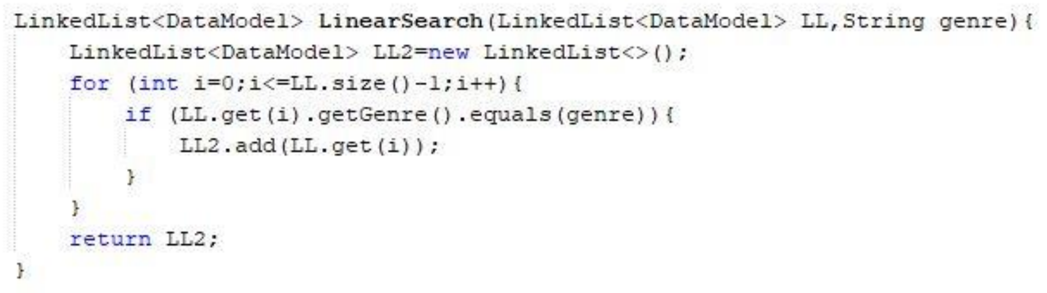


Figure 4: Linear Search Example (Sehgal, 2017)

This is an example of linear search. It basically starts from the left side array and compare each element one by one. If the element matches, then the index is returned.

The linear search algorithm was used in search by genre function in the program. The linear search algorithm code is in the LinearSearch method of the controller class. First, a variable named genre takes the input from combo box cmbSelectGenre. The genre of each book is extracted in the method using a for loop. If the genre matches the genre in the combo box, then the books information is passed to another linked list named LL2. Then the names of the books are shown to the user in the btnSearchGenreActionPermformed method of the app class.



```
LinkedList<DataModel> LinearSearch(LinkedList<DataModel> LL,String genre){
    LinkedList<DataModel> LL2=new LinkedList<>();
    for (int i=0;i<=LL.size()-1;i++){
        if (LL.get(i).getGenre().equals(genre)){
            LL2.add(LL.get(i));
        }
    }
    return LL2;
}
```

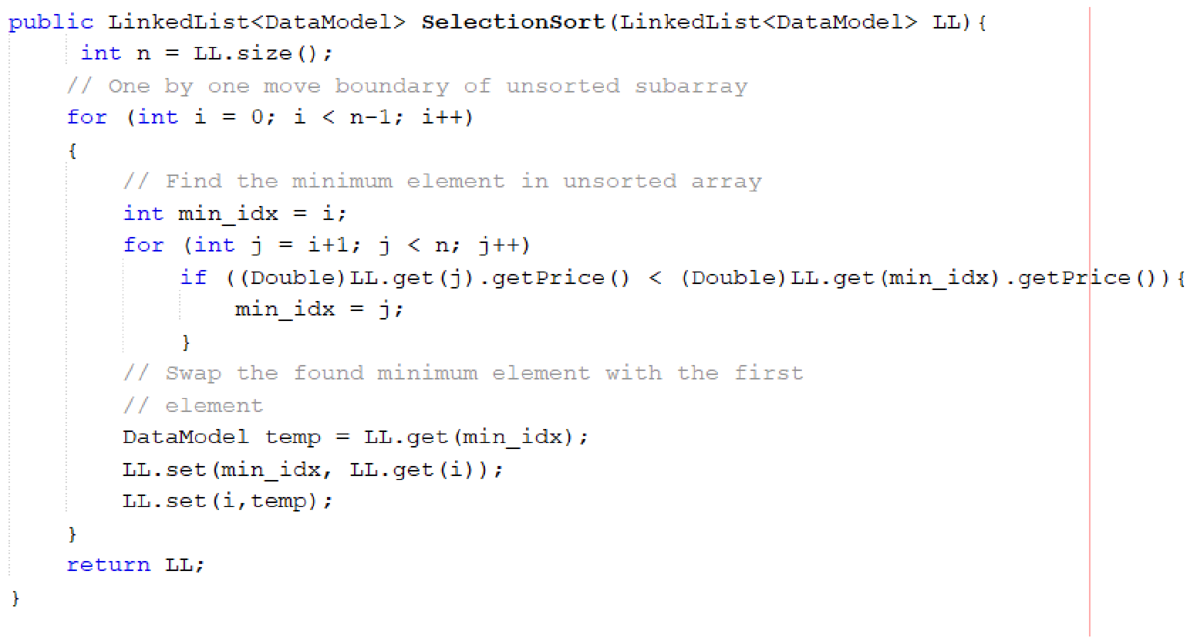
Figure 5: Linear Search Implementation

2.5. Selection Sort

The selection sort algorithm is based on the idea of finding the minimum or maximum element in an unsorted array and then putting it in its correct position in a sorted array. (heackerearth, 2020)

Process of selection sort are as follows:

- Finding the smallest number and swapping it with the first element
- Repeating the same process
- Stopping when the unsorted portion of length is 1.



```
public LinkedList<DataModel> SelectionSort(LinkedList<DataModel> LL){
     int n = LL.size();
    // One by one move boundary of unsorted subarray
    for (int i = 0; i < n-1; i++)
    {
        // Find the minimum element in unsorted array
        int min_idx = i;
        for (int j = i+1; j < n; j++)
            if ((Double)LL.get(j).getPrice() < (Double)LL.get(min_idx).getPrice()){
                min_idx = j;
            }
        // Swap the found minimum element with the first
        // element
        DataModel temp = LL.get(min_idx);
        LL.set(min_idx, LL.get(i));
        LL.set(i,temp);
    }
    return LL;
}
```

Figure 6: Selection Sort Implementation

In this project, sorting is needed for binary search algorithm, so we choose selection sort. The above figure shows the selection sort code which was implemented during the making of the project.

2.6. Methods Explanation

2.6.1. App Class

App()

This method is the class constructor of the App class. It initializes all the swing components. The values that are initially displayed in the table of the program are also added to the table in this method. It does not accept any arguments.

clearInput()

This is a private method that clears all the input fields of the program when it is called. It does not take any accept any arguments. This is a void type method.

tableModel()

This is private method that is used to access the default table model. In this method an instance of the DefaultTableModel class is stored in the model variable. The method returns the table model when it is called.

fileMenuOpenActionPerformed()

This method opens the text file containing more records and then adds them to the table. It opens a dialog box for selecting the file. It also displays error messages when an improper file is selected. It is called when the user clicks the Open option from File Menu.

fileMenuExitActionPerformed()

This method exits the program. It is called when the user clicks the Exit option from the File menu of the Program’s menu bar.

helpMenuManualActionPerformed()

This method accesses the user manual of the program which is stored in the src folder of the project. It is called when the user clicks the Manual option from the Help menu.

editMenuDeleteRowActionPerformed()

This method deletes a row from the table. The method is called when the user clicks the Delete Selected Row option from the Edit Menu of the Program’s menu bar. It deletes the selected row from the table.

editMenuClearTableActionPerformed()

This method clears all the data from the table. It is called when the user clicks the Clear Table option from the Edit menu.

btnClearActionPerformed()

This method clears all the input fields of the program by calling the clearInput method. It is called when the user clicks the Clear button.

btnAddActionPerformed()

This method extracts the input from all the text fields of the add new item section and adds the data to the table. It also displays the appropriate messages when a user tries to add incomplete or invalid data. It is called when the user clicks the add button.

btnSearchGenreActionPerformed()

This method displays the total number and the names of books of a single category (genre). It is called when the user clicks the search by genre button. It also displays an error message when the user clicks the button without selecting any genre.

btnSearchPriceActionPerformed()

This method extracts the input from the txtSearch text field where the user inputs price and searches for the books with the matching price. It creates an instance of the Algorithms class and calls the SelectionSort and BinarySearch methods for searching through the books. It is called when the user clicks the search by price button.

2.6.2. Algorithms Class

SelectionSort()

This method contains the algorithm for selection sort. It takes a linked list containing instances of the DataModel class as argument and sorts the elements of the list in ascending order using the algorithm. It returns the sorted linked list when it is called.

BinarySearch()

This method contains the algorithm for Binary Search. It takes a linked list containing instances of the DataModel class as well as values for a lowest and highest index, and a value for searching, as the arguments. It then searches the linked list for the element having the same price as the argument provided for searching using the binary search algorithm. It returns the matched element when it is called.

LinearSearch()

This method contains the algorithm for Linear Search. It takes a linked list containing instances of the DataModel class and a value for searching as the arguments. It searches through the whole list to find the element with the matching genre. It returns the matching element when it is called.

2.6.3. DataModel Class

DataModel()

This is the class constructor for the DataModel class used for making instances of the class. It takes an integer value for book id, string values for book name, author, book type, genre, publication and double value for price as the arguments.

getBookID()

This method returns the value for the bookID variable of an instance of the DataModel class when it is called. Its return type is int.

getBookName()

This method returns the value for the bookName variable of an instance of the DataModel class when it is called. Its return type is String.

getAuthor()

This method returns the value for the author variable of an instance of the DataModel class when it is called. Its return type is also String.

getBookType()

This method returns the value for the bookType variable of an instance of the DataModel class when it is called. Its return type is also String.

getGenre()

This method returns the value for the genre variable of an instance of the DataModel class when it is called. Its return type is also String.

getPublication()

This method returns the value for the publication variable of an instance of the DataModel class when it is called. Its return type is also String.

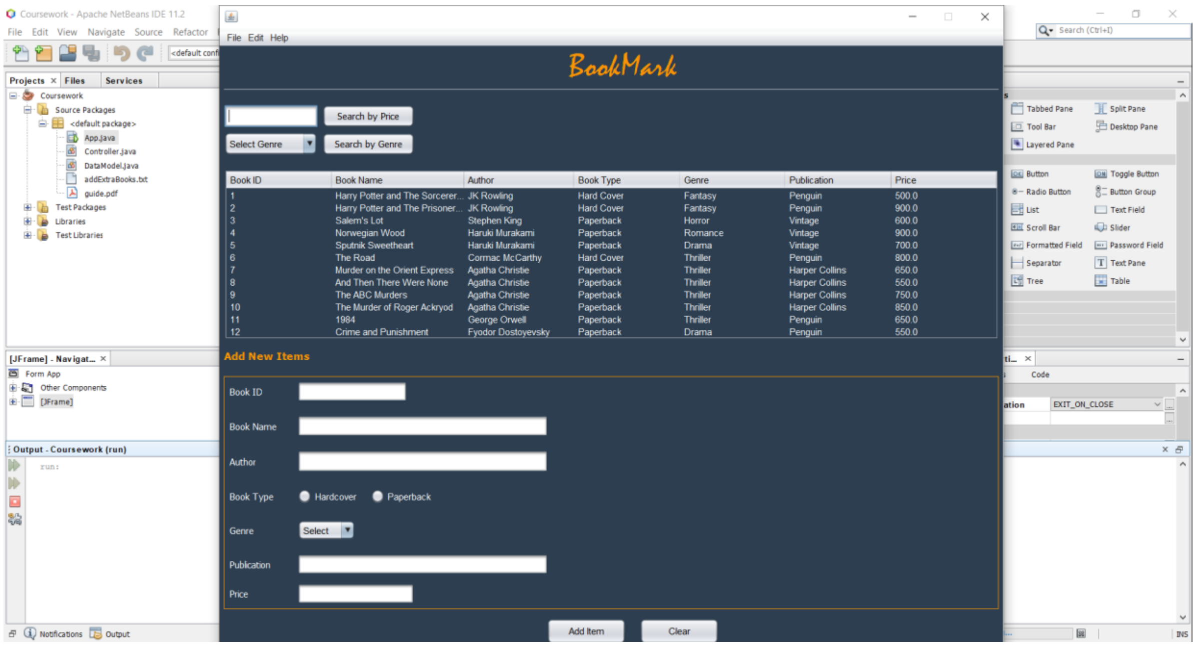
getPrice()

This method returns the value for the price variable of an instance of the DataModel class when it is called. Its return type is double.

2.7. Testing and Evidence

In the testing section, all kinds of testing are shown with screen sorts.

2.8. Program Testing



Book ID	Book Name	Author	Book Type	Genre	Publication	Price
1	Harry Potter and The Sorcerer...	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
2	Harry Potter and The Prisoner...	JK Rowling	Hard Cover	Fantasy	Penguin	900.0
3	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the Orient Express	Agatha Christie	Paperback	Thriller	Harper Collins	650.0
8	And Then There Were None	Agatha Christie	Paperback	Thriller	Harper Collins	550.0
9	The ABC Murders	Agatha Christie	Paperback	Thriller	Harper Collins	750.0
10	The Murder of Roger Ackryod	Agatha Christie	Paperback	Thriller	Harper Collins	850.0
11	1984	George Orwell	Paperback	Thriller	Penguin	650.0
12	Crime and Punishment	Fyodor Dostoyevsky	Paperback	Drama	Penguin	550.0

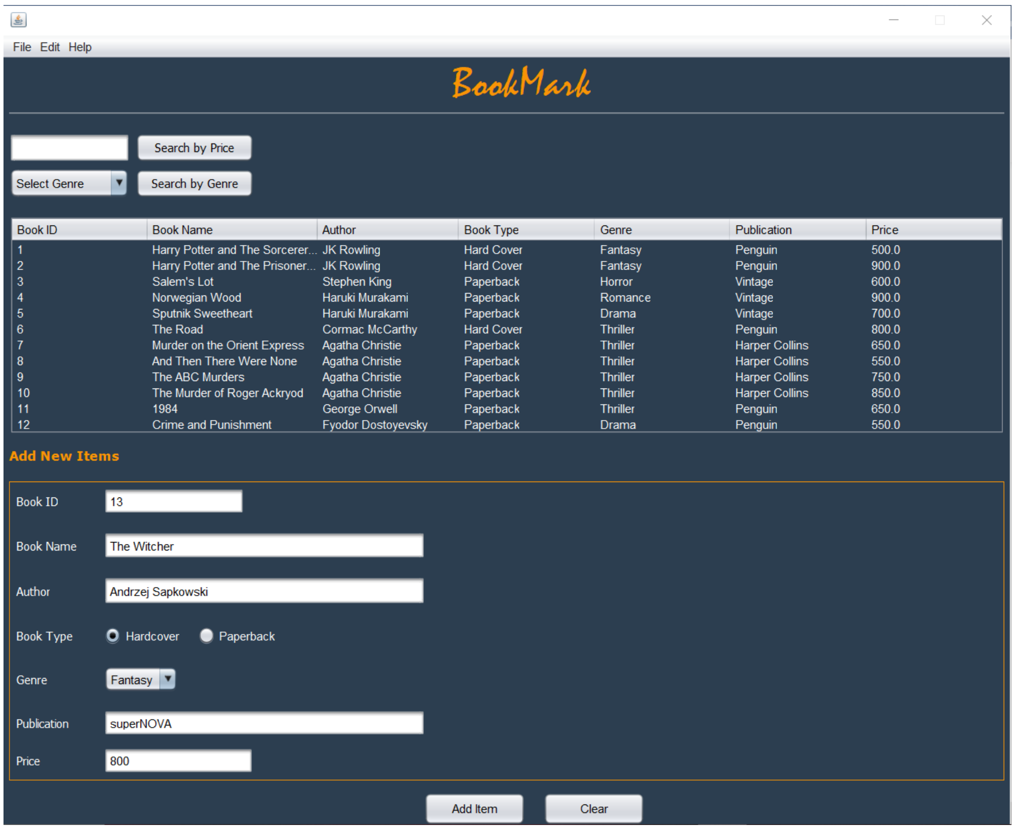
Figure 7: Program Testing

Test No.	1
Action	Running the program in the NetBeans IDE.
Expected Result	The program would run on NetBeans successfully without any error.
Actual Result	The program ran successfully.
Test Result	The test was successful.

Table 2: Test 1

2.9. Feature Testing

2.9.1. Adding Items to table

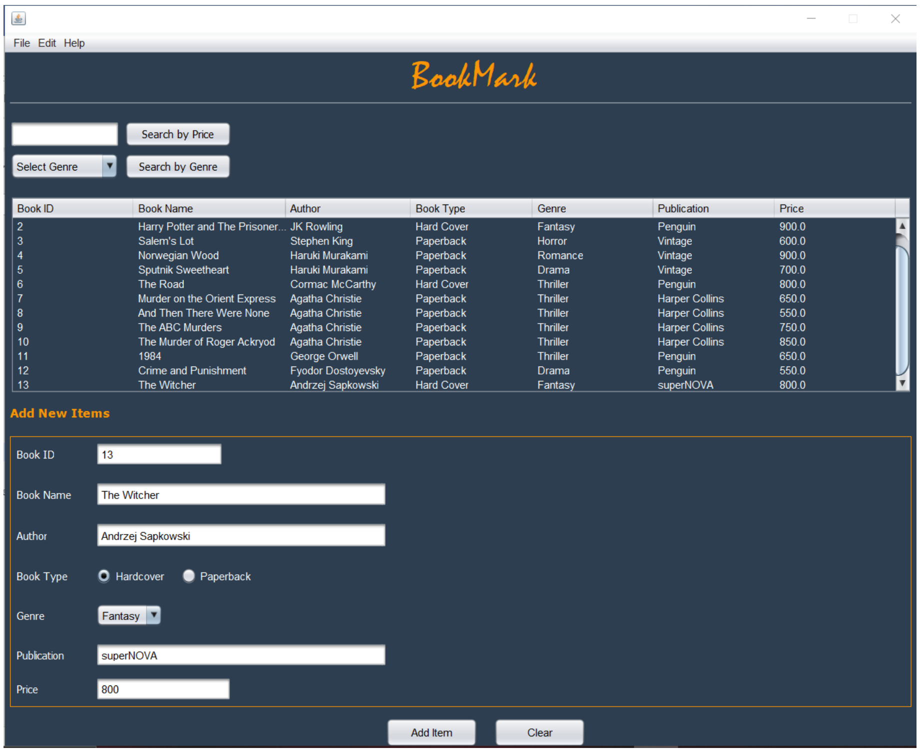


Book ID	Book Name	Author	Book Type	Genre	Publication	Price
1	Harry Potter and The Sorcerer...	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
2	Harry Potter and The Prisoner...	JK Rowling	Hard Cover	Fantasy	Penguin	900.0
3	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the Orient Express	Agatha Christie	Paperback	Thriller	Harper Collins	650.0
8	And Then There Were None	Agatha Christie	Paperback	Thriller	Harper Collins	550.0
9	The ABC Murders	Agatha Christie	Paperback	Thriller	Harper Collins	750.0
10	The Murder of Roger Ackryod	Agatha Christie	Paperback	Thriller	Harper Collins	850.0
11	1984	George Orwell	Paperback	Thriller	Penguin	650.0
12	Crime and Punishment	Fyodor Dostoyevsky	Paperback	Drama	Penguin	550.0

Add New Items

Book ID	13
Book Name	The Witcher
Author	Andrzej Sapkowski
Book Type	Hardcover / Paperback
Genre	Fantasy
Publication	superNOVA
Price	800

Figure 8: Test 2



Book ID	Book Name	Author	Book Type	Genre	Publication	Price
2	Harry Potter and The Prisoner...	JK Rowling	Hard Cover	Fantasy	Penguin	900.0
3	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the Orient Express	Agatha Christie	Paperback	Thriller	Harper Collins	650.0
8	And Then There Were None	Agatha Christie	Paperback	Thriller	Harper Collins	550.0
9	The ABC Murders	Agatha Christie	Paperback	Thriller	Harper Collins	750.0
10	The Murder of Roger Ackryod	Agatha Christie	Paperback	Thriller	Harper Collins	850.0
11	1984	George Orwell	Paperback	Thriller	Penguin	650.0
12	Crime and Punishment	Fyodor Dostoyevsky	Paperback	Drama	Penguin	550.0
13	The Witcher	Andrzej Sapkowski	Hard Cover	Fantasy	superNOVA	800.0

Add New Items

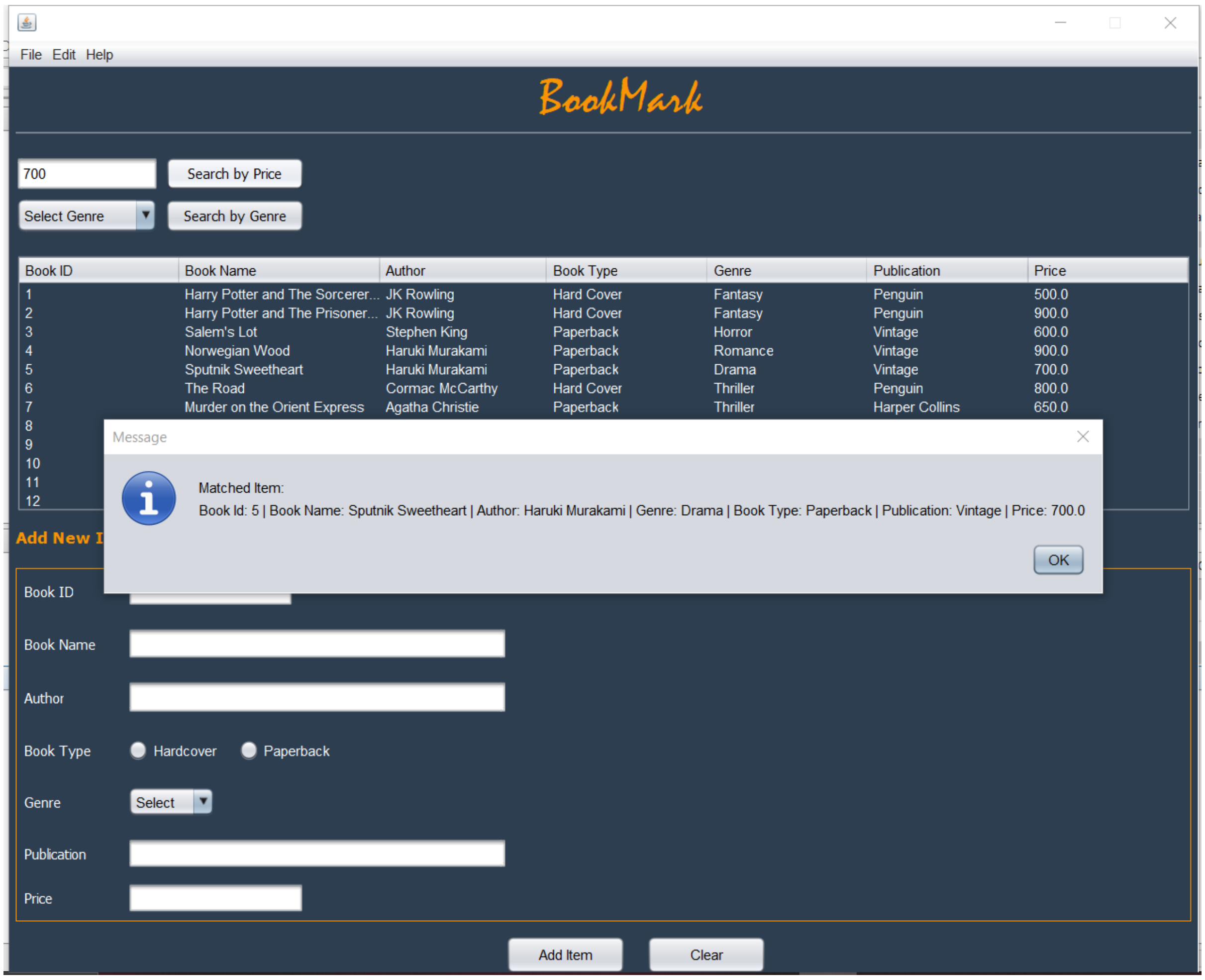
Book ID	13
Book Name	The Witcher
Author	Andrzej Sapkowski
Book Type	Hardcover / Paperback
Genre	Fantasy
Publication	superNOVA
Price	800

Figure 9: Test 2

Test No.	2
Action	Adding new item in the table by clicking add item button.
Expected Result	The button would add new item on the table.
Actual Result	New item was successfully added on the table by clicking add item button.
Test Result	The test was successful.

Table 3: Test 2

2.9.2. Searching item based on price



Book ID	Book Name	Author	Book Type	Genre	Publication	Price
1	Harry Potter and The Sorcerer...	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
2	Harry Potter and The Prisoner...	JK Rowling	Hard Cover	Fantasy	Penguin	900.0
3	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the Orient Express	Agatha Christie	Paperback	Thriller	Harper Collins	650.0
8						
9						
10						
11						
12						

Message

Matched Item:
Book Id: 5 | Book Name: Sputnik Sweetheart | Author: Haruki Murakami | Genre: Drama | Book Type: Paperback | Publication: Vintage | Price: 700.0

OK

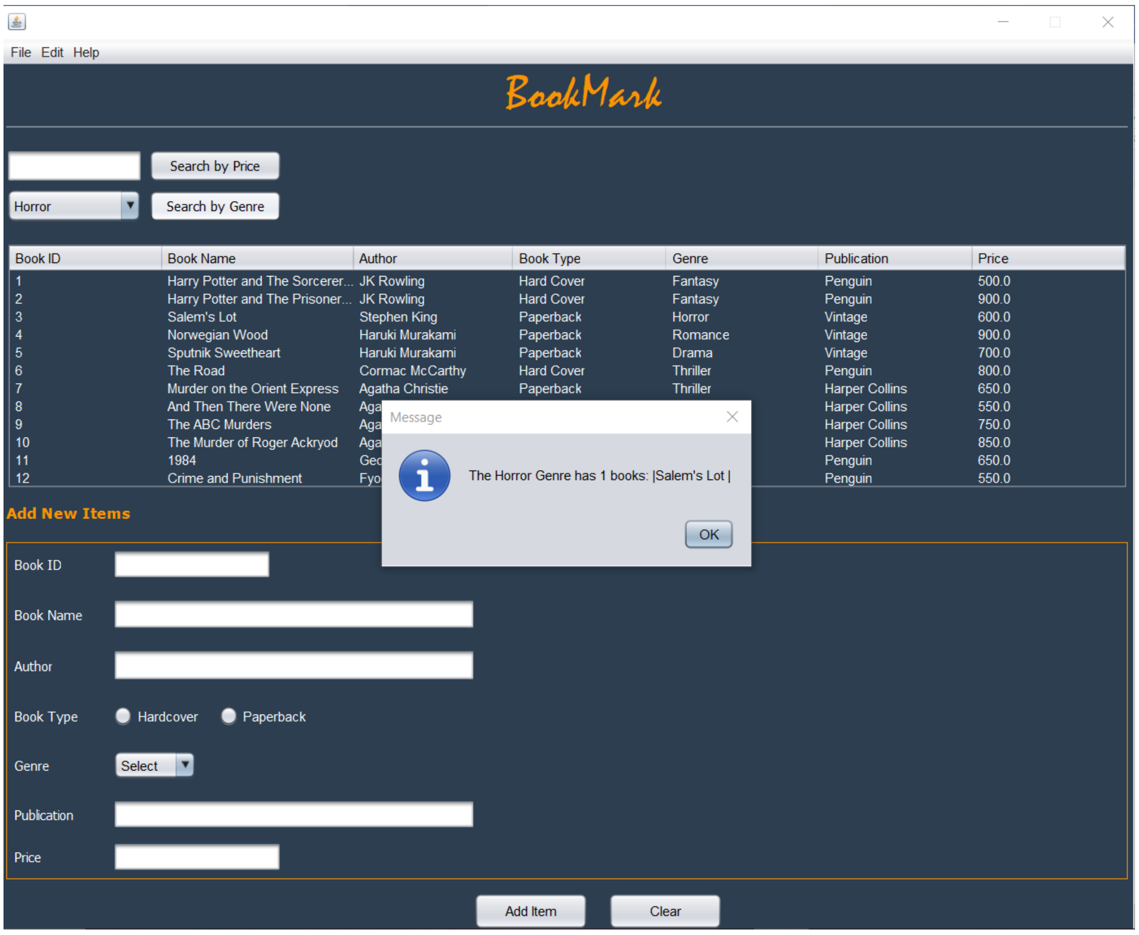
Book ID	
Book Name	
Author	
Book Type	Hardcover / Paperback
Genre	Select
Publication	
Price	

Figure 10: Test 3

Test No.	3
Action	Searching item in the table based by price.
Expected Result	The search by price button would search the book from the table based on price.
Actual Result	The search by Price button successfully worked and details of book was shown.
Test Result	The test was successful.

Table 4: Test 3

2.9.3. Searching for number of books in category



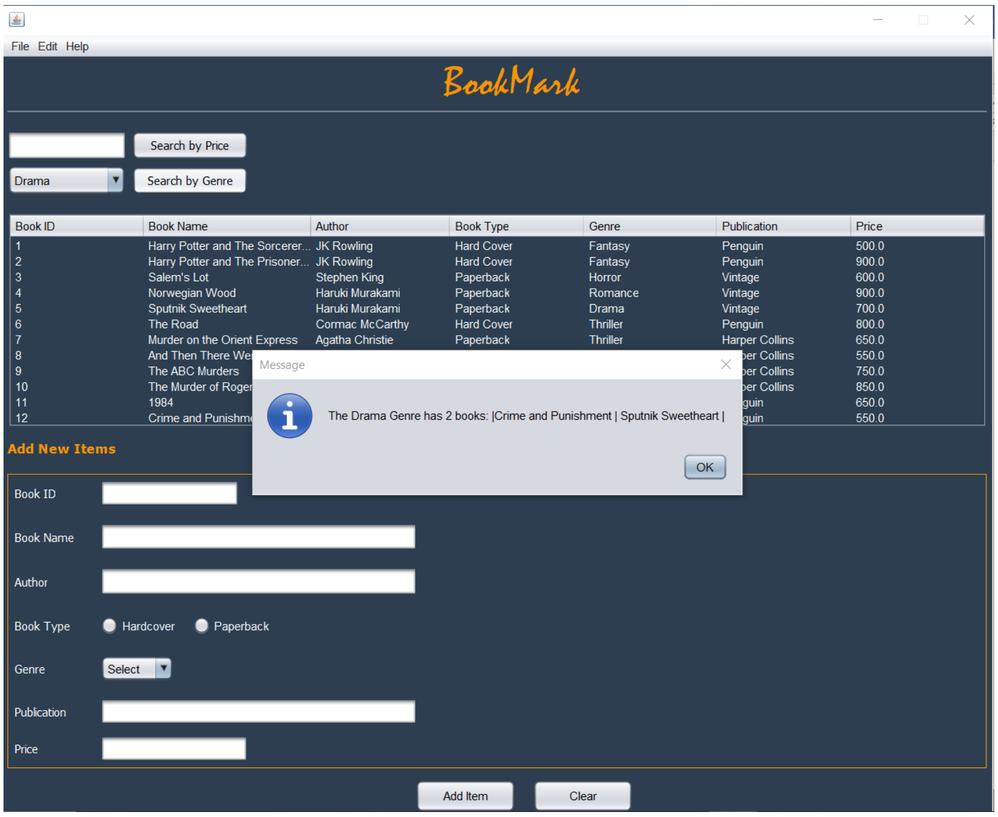
Book ID	Book Name	Author	Book Type	Genre	Publication	Price
1	Harry Potter and The Sorcerer...	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
2	Harry Potter and The Prisoner...	JK Rowling	Hard Cover	Fantasy	Penguin	900.0
3	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the Orient Express	Agatha Christie	Paperback	Thriller	Harper Collins	650.0
8	And Then There Were None	Aga			Harper Collins	550.0
9	The ABC Murders	Aga			Harper Collins	750.0
10	The Murder of Roger Ackryod	Aga			Harper Collins	850.0
11	1984	Geo			Penguin	650.0
12	Crime and Punishment	Fyo			Penguin	550.0

Message

The Horror Genre has 1 books: |Salem's Lot |

OK

Figure 11: Test 4



Book ID	Book Name	Author	Book Type	Genre	Publication	Price
1	Harry Potter and The Sorcerer...	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
2	Harry Potter and The Prisoner...	JK Rowling	Hard Cover	Fantasy	Penguin	900.0
3	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the Orient Express	Agatha Christie	Paperback	Thriller	Harper Collins	650.0
8	And Then There We				per Collins	550.0
9	The ABC Murders				per Collins	750.0
10	The Murder of Roger				per Collins	850.0
11	1984				guin	650.0
12	Crime and Punishm				guin	550.0

Message

The Drama Genre has 2 books: |Crime and Punishment | Sputnik Sweetheart |

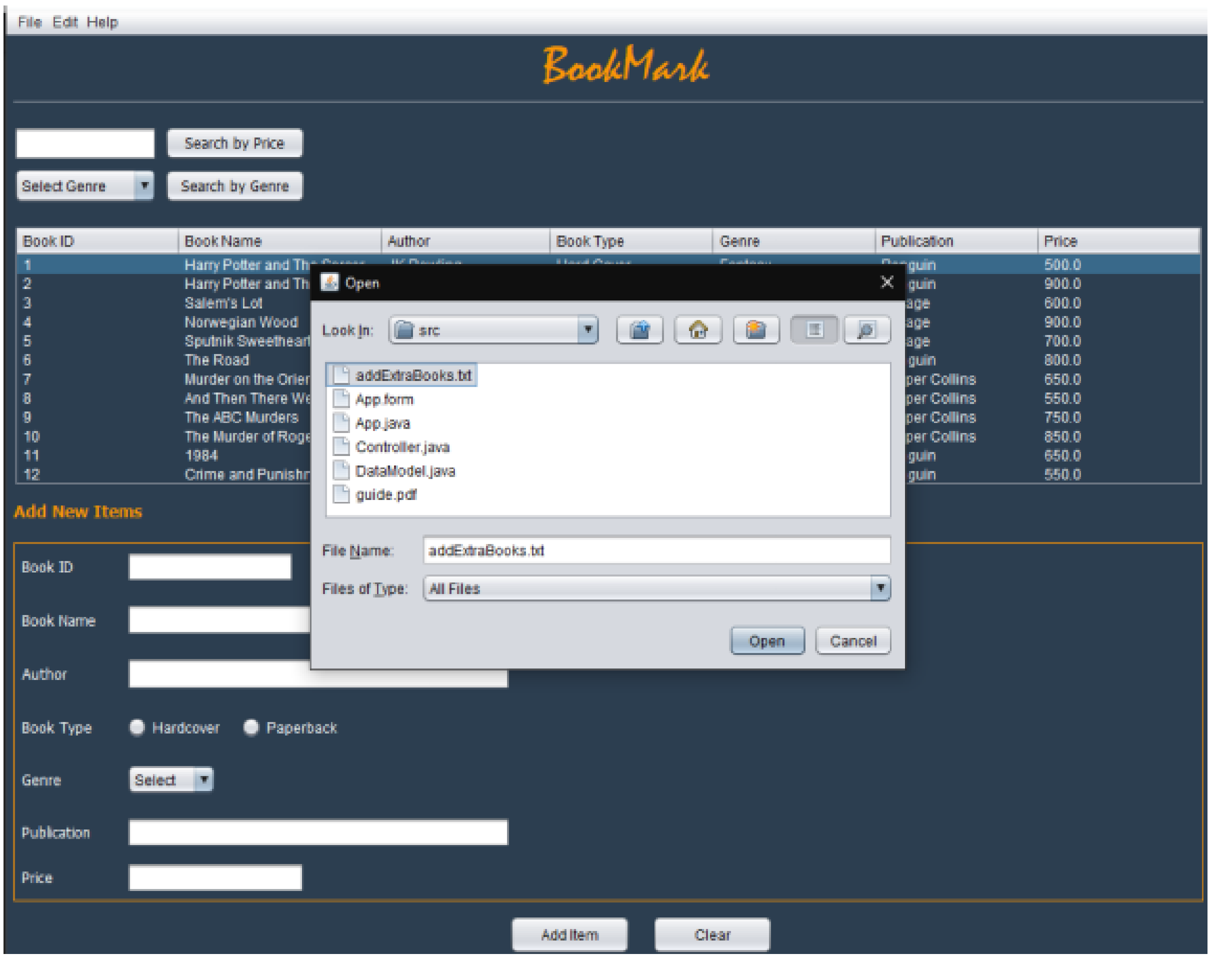
OK

Figure 12 Test 4

Test No.	4
Action	Searching for number of books in category.
Expected Result	The search by Genre button would search the number of books in category from the table based on genre.
Actual Result	The search by Genre button successfully worked and number of books on a genre was shown on a windows panel.
Test Result	The test was successful.

Table 5 Test 4

2.9.4. Opening a file from menu

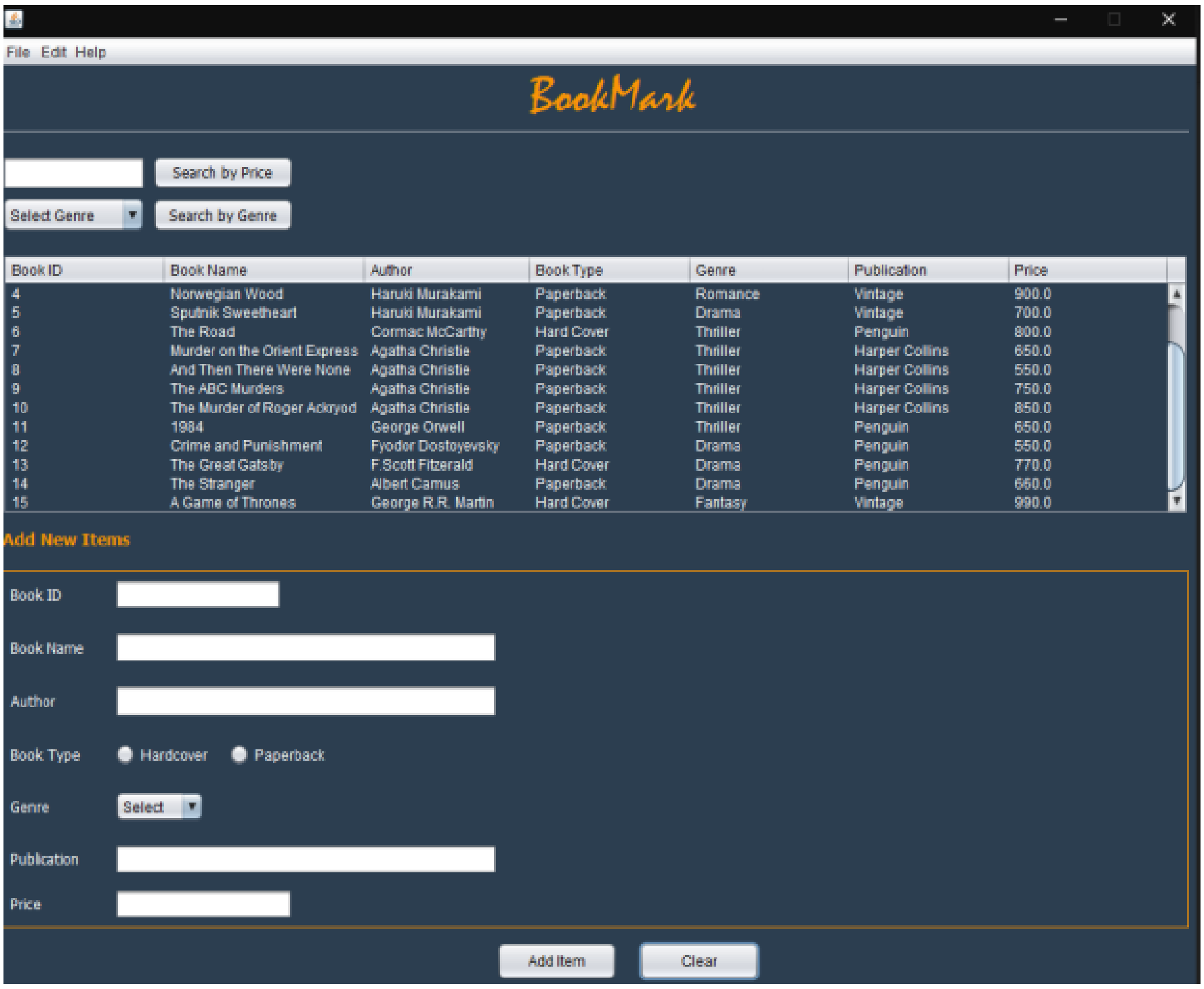


Book ID	Book Name	Author	Book Type	Genre	Publication	Price
1	Harry Potter and Th	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
2	Harry Potter and Th				guin	900.0
3	Salem's Lot				age	600.0
4	Norwegian Wood				age	900.0
5	Sputnik Sweetheart				age	700.0
6	The Road				guin	800.0
7	Murder on the Orien				per Collins	650.0
8	And Then There We				per Collins	550.0
9	The ABC Murders				per Collins	750.0
10	The Murder of Roge				per Collins	850.0
11	1984				guin	650.0
12	Crime and Punishm				guin	550.0

File Name: addExtraBooks.txt

Files of Type: All Files

Figure 13: Test 5



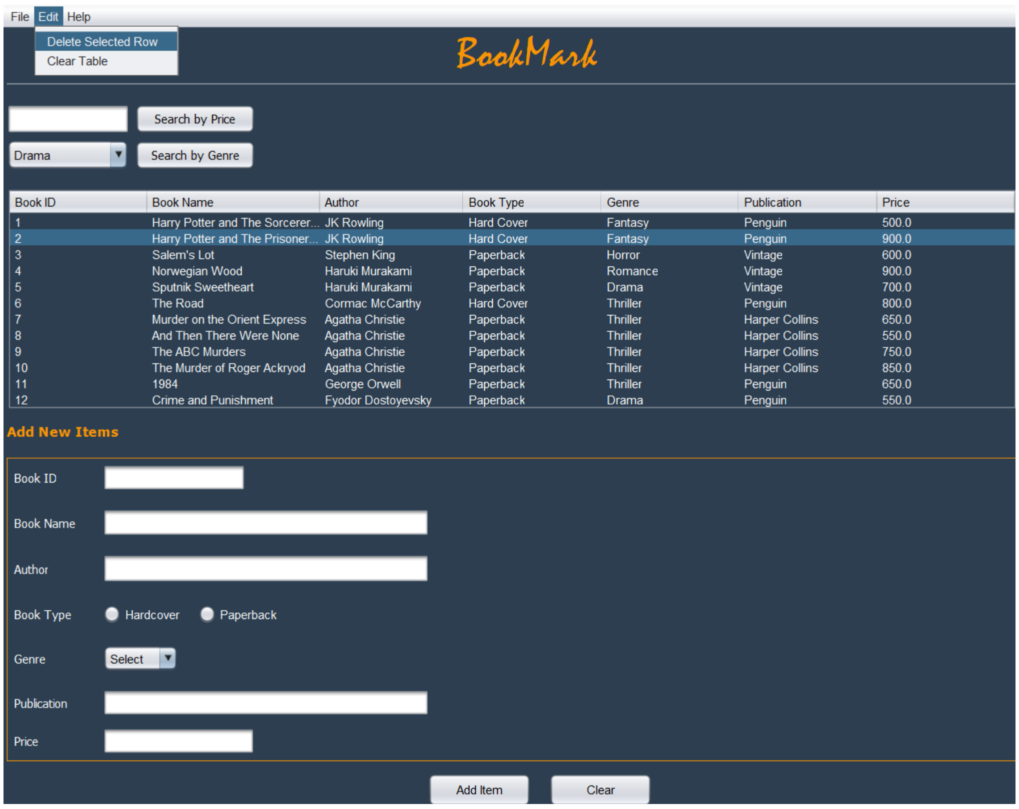
Book ID	Book Name	Author	Book Type	Genre	Publication	Price
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the Orient Express	Agatha Christie	Paperback	Thriller	Harper Collins	650.0
8	And Then There Were None	Agatha Christie	Paperback	Thriller	Harper Collins	550.0
9	The ABC Murders	Agatha Christie	Paperback	Thriller	Harper Collins	750.0
10	The Murder of Roger Ackryod	Agatha Christie	Paperback	Thriller	Harper Collins	850.0
11	1984	George Orwell	Paperback	Thriller	Penguin	650.0
12	Crime and Punishment	Fyodor Dostoyevsky	Paperback	Drama	Penguin	550.0
13	The Great Gatsby	F.Scott Fitzerald	Hard Cover	Drama	Penguin	770.0
14	The Stranger	Albert Camus	Paperback	Drama	Penguin	660.0
15	A Game of Thrones	George R.R. Martin	Hard Cover	Fantasy	Vintage	990.0

Figure 14: Test 5

Test No.	5
Action	Opening a file from menu
Expected Result	The opened file would be added on the book table.
Actual Result	The file was successfully added on the table.
Test Result	The test was successful.

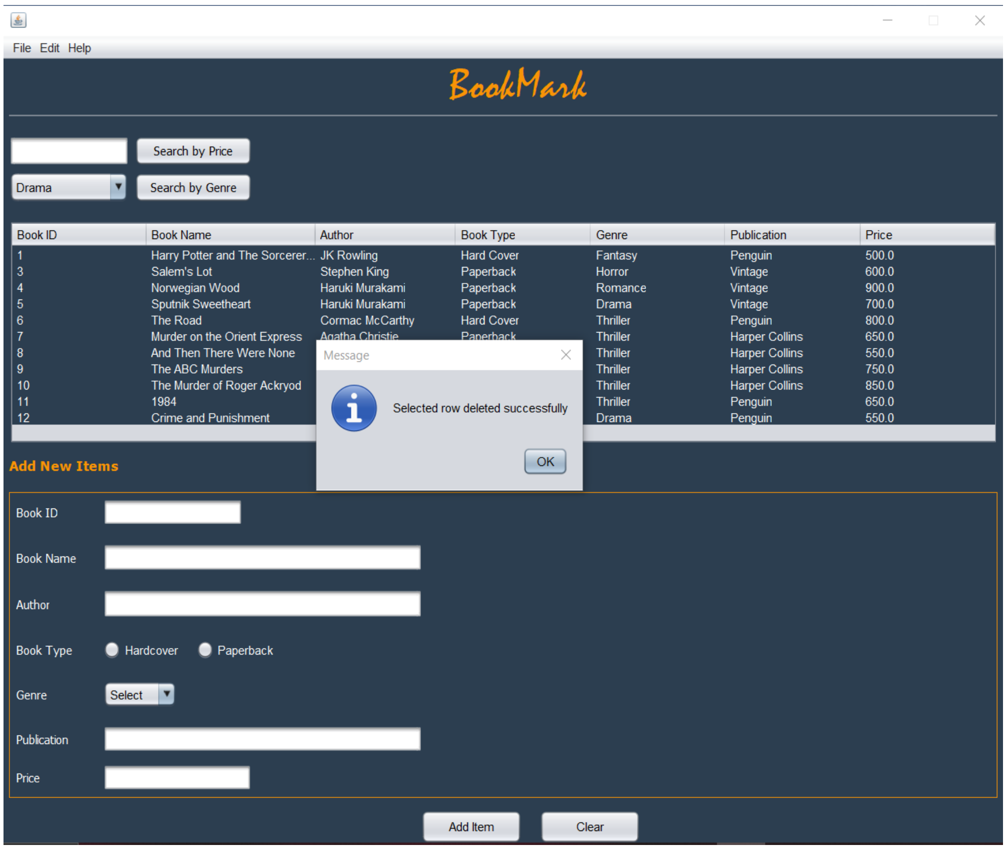
Table 6: Test 5

2.9.5. Delete Selected Row



Book ID	Book Name	Author	Book Type	Genre	Publication	Price
1	Harry Potter and The Sorcerer...	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
2	Harry Potter and The Prisoner...	JK Rowling	Hard Cover	Fantasy	Penguin	900.0
3	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the Orient Express	Agatha Christie	Paperback	Thriller	Harper Collins	650.0
8	And Then There Were None	Agatha Christie	Paperback	Thriller	Harper Collins	550.0
9	The ABC Murders	Agatha Christie	Paperback	Thriller	Harper Collins	750.0
10	The Murder of Roger Ackryod	Agatha Christie	Paperback	Thriller	Harper Collins	850.0
11	1984	George Orwell	Paperback	Thriller	Penguin	650.0
12	Crime and Punishment	Fyodor Dostoyevsky	Paperback	Drama	Penguin	550.0

Figure 15: Test 5



Selected row deleted successfully

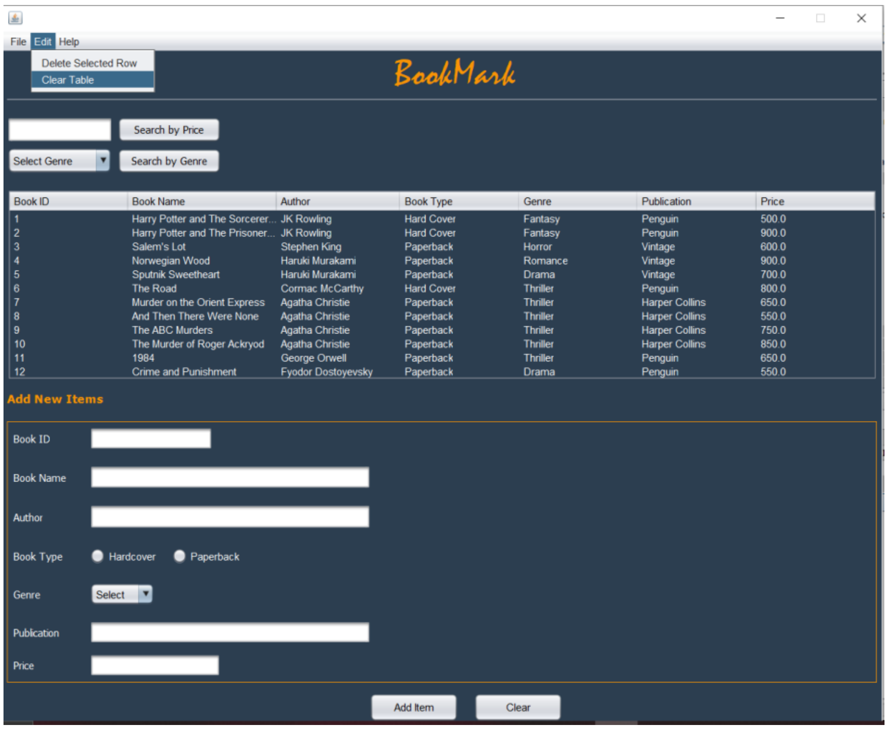
Book ID	Book Name	Author	Book Type	Genre	Publication	Price
1	Harry Potter and The Sorcerer...	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
3	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the Orient Express	Agatha Christie	Paperback	Thriller	Harper Collins	650.0
8	And Then There Were None			Thriller	Harper Collins	550.0
9	The ABC Murders			Thriller	Harper Collins	750.0
10	The Murder of Roger Ackryod			Thriller	Harper Collins	850.0
11	1984			Thriller	Penguin	650.0
12	Crime and Punishment			Drama	Penguin	550.0

Figure 16: Test 6

Test No.	6
Action	Delete the selected row in the table.
Expected Result	Delete selected row option in the menu bar would delete the row from the table.
Actual Result	The selected row was deleted from the table
Test Result	The test was successful.

Table 7: Test 6

2.9.6. Clear Table



Book ID	Book Name	Author	Book Type	Genre	Publication	Price
1	Harry Potter and The Sorcerer...	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
2	Harry Potter and The Prisoner...	JK Rowling	Hard Cover	Fantasy	Penguin	900.0
3	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the Orient Express	Agatha Christie	Paperback	Thriller	Harper Collins	650.0
8	And Then There Were None	Agatha Christie	Paperback	Thriller	Harper Collins	550.0
9	The ABC Murders	Agatha Christie	Paperback	Thriller	Harper Collins	750.0
10	The Murder of Roger Ackryod	Agatha Christie	Paperback	Thriller	Harper Collins	850.0
11	1984	George Orwell	Paperback	Thriller	Penguin	650.0
12	Crime and Punishment	Fyodor Dostoyevsky	Paperback	Drama	Penguin	550.0

Figure 17: Test 7

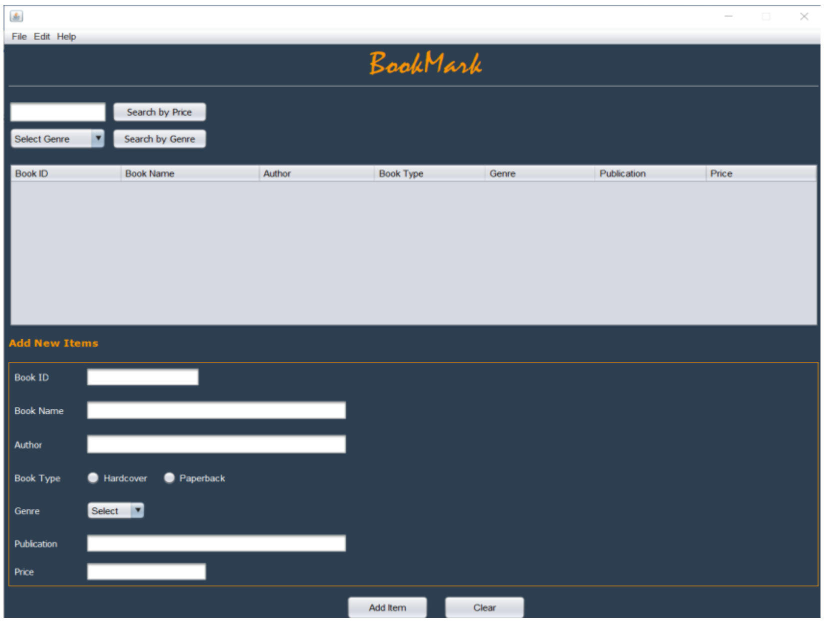
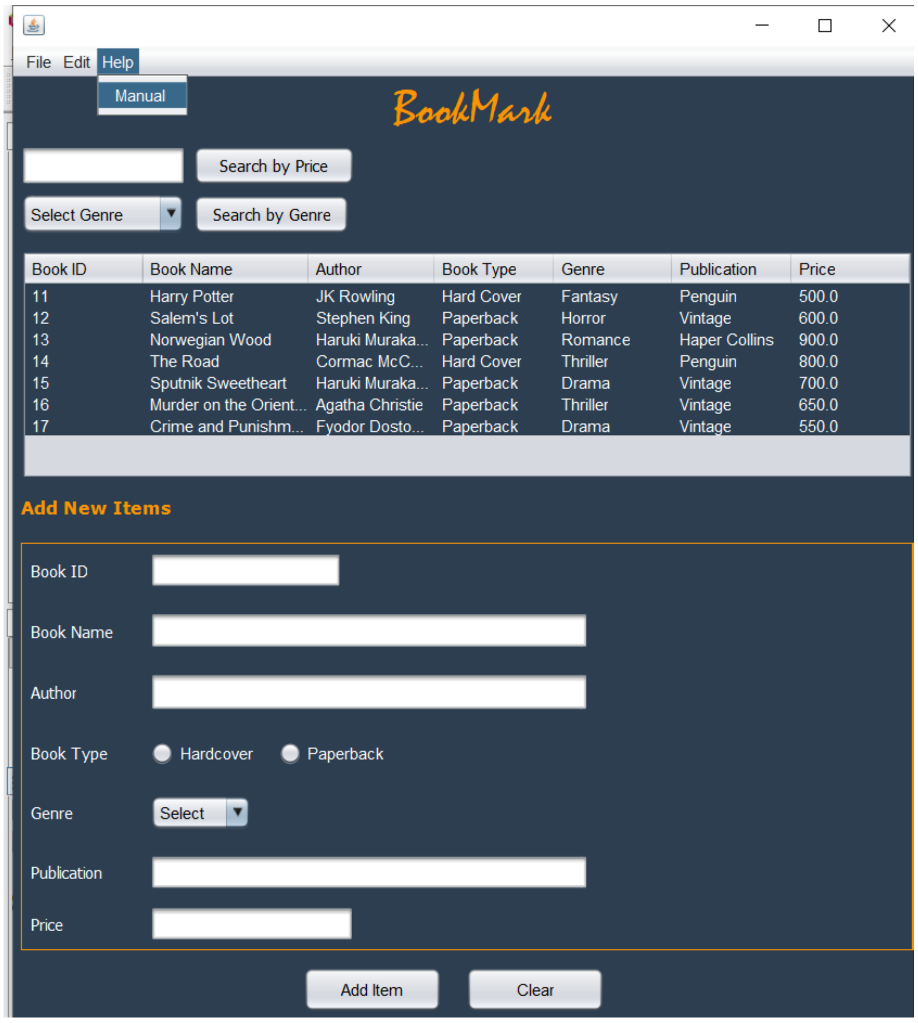


Figure 18: Test 7

Test No.	7
Action	Clear the entire table.
Expected Result	Clear table option in the menu bar would delete the row from the table.
Actual Result	The entire table was deleted.
Test Result	The test was successful.

Table 8 Test 7

2.9.7. Open Manual



Book ID	Book Name	Author	Book Type	Genre	Publication	Price
11	Harry Potter	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
12	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
13	Norwegian Wood	Haruki Muraka...	Paperback	Romance	Haper Collins	900.0
14	The Road	Cormac McC...	Hard Cover	Thriller	Penguin	800.0
15	Sputnik Sweetheart	Haruki Muraka...	Paperback	Drama	Vintage	700.0
16	Murder on the Orient...	Agatha Christie	Paperback	Thriller	Vintage	650.0
17	Crime and Punishm...	Fyodor Dosto...	Paperback	Drama	Vintage	550.0

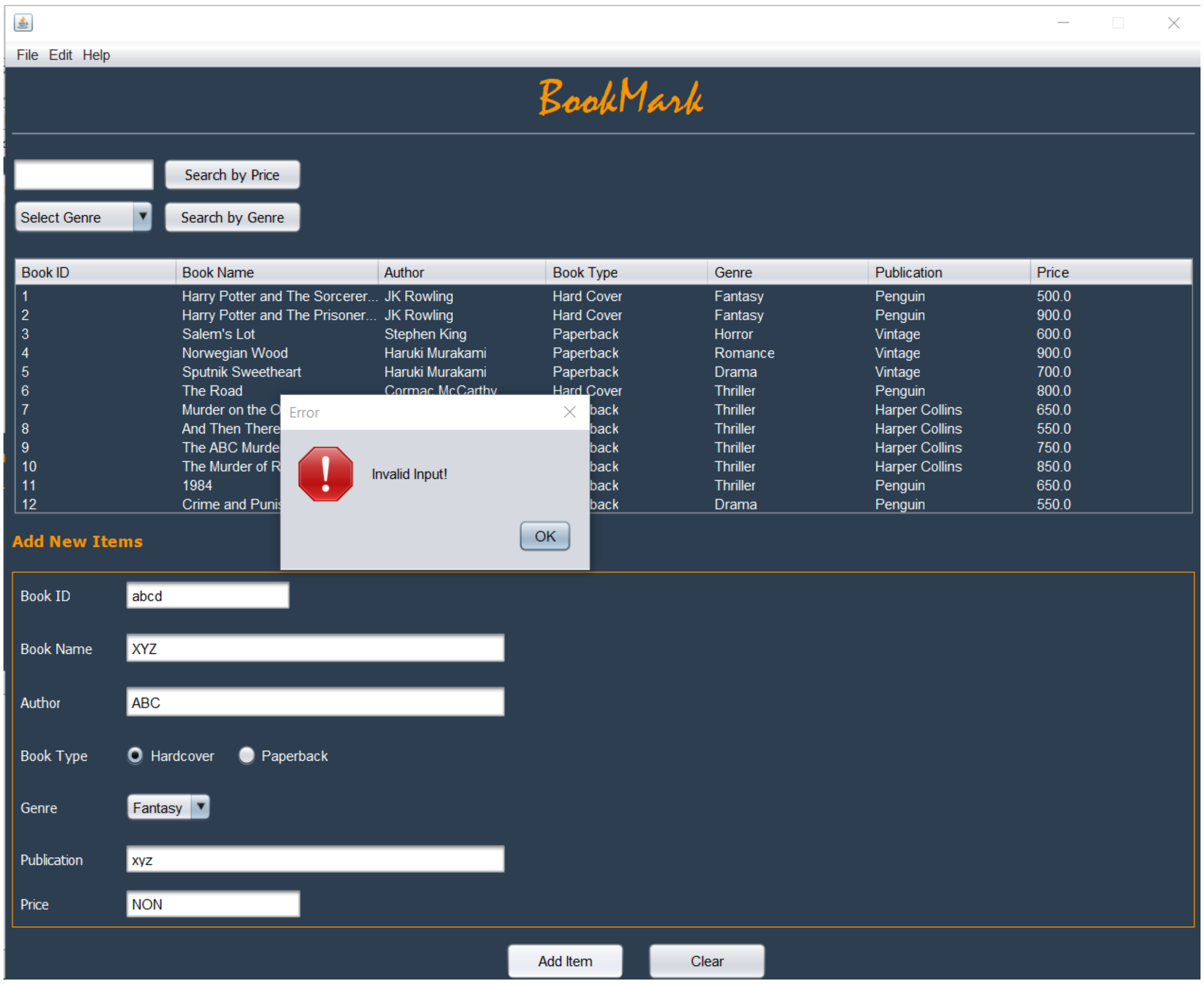
Figure 19 Test 8

Test No.	8
Action	Open User Manual while clicking manual
Expected Result	A pdf file would open when clicking manual
Actual Result	A pdf file opened when clicking manual
Test Result	The test was successful.

Table 9 Test 8

2.10. Validation Testing

2.10.1. Unsuitable values entered



Book ID	Book Name	Author	Book Type	Genre	Publication	Price
1	Harry Potter and The Sorcerer...	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
2	Harry Potter and The Prisoner...	JK Rowling	Hard Cover	Fantasy	Penguin	900.0
3	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the C		back	Thriller	Harper Collins	650.0
8	And Then There		back	Thriller	Harper Collins	550.0
9	The ABC Murde		back	Thriller	Harper Collins	750.0
10	The Murder of R		back	Thriller	Harper Collins	850.0
11	1984		back	Thriller	Penguin	650.0
12	Crime and Punis		back	Drama	Penguin	550.0

Error

Invalid Input!

OK

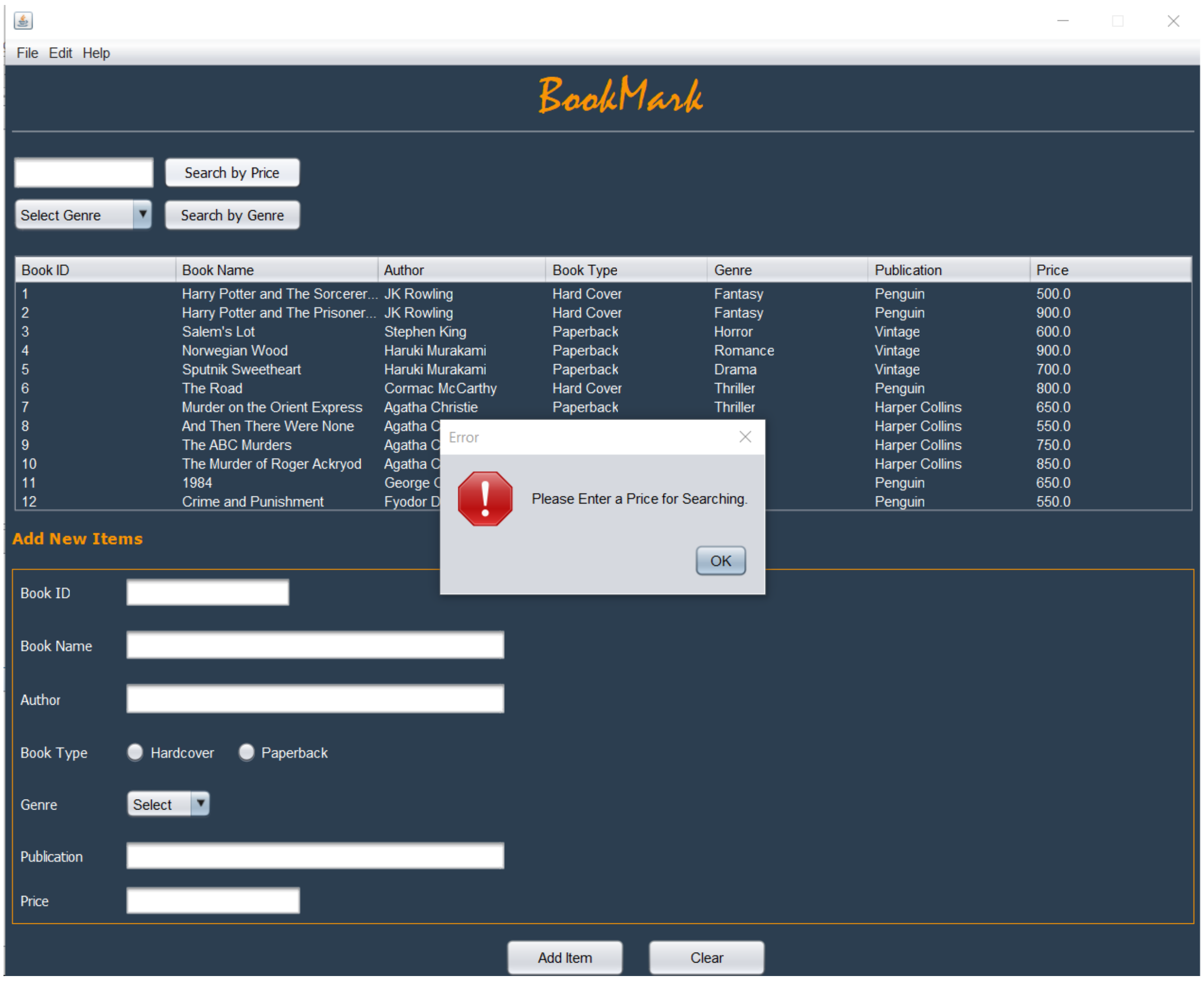
Book ID	abcd
Book Name	XYZ
Author	ABC
Book Type	[✓ Hardcover] Paperback
Genre	Fantasy
Publication	xyz
Price	NON

Figure 20: Validation 1

Test No.	9
Action	Adding item in table using unsuitable values.
Expected Result	An error message would pop up.
Actual Result	An error message popped up while adding item in table with unsuitable data.
Test Result	The test was successful.

Table 10: Test 9

2.10.2. No value entered



Book ID	Book Name	Author	Book Type	Genre	Publication	Price
1	Harry Potter and The Sorcerer...	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
2	Harry Potter and The Prisoner...	JK Rowling	Hard Cover	Fantasy	Penguin	900.0
3	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the Orient Express	Agatha Christie	Paperback	Thriller	Harper Collins	650.0
8	And Then There Were None	Agatha C			Harper Collins	550.0
9	The ABC Murders	Agatha C			Harper Collins	750.0
10	The Murder of Roger Ackryod	Agatha C			Harper Collins	850.0
11	1984	George C			Penguin	650.0
12	Crime and Punishment	Fyodor D			Penguin	550.0

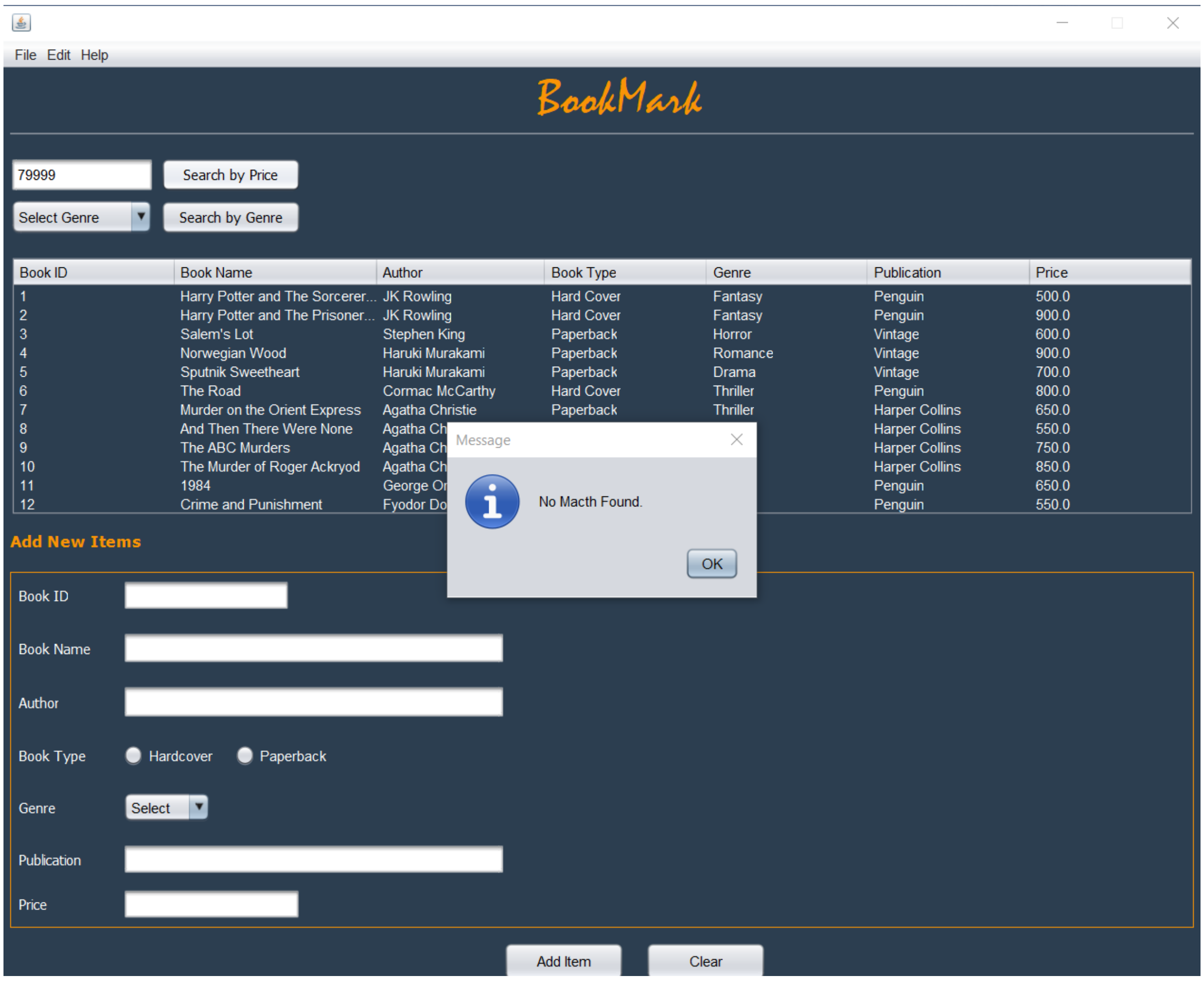
Please Enter a Price for Searching.

Figure 21: Validation 2

Test No.	10
Action	Searching by price in table without adding any details.
Expected Result	An error message would pop up.
Actual Result	An error message popped up while searching by price in table without any details.
Test Result	The test was successful.

Table 11: Test 10

2.10.3. Searching unregistered book



Book ID	Book Name	Author	Book Type	Genre	Publication	Price
1	Harry Potter and The Sorcerer...	JK Rowling	Hard Cover	Fantasy	Penguin	500.0
2	Harry Potter and The Prisoner...	JK Rowling	Hard Cover	Fantasy	Penguin	900.0
3	Salem's Lot	Stephen King	Paperback	Horror	Vintage	600.0
4	Norwegian Wood	Haruki Murakami	Paperback	Romance	Vintage	900.0
5	Sputnik Sweetheart	Haruki Murakami	Paperback	Drama	Vintage	700.0
6	The Road	Cormac McCarthy	Hard Cover	Thriller	Penguin	800.0
7	Murder on the Orient Express	Agatha Christie	Paperback	Thriller	Harper Collins	650.0
8	And Then There Were None	Agatha Ch			Harper Collins	550.0
9	The ABC Murders	Agatha Ch			Harper Collins	750.0
10	The Murder of Roger Ackryod	Agatha Ch			Harper Collins	850.0
11	1984	George Or			Penguin	650.0
12	Crime and Punishment	Fyodor Do			Penguin	550.0

Message

No Macth Found.

Figure 22: Validation 3

Test No.	11
Action	Searching unregistered book.
Expected Result	No match message would pop up.
Actual Result	No match message popped up while searching unregistered book by price.
Test Result	The test was successful.

Table 12: Test 11

2.11. Conclusion

The groupwork has been completed in time. All requirements have been met and everything has been included in the documentation.

This groupwork has been a significant experience for our group during this semester. We learned about and practiced various searching and sorting algorithms that helped us improve our coding skills. We even gained experience on working with a group on a programming project. The project also helped us improve on our weaknesses and helped use realize our mistakes in the programming projects that we had done in our previous year. We were also able to use our knowledge that we gained from the Software Engineering module in this groupwork for planning out the program.

In conclusion, this coursework has been very helpful to us as it provided us with a way to apply our new gained knowledge in practice. We have learned a lot by working on this assignment. The knowledge that we have gained will be very useful to us in the future.

2.12. References

geeksforgeeks. (2020) *GeeksforGeeks* [Online]. Available from: https://www.geeksforgeeks.org/binary-search/ [Accessed 13 January 2020].

heackerearth. (2020) *Selection Sort* [Online]. Available from: https://l.messenger.com/l.php?u=https%3A%2F%2Fwww.hackerearth.com%2Fpractice%2Falgorithms%2Fsorting%2Fselection-sort%2Ftutorial%2F&h=AT1qS7pM-fZq6_-ljh0AjKZBeI9PsmZY6Q5jqtZKgO76istI6A4oEA8e-fos3rO-7ioTQRx6NCiF19eEPrMrF60tVzuzQiZK0QdwqIf35e3Qahvbok_emE_0u2 [Accessed 15 Januuary 2020].

Jeet Singh, A.P. (2017) *Devopedia* [Online]. Available from: https://devopedia.org/binary-search [Accessed 12 January 2020].

Sehgal, K. (2017) *Medium* [Online]. Available from: https://medium.com/karuna-sehgal/an-simplified-explanation-of-linear-search-5056942ba965 [Accessed 13 January 2020].

2.13. Bibliography

Cutajar, J. (2018) *Beginning Java Data Structures and Algorithms*. 1st ed. Mumbai: Packt Publising.

Puntambekar, A.A. (2010) *Design and Analysis of Algorithm*. 1st ed. Pune: Technical Publication Pune.

Weiss, M.A. (2002) *Data Structures and Problem Solving Using Java*. 2nd ed. Florida: Addison Wesley.

2.14. Appendix

2.14.1. Class: App

```
import java.awt.Desktop;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.io.IOException;

import java.util.*;

import javax.swing.JFileChooser;

import javax.swing.JOptionPane;

import javax.swing.table.*;


public final class App extends javax.swing.JFrame {

   LinkedList<DataModel>LL=new LinkedList<>();

   public App() {

      initComponents();

      LL.add(0,new DataModel(1,"Harry Potter and The Sorcerer's Stone","JK Rowling","Hard Cover","Fantasy","Penguin",500));

      LL.add(1,new DataModel(2,"Harry Potter and The Prisoner of Azkaban","JK Rowling","Hard Cover","Fantasy","Penguin",900));

      LL.add(2,new DataModel(3,"Salem's Lot","Stephen King","Paperback","Horror","Vintage",600));

      LL.add(3,new DataModel(4,"Norwegian Wood","Haruki Murakami","Paperback","Romance","Vintage",900));

      LL.add(4,new DataModel(5,"Sputnik Sweetheart","Haruki Murakami","Paperback","Drama","Vintage",700));
      LL.add(5,new DataModel(6,"The Road","Cormac McCarthy","Hard Cover","Thriller","Penguin",800));

      LL.add(6,new DataModel(7,"Murder on the Orient Express","Agatha Christie","Paperback","Thriller","Harper Collins",650));

      LL.add(7,new DataModel(8,"And Then There Were None","Agatha Christie","Paperback","Thriller","Harper Collins",550));

      LL.add(8,new DataModel(9,"The ABC Murders","Agatha Christie","Paperback","Thriller","Harper Collins",750));

      LL.add(9,new DataModel(10,"The Murder of Roger Ackryod","Agatha Christie","Paperback","Thriller","Harper Collins",850));

      LL.add(10,new DataModel(11,"1984","George Orwell","Paperback","Thriller","Penguin",650));

      LL.add(11,new DataModel(12,"Crime and Punishment","Fyodor Dostoyevsky","Paperback","Drama","Penguin",550));


      for (int i = 0; i <LL.size(); i++){
          int bookID = LL.get(i).getBookID();
          String bookName = LL.get(i).getBookName();
          String author = LL.get(i).getAuthor();
          String bookType= LL.get(i).getBookType();
          String genre= LL.get(i).getGenre();
          String publication= LL.get(i).getPublication();
          double price = LL.get(i).getPrice();
          Object[] rowData = {bookID,bookName,author,bookType,genre,publication,price};
          tableModel().addRow(rowData);
      }
```

```
}

@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    btnGrpType = new javax.swing.ButtonGroup();
    mainPanel = new javax.swing.JPanel();
    txtSearch = new javax.swing.JTextField();
    btnSearchPrice = new javax.swing.JButton();
    cmbSelectGenre = new javax.swing.JComboBox<>();
    btnSearchGenre = new javax.swing.JButton();
    jScrollPane1 = new javax.swing.JScrollPane();
    jTable = new javax.swing.JTable();
    lblAddNewItems = new javax.swing.JLabel();
    newItemsInputPanel = new javax.swing.JPanel();
    lblBookID = new javax.swing.JLabel();
    txtBookID = new javax.swing.JTextField();
    lblBookName = new javax.swing.JLabel();
    txtBookName = new javax.swing.JTextField();
    lblAuthor = new javax.swing.JLabel();
    txtAuthor = new javax.swing.JTextField();
    lblBookType = new javax.swing.JLabel();
```

```
rbtnHardCover = new javax.swing.JRadioButton();

rbtnPaperback = new javax.swing.JRadioButton();

lblGenre = new javax.swing.JLabel();

cmbGenre = new javax.swing.JComboBox<>();

lblPublication = new javax.swing.JLabel();

txtPublication = new javax.swing.JTextField();

lblPrice = new javax.swing.JLabel();

txtPrice = new javax.swing.JTextField();

btnAdd = new javax.swing.JButton();

btnClear = new javax.swing.JButton();

lblLogo = new javax.swing.JLabel();

separator = new javax.swing.JSeparator();

menuBar = new javax.swing.JMenuBar();

fileMenu = new javax.swing.JMenu();

fileMenuOpen = new javax.swing.JMenuItem();

fileMenuExit = new javax.swing.JMenuItem();

editMenu = new javax.swing.JMenu();

editMenuDeleteRow = new javax.swing.JMenuItem();

editMenuClearTable = new javax.swing.JMenuItem();

helpMenu = new javax.swing.JMenu();

helpMenuManual = new javax.swing.JMenuItem();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

setLocation(new java.awt.Point(450, 100));
```

```
        setResizable(false);

        mainPanel.setBackground(new java.awt.Color(44, 62, 80));

        txtSearch.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtSearchActionPerformed(evt);
            }
        });

        btnSearchPrice.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
        btnSearchPrice.setText("Search by Price");
        btnSearchPrice.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnSearchPriceActionPerformed(evt);
            }
        });

        cmbSelectGenre.setModel(new javax.swing.DefaultComboBoxModel<>(new
String[] { "Select Genre", "Horror", "Fantasy", "Drama", "Thriller" }));
        cmbSelectGenre.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                cmbSelectGenreActionPerformed(evt);
            }
        });
        btnSearchGenre.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
        btnSearchGenre.setText("Search by Genre");
        btnSearchGenre.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnSearchGenreActionPerformed(evt);
            }
        });

        jTable.setBackground(new java.awt.Color(44, 62, 80));
        jTable.setForeground(new java.awt.Color(255, 255, 255));
        jTable.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {


            },
            new String [] {
                "Book ID", "Book Name", "Author", "Book Type", "Genre", "Publication",
"Price"
            }
        ) {
            boolean[] canEdit = new boolean [] {
                false, false, false, false, false, false, false
            };

            public boolean isCellEditable(int rowIndex, int columnIndex) {
```

```
        return canEdit [columnIndex];
    }
});
jScrollPane1.setViewportView(jTable);
if (jTable.getColumnModel().getColumnCount() > 0) {
    jTable.getColumnModel().getColumn(0).setResizable(false);
    jTable.getColumnModel().getColumn(0).setPreferredWidth(1);
    jTable.getColumnModel().getColumn(1).setResizable(false);
    jTable.getColumnModel().getColumn(1).setPreferredWidth(50);
    jTable.getColumnModel().getColumn(2).setResizable(false);
    jTable.getColumnModel().getColumn(2).setPreferredWidth(20);
    jTable.getColumnModel().getColumn(3).setResizable(false);
    jTable.getColumnModel().getColumn(3).setPreferredWidth(10);
    jTable.getColumnModel().getColumn(4).setResizable(false);
    jTable.getColumnModel().getColumn(4).setPreferredWidth(10);
    jTable.getColumnModel().getColumn(5).setResizable(false);
    jTable.getColumnModel().getColumn(5).setPreferredWidth(15);
    jTable.getColumnModel().getColumn(6).setResizable(false);
    jTable.getColumnModel().getColumn(6).setPreferredWidth(1);
}

lblAddNewItems.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N
lblAddNewItems.setForeground(new java.awt.Color(248, 148, 6));
lblAddNewItems.setText("Add New Items");
```

```
        newItemsInputPanel.setBackground(new java.awt.Color(44, 62, 80));

newItemsInputPanel.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(248, 148, 6)));

        lblBookID.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
        lblBookID.setForeground(new java.awt.Color(255, 255, 255));
        lblBookID.setText("Book ID");

        lblBookName.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
        lblBookName.setForeground(new java.awt.Color(255, 255, 255));
        lblBookName.setText("Book Name");

        lblAuthor.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
        lblAuthor.setForeground(new java.awt.Color(255, 255, 255));
        lblAuthor.setText("Author");

        lblBookType.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
        lblBookType.setForeground(new java.awt.Color(255, 255, 255));
        lblBookType.setText("Book Type");

        btnGrpType.add(rbtnHardCover);
        rbtnHardCover.setForeground(new java.awt.Color(255, 255, 255));
        rbtnHardCover.setText("Hardcover");
        btnGrpType.add(rbtnPaperback);
        rbtnPaperback.setForeground(new java.awt.Color(255, 255, 255));
        rbtnPaperback.setText("Paperback");

        lblGenre.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
        lblGenre.setForeground(new java.awt.Color(255, 255, 255));
        lblGenre.setText("Genre");

        cmbGenre.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] {
"Select", "Horror", "Fantasy", "Drama", "Thriller" }));

        lblPublication.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
        lblPublication.setForeground(new java.awt.Color(255, 255, 255));
        lblPublication.setText("Publication");

        lblPrice.setFont(new java.awt.Font("Tahoma", 0, 12)); // NOI18N
        lblPrice.setForeground(new java.awt.Color(255, 255, 255));
        lblPrice.setText("Price");

        javax.swing.GroupLayout newItemsInputPanelLayout = new
javax.swing.GroupLayout(newItemsInputPanel);
        newItemsInputPanel.setLayout(newItemsInputPanelLayout);
        newItemsInputPanelLayout.setHorizontalGroup(
newItemsInputPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
            .addGroup(newItemsInputPanelLayout.createSequentialGroup()
                .addContainerGap()

.addGroup(newItemsInputPanelLayout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.LEADING)
                    .addComponent(lblBookID,
javax.swing.GroupLayout.PREFERRED_SIZE,                                   70,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblBookName,
javax.swing.GroupLayout.PREFERRED_SIZE,                                   70,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblAuthor,
javax.swing.GroupLayout.PREFERRED_SIZE,                                   70,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblBookType,
javax.swing.GroupLayout.PREFERRED_SIZE,                                   70,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblGenre,
javax.swing.GroupLayout.PREFERRED_SIZE,                                   70,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblPublication,
javax.swing.GroupLayout.PREFERRED_SIZE,                                   70,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lblPrice, javax.swing.GroupLayout.PREFERRED_SIZE,
70, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(18, 18, 18)
```

```
.addGroup(newItemsInputPanelLayout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.LEADING, false)
                    .addGroup(newItemsInputPanelLayout.createSequentialGroup()
                        .addComponent(rbtnHardCover)
                        .addGap(18, 18, 18)
                        .addComponent(rbtnPaperback))
                    .addComponent(txtBookID,
javax.swing.GroupLayout.PREFERRED_SIZE,                                   141,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(cmbGenre,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtPrice, javax.swing.GroupLayout.PREFERRED_SIZE,
150, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtBookName,
javax.swing.GroupLayout.DEFAULT_SIZE, 323, Short.MAX_VALUE)
                    .addComponent(txtAuthor)
                    .addComponent(txtPublication))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        newItemsInputPanelLayout.setVerticalGroup(

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

.addGroup(newItemsInputPanelLayout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.BASELINE)
               .addComponent(txtBookID,
javax.swing.GroupLayout.PREFERRED_SIZE,                                 26,
javax.swing.GroupLayout.PREFERRED_SIZE)
               .addComponent(lblBookID,   javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

.addGroup(newItemsInputPanelLayout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.LEADING)
               .addGroup(newItemsInputPanelLayout.createSequentialGroup()
                  .addGap(19, 19, 19)
                  .addComponent(lblBookName,
javax.swing.GroupLayout.PREFERRED_SIZE,                                 26,
javax.swing.GroupLayout.PREFERRED_SIZE))
               .addGroup(newItemsInputPanelLayout.createSequentialGroup()
                  .addGap(18, 18, 18)
                  .addComponent(txtBookName,
javax.swing.GroupLayout.PREFERRED_SIZE,                                 27,
javax.swing.GroupLayout.PREFERRED_SIZE)))
            .addGap(18, 18, 18)

.addGroup(newItemsInputPanelLayout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.LEADING, false)
               .addComponent(lblAuthor,    javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
```

```
                .addComponent(txtAuthor,    javax.swing.GroupLayout.DEFAULT_SIZE,
26, Short.MAX_VALUE))
             .addGap(18, 18, 18)

.addGroup(newItemsInputPanelLayout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.BASELINE)
                .addComponent(lblBookType,
javax.swing.GroupLayout.PREFERRED_SIZE,                                  26,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(rbtnHardCover)
                .addComponent(rbtnPaperback))
             .addGap(18, 18, 18)

.addGroup(newItemsInputPanelLayout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.BASELINE)
                .addComponent(lblGenre,
javax.swing.GroupLayout.PREFERRED_SIZE,                                  26,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(cmbGenre,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
             .addGap(18, 18, 18)

.addGroup(newItemsInputPanelLayout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.TRAILING)
                .addComponent(lblPublication,
javax.swing.GroupLayout.PREFERRED_SIZE,                                  26,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                .addComponent(txtPublication,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    26,
javax.swing.GroupLayout.PREFERRED_SIZE))

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

.addGroup(newItemsInputPanelLayout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.LEADING)
                .addComponent(lblPrice, javax.swing.GroupLayout.PREFERRED_SIZE,
26, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(txtPrice, javax.swing.GroupLayout.PREFERRED_SIZE,
26, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addContainerGap())
        );

        btnAdd.setText("Add Item");
        btnAdd.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnAddActionPerformed(evt);
            }
        });

        btnClear.setText("Clear");
        btnClear.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnClearActionPerformed(evt);
            }
        });

        lblLogo.setFont(new java.awt.Font("Mistral", 0, 43)); // NOI18N
        lblLogo.setForeground(new java.awt.Color(248, 148, 6));
        lblLogo.setText("BookMark");

        separator.setBackground(new java.awt.Color(248, 148, 6));
        separator.setForeground(new java.awt.Color(248, 148, 6));

        javax.swing.GroupLayout mainPanelLayout = new
javax.swing.GroupLayout(mainPanel);
        mainPanel.setLayout(mainPanelLayout);
        mainPanelLayout.setHorizontalGroup(

mainPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
            .addGroup(mainPanelLayout.createSequentialGroup()

.addGroup(mainPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
                    .addGroup(mainPanelLayout.createSequentialGroup()
                        .addGap(423, 423, 423)
                        .addComponent(btnAdd,
javax.swing.GroupLayout.PREFERRED_SIZE, 102,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                    .addGap(18, 18, 18)
                    .addComponent(btnClear,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    102,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(mainPanelLayout.createSequentialGroup()
                    .addGap(6, 6, 6)

.addGroup(mainPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING, false)
                        .addGroup(mainPanelLayout.createSequentialGroup()
                            .addComponent(txtSearch,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    121,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(btnSearchPrice,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    119,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGroup(mainPanelLayout.createSequentialGroup()
                            .addComponent(cmbSelectGenre,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    121,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(btnSearchGenre,
javax.swing.GroupLayout.PREFERRED_SIZE, 1, Short.MAX_VALUE)))))
                .addGap(0, 369, Short.MAX_VALUE))
            .addGroup(mainPanelLayout.createSequentialGroup()
            .addContainerGap()

.addGroup(mainPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.LEADING)
               .addComponent(jScrollPane1)
               .addComponent(newItemsInputPanel,
javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
               .addGroup(mainPanelLayout.createSequentialGroup()
                  .addComponent(lblAddNewItems,
javax.swing.GroupLayout.PREFERRED_SIZE,                                  143,
javax.swing.GroupLayout.PREFERRED_SIZE)
                  .addGap(0, 0, Short.MAX_VALUE))
               .addComponent(separator,
javax.swing.GroupLayout.Alignment.TRAILING))
            .addContainerGap())
         .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
mainPanelLayout.createSequentialGroup()
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
            .addComponent(lblLogo)
            .addGap(422, 422, 422))
      );
      mainPanelLayout.setVerticalGroup(

mainPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
```

```
            .addGroup(mainPanelLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(lblLogo, javax.swing.GroupLayout.PREFERRED_SIZE, 42, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(separator, javax.swing.GroupLayout.PREFERRED_SIZE, 10, javax.swing.GroupLayout.PREFERRED_SIZE)

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

.addGroup(mainPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(txtSearch, javax.swing.GroupLayout.PREFERRED_SIZE, 29, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnSearchPrice, javax.swing.GroupLayout.PREFERRED_SIZE, 29, javax.swing.GroupLayout.PREFERRED_SIZE))

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

.addGroup(mainPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(cmbSelectGenre, javax.swing.GroupLayout.PREFERRED_SIZE, 30, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(btnSearchGenre,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    29,
javax.swing.GroupLayout.PREFERRED_SIZE))
             .addGap(18, 18, 18)
             .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE,                                   220,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
             .addComponent(lblAddNewItems,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    29,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
             .addComponent(newItemsInputPanel,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

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

.addGroup(mainPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignme
nt.BASELINE)
                .addComponent(btnAdd, javax.swing.GroupLayout.PREFERRED_SIZE,
33, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(btnClear,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    33,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    );

    fileMenu.setText("File");

    fileMenuOpen.setText("Open");
    fileMenuOpen.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            fileMenuOpenActionPerformed(evt);
        }
    });
    fileMenu.add(fileMenuOpen);

    fileMenuExit.setText("Exit");
    fileMenuExit.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            fileMenuExitActionPerformed(evt);
        }
    });
    fileMenu.add(fileMenuExit);

    menuBar.add(fileMenu);

    editMenu.setText("Edit");
```

```
editMenuDeleteRow.setText("Delete Selected Row");
editMenuDeleteRow.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        editMenuDeleteRowActionPerformed(evt);
    }
});
editMenu.add(editMenuDeleteRow);

editMenuClearTable.setText("Clear Table");
editMenuClearTable.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        editMenuClearTableActionPerformed(evt);
    }
});
editMenu.add(editMenuClearTable);

menuBar.add(editMenu);

helpMenu.setText("Help");

helpMenuManual.setText("Manual");
helpMenuManual.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
            helpMenuManualActionPerformed(evt);
        }
    });
    helpMenu.add(helpMenuManual);

    menuBar.add(helpMenu);

    setJMenuBar(menuBar);

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
    getContentPane().setLayout(layout);
    layout.setHorizontalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addComponent(mainPanel, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
    );
    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addComponent(mainPanel, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
    );

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

```
private void clearInput(){
    txtBookName.setText("");
    txtBookID.setText("");
    txtPublication.setText("");
    txtPrice.setText("");
    txtAuthor.setText("");
    txtSearch.setText("");
    cmbSelectGenre.setSelectedIndex(0);
    btnGrpType.clearSelection();
    cmbGenre.setSelectedIndex(0);
}

private DefaultTableModel tableModel(){
    DefaultTableModel model = (DefaultTableModel)jTable.getModel();
    return model;
}

private void fileMenuOpenActionPerformed(java.awt.event.ActionEvent evt) {
    File selectedFile = null;
    JFileChooser fileChooser = new JFileChooser();
    fileChooser.setCurrentDirectory(new File(".//src//"));
    int result = fileChooser.showOpenDialog(this);
    selectedFile = fileChooser.getSelectedFile();
    if (selectedFile!=null) {

      try{

        BufferedReader br = new BufferedReader(new FileReader (selectedFile));


        Object[] tableLines = br.lines().toArray();

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


          String line = tableLines[i].toString().trim();

          String[] dataRow = line.split(",");




          int BookID = Integer.parseInt(dataRow[0]);

          String BookName = dataRow[1];

          String Author = dataRow[2];

          String BookType = dataRow[3];

          String Genre = dataRow[4];

          String Publication = dataRow [5];

          double Price = Double.parseDouble(dataRow[6]);

          LL.add(new
DataModel(BookID,BookName,Author,BookType,Genre,Publication,Price));

          Object[]
rowData={BookID,BookName,Author,BookType,Genre,Publication,Price};

          tableModel().addRow(rowData);

        }

      } catch (Exception ex){
```

```
                JOptionPane.showMessageDialog(rootPane,"An Error Occured While Selecting the File!","Error",JOptionPane.ERROR_MESSAGE);
            }
        }else{
            JOptionPane.showMessageDialog(rootPane,"File Not Selected.");
        }
    }

    private void fileMenuExitActionPerformed(java.awt.event.ActionEvent evt) {
        System.exit(0);
    }

    private void helpMenuManualActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            File file1 = new File(".//src//user_manual.pdf");
            Desktop.getDesktop().open(file1);
        } catch (IOException ex) {
            JOptionPane.showMessageDialog(rootPane,"The program cannot locate the help file.","Error",JOptionPane.ERROR_MESSAGE);
        }
    }

    private void editMenuDeleteRowActionPerformed(java.awt.event.ActionEvent evt) {
        if(jTable.getSelectedRow()!=-1){
            int bookID=(Integer)jTable.getValueAt(jTable.getSelectedRow(), 0);
            for(int i=0;i<=LL.size()-1;i++){
                if(LL.get(i).getBookID()==bookID){
                    LL.remove(i);
                }
            }
            tableModel().removeRow(jTable.getSelectedRow());
            JOptionPane.showMessageDialog(rootPane, "Selected row deleted successfully");
        }else{
            JOptionPane.showMessageDialog(rootPane,"Please Select a Row First!","Error",JOptionPane.ERROR_MESSAGE);
        }
    }

    private void editMenuClearTableActionPerformed(java.awt.event.ActionEvent evt) {
        tableModel().setRowCount(0);
        LL.clear();
    }

    private void btnClearActionPerformed(java.awt.event.ActionEvent evt) {
        clearInput();
    }

    private void btnAddActionPerformed(java.awt.event.ActionEvent evt) {
        rbtnHardCover.setActionCommand("Hard Cover");
        rbtnPaperback.setActionCommand("Paperback");
        if          (txtPublication.getText().equals("")||cmbGenre.getSelectedIndex()==0||
txtBookID.getText().equals("") ||
            txtBookName.getText().equals("")||
txtAuthor.getText().equals("")||txtPrice.getText().equals("")||btnGrpType.getSelection(
)==null){
            JOptionPane.showMessageDialog(rootPane,"Invalid
Input!","Error",JOptionPane.ERROR_MESSAGE);
        }else{
            try{
                int bookID= Integer.parseInt(txtBookID.getText());
                double price= Double.parseDouble(txtPrice.getText());
                String bookName=txtBookName.getText();
                String author=txtAuthor.getText();
                String genre=(String)cmbGenre.getSelectedItem();
                String publication=txtPublication.getText();
                String bookType="";

                if (rbtnHardCover.isSelected()){
                    bookType= rbtnHardCover.getActionCommand();
                }else if (rbtnPaperback.isSelected()){
                    bookType= rbtnPaperback.getActionCommand();
                }

                DataModel                                   rowData=new
DataModel(bookID,bookName,author,bookType,genre,publication,price);
```

```
            Object[] objDataList =
{bookID,bookName,author,bookType,genre,publication,price};


            LL.add(rowData);
            tableModel().addRow(objDataList);


        }catch(NumberFormatException e){
            JOptionPane.showMessageDialog(rootPane,"Invalid
Input!","Error",JOptionPane.ERROR_MESSAGE);
        }
    }


}


private void btnSearchGenreActionPerformed(java.awt.event.ActionEvent evt) {
    if (cmbSelectGenre.getSelectedIndex()==0){
        JOptionPane.showMessageDialog(rootPane,"Please Select A Genre
First!","Error",JOptionPane.ERROR_MESSAGE);
    }else{
        String Genre=(String)cmbSelectGenre.getSelectedItem();
        LinkedList<DataModel>LL2 = new Algorithms().LinearSearch(LL,Genre);
        int quantity=LL2.size();
        String books="|";
        for(int i=0;i<=LL2.size()-1;i++){
            books=books+LL2.get(i).getBookName()+" | ";
        }
        JOptionPane.showMessageDialog(rootPane, "The "+Genre+" Genre has "+quantity+" books: "+books);
    }
}


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


}


private void btnSearchPriceActionPerformed(java.awt.event.ActionEvent evt) {
    if(txtSearch.getText().isEmpty()){
        JOptionPane.showMessageDialog(rootPane,"Please Enter a Price for Searching.","Error",JOptionPane.ERROR_MESSAGE);
    }else{
        try{
            double searchedPrice=Double.parseDouble(txtSearch.getText());
            DataModel a = new Algorithms().BinarySearch(new Algorithms().SelectionSort(LL),0,LL.size()-1,searchedPrice);
            if(a==null){
                JOptionPane.showMessageDialog(rootPane, "No Macth Found.");
            }else{
                String bookID = "Book Id: "+a.getBookID()+" | ";
                String bookName = "Book Name: "+a.getBookName()+" | ";
                String author = "Author: "+a.getAuthor()+" | ";
                String genre = "Genre: "+a.getGenre()+" | ";
                String bookType = "Book Type: "+a.getBookType()+" | ";
                String publication ="Publication: "+a.getPublication()+" | ";
                String price = "Price: "+a.getPrice();
                JOptionPane.showMessageDialog(rootPane, " Matched Item:\n "+bookID+bookName+author+genre+bookType+publication+price);
            }
        }catch(NumberFormatException e){
            JOptionPane.showMessageDialog(rootPane,"Invalid Input!","Error",JOptionPane.ERROR_MESSAGE);
        }catch(StackOverflowError e){
            JOptionPane.showMessageDialog(rootPane, "No Macth Found.");
        }catch(NoSuchElementException e){
            JOptionPane.showMessageDialog(rootPane,"The Table is Empty!","Error",JOptionPane.ERROR_MESSAGE);
        }
    }
}

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

}

public static void main(String args[]) {
    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(App.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {
java.util.logging.Logger.getLogger(App.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
java.util.logging.Logger.getLogger(App.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
java.util.logging.Logger.getLogger(App.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new App().setVisible(true);
            }
        });
    }
    private javax.swing.JButton btnAdd;
    private javax.swing.JButton btnClear;
    private javax.swing.ButtonGroup btnGrpType;
    private javax.swing.JButton btnSearchGenre;
    private javax.swing.JButton btnSearchPrice;
    private javax.swing.JComboBox<String> cmbGenre;
    private javax.swing.JComboBox<String> cmbSelectGenre;
    private javax.swing.JMenu editMenu;
    private javax.swing.JMenuItem editMenuClearTable;
    private javax.swing.JMenuItem editMenuDeleteRow;
    private javax.swing.JMenu fileMenu;
    private javax.swing.JMenuItem fileMenuExit;
    private javax.swing.JMenuItem fileMenuOpen;
    private javax.swing.JMenu helpMenu;
    private javax.swing.JMenuItem helpMenuManual;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTable jTable;
    private javax.swing.JLabel lblAddNewItems;
    private javax.swing.JLabel lblAuthor;
    private javax.swing.JLabel lblBookID;
    private javax.swing.JLabel lblBookName;
    private javax.swing.JLabel lblBookType;

    private javax.swing.JLabel lblGenre;

    private javax.swing.JLabel lblLogo;

    private javax.swing.JLabel lblPrice;

    private javax.swing.JLabel lblPublication;

    private javax.swing.JPanel mainPanel;

    private javax.swing.JMenuBar menuBar;

    private javax.swing.JPanel newItemsInputPanel;

    private javax.swing.JRadioButton rbtnHardCover;

    private javax.swing.JRadioButton rbtnPaperback;

    private javax.swing.JSeparator separator;

    private javax.swing.JTextField txtAuthor;

    private javax.swing.JTextField txtBookID;

    private javax.swing.JTextField txtBookName;

    private javax.swing.JTextField txtPrice;

    private javax.swing.JTextField txtPublication;

    private javax.swing.JTextField txtSearch;

}
```

2.14.2. Class: DataModel

```
public class DataModel {



    int bookID;

    String bookName,author,bookType,genre,publication;

    double price;
```

```
DataModel(int bookID,String bookName,String author,String bookType,String genre,String publication, double price){
    this.bookID=bookID;
    this.bookName=bookName;
    this.author=author;
    this.bookType=bookType;
    this.genre=genre;
    this.publication=publication;
    this.price=price;
}

int getBookID(){
    return bookID;
}

String getBookName(){
    return bookName;
}

String getAuthor(){
    return author;
}

String getBookType(){
    return bookType;
    }

    String getGenre(){
        return genre;
    }

    String getPublication(){
        return publication;
    }

    double getPrice(){
        return price;
    }
}
```

2.14.3. Class: Algorithms

```
import java.util.LinkedList;

public class Algorithms {

    LinkedList<DataModel> SelectionSort(LinkedList<DataModel> LL){
         int n = LL.size();
        for (int i = 0; i < n-1; i++)
        {
            int min_idx = i;
            for (int j = i+1; j < n; j++)
```

```
                if ((Double)LL.get(j).getPrice() < (Double)LL.get(min_idx).getPrice()){
                    min_idx = j;
                }
            DataModel temp = LL.get(min_idx);
            LL.set(min_idx, LL.get(i));
            LL.set(i,temp);
        }
        return LL;
    }

    DataModel BinarySearch (LinkedList<DataModel> LL,int low,int high,double price){
        if (high>0)
        {
            int mid = low+((high-low)/2);
            if (LL.get(mid).getPrice() == price) {
               return LL.get(mid);
            }else if (LL.get(mid).getPrice() > price){
               return BinarySearch(LL, low, mid-1, price);
            }else{
                return BinarySearch(LL, mid+1, high, price);
            }
        } else{
            return null;
        }
    }


    LinkedList<DataModel> LinearSearch(LinkedList<DataModel> LL,String genre){
        LinkedList<DataModel> LL2=new LinkedList<>();
        for (int i=0;i<=LL.size()-1;i++){
            if (LL.get(i).getGenre().equals(genre)){
                LL2.add(LL.get(i));
            }
        }
        return LL2;
    }
}
```