



Islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CS5004NI Emerging Programming Platforms and Technologies

Assessment Weightage & Type

30% Group Coursework

Year and Semester

2019-20 Autumn / 2020-21 Spring

Group Name:

SN	Student Name (Section name)	College ID	University ID
1	Aashna Shrestha (L2C13)	NP01CP4S210103	20048800
2	Rabina Shrestha (L2C13)	NP01CP4S210039	20049416
3	Subriti Aryal (L2C13)	NP01CP4S210044	20049062

Assignment Due Date: 10th January 2022

Assignment Submission Date: 10th January 2022

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Contents

Proposal.....	1
Individual Tasks	7
Introduction.	8
Binary Search.....	9
Merge Sort.	11
Method Descriptions.....	16
Testing.	24
Test 1: To test if the program can run in NetBeans.	24
Test 2: To add item details in the table.....	25
Test 3: To search for an instrument based on price.....	29
Test 4: To search for the number of Instruments available in a category.	30
Test 5: To open a file from menu.....	31
Test 6: System Validation.	36
Conclusion	42
Bibliography.	43
Appendix	44

List of Figures

Figure 1: Main UI for the system	4
Figure 2: UI for Login.....	4
Figure 3: UI for Registration	5
Figure 4: UI for Admin to input instrument details	5
Figure 5: UI for Members to view instrument details	6
Figure 6: Flowchart of Binary Search.	10
Figure 7: Merge Sort Part 1.....	12
Figure 8: Merge Sort Part 2.....	13
Figure 9: Merge Sort Part 3.....	14
Figure 10: Merge Sort Part 4.....	15
Figure 11: Program can run in NetBeans: Test Successful.....	24
Figure 12: Adding Instruments.	26
Figure 13: Instruments has been added.....	26
Figure 14: Add Instruments: Test Successful.....	27
Figure 15: Registration.....	27
Figure 16: Registration Complete.....	28
Figure 17: Registration: Test Successful.....	28
Figure 18: Search an Instrument using Price: Test Successful.....	29
Figure 19: Number of Instruments Available in a Category: Test Successful.....	30
Figure 20: Open: Import Instruments.....	32
Figure 21: Open: Import Instruments: Successful.	32
Figure 22: Open: External Files.....	33
Figure 23: Open: External Files: Choosing the File.	33
Figure 24: Open: External Files: Test Successful.	34
Figure 25: Help: User Manual.	34
Figure 26: Help: Open User Manual: Test Successful.....	35
Figure 27: Admin: Wrong Username: Test Successful.....	37
Figure 28: Admin: Incorrect Password: Text Successful.	37
Figure 29: Admin: Empty Fields: Test Successful.	38
Figure 30: Admin: Price Validation: Test Successful.....	38

Figure 31: Member: Wrong Username: Test Successful.....	39
Figure 32: Member: Invalid Password: Test Successful.....	39
Figure 33: Member: Search Validation: Test Successful.....	40
Figure 34: Registration: Empty Fields: Test Successful.	40
Figure 35: Registration: Age Validation: Test Successful.....	41
Figure 36: Registration: Password Validation: Test Successful.	41
Figure 37: Home Page.	45
Figure 38: Admin Button.....	46
Figure 39: Admin Login Details.	46
Figure 40: Admin: Add Instruments.	46
Figure 41: Admin: Instrument Added.....	47
Figure 42: Member Button.....	48
Figure 43: Member Login Details.	48
Figure 44: Member: Instruments.	48
Figure 45: Member: Instruments Part 2.....	49
Figure 46: Member: Instruments: Sort By: Category.	50
Figure 47: Member: Instruments: Specific Category Search.	50
Figure 48: Member: Instruments: Sort By: Price.	51
Figure 49: Member: Instruments: Specific Price Search.	51
Figure 50: Register Button.	52
Figure 51: Registration.	52
Figure 52: Registration Complete.....	53
Figure 53: Open External Files.....	54
Figure 54: Opening External Files.	54

List of Tables

Table 1: Group Members.	1
Table 2: GUI Components.....	2
Table 3: Individual Tasks.....	7
Table 4: Method Description for GUI.java	22
Table 5: Method Description for MusicalInstrument.java.....	22
Table 6: Method Description for MergeSort.java	23
Table 7: Method Description for BinarySearch.java	23
Table 8: To test if the program can run in NetBeans.	24
Table 9: To add item details in the table.....	25
Table 10: To search for an instrument based on price.	29
Table 11: To search the number of Instruments available in a Category.	30
Table 12: To open a file from menu.....	31
Table 13: To open a file from menu.....	36

Proposal.

Propose title of the system.

The proposed title for the project is “Symphony Academy of Music” Information System.

Brief information about your system.

The proposal is about a Java application that manages the information system of a Music Academy. The information system will allow users to register as a member and search for instruments. It allows the admin to login with their credentials to input the details of new instruments into the academy. The system permits the customers to view and research the products of their choice beforehand by logging in with their username and a registered password. It will also aid in reducing the crowd in the academy by allowing registration via online forms. All the data generated within the system is stored in a .csv file.

Group Members:

Student Name	College ID	University ID
Aashna Shrestha	NP01CP4S210103	20048800
Rabina Shrestha	NP01CP4S210039	20049416
Subriti Aryal	NP01CP4S210044	20049062

Table 1: Group Members.

List of GUI components and their datatype.

Components	Datatype	Purpose
JFrame	Object	The frame creates a boundary which holds other components within it.
JPanel	Object	The panel divides the frame into sections and groups the components.
JLabel	String	Displays text or icons.
TextField	String/ int	Allows users to input text in the registration form. TextField is also used as a search box for searching the category.
JButton	Object	Allows users to choose various functionalities within the system; whether they want to login or register as a new member. A button will also be available to submit the registration form. Each button is inbuilt with some action event.
JRadioButton	String	JRadioButton will be available as a choice to search the instrument according to price or category.
JComboBox	Array	JComboBox will allow members to select a price range for the display of relative instrument. The options will be stored in an array.
JOptionPane	String	Displays the search result of the instruments or alerts the users when required.
JTable	Object	Displays the data from .csv file and updates the data from the form into the table.

Table 2: GUI Components.

List of features

- The system allows existing members to login or new members to register.
- To register, users would have to fill a form containing their personal details. The form will be composed of JLabels, JTextFields, and JButtons.
- Alert messages and input validations are implemented for the input of appropriate details into the form.
- The login credentials and their personal details would be stored in a .csv file.
- Once a member is logged in, they can search for:
 - instruments available in the academy's store according to the price.
 - instruments available in the academy's store according to the category.
- The admin can login into the system with a username and password.
- The system would enable the admin to add instruments into the academy and an access to a .csv file containing all the details.
- The system will contain a menu bar with the following options:
 - File – It will allow users to open an existing file or exit the system.
 - Help – It will display a pdf containing user guide for the system.

Wireframes

This wireframe resembles the main UI of the Music Academy System. It gives a brief description of the Academy and provides the user with a few options to proceed with; Login as Admin, Login as existing Member or to Register as a new Member.

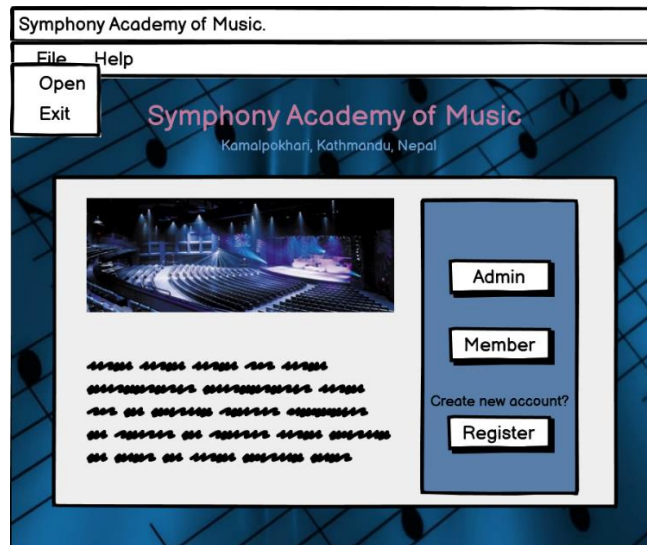


Figure 1: Main UI for the system

This wireframe resembles the Login Portal for the system. The similar panel is used for the Admin Login and Member Login but with valid login credentials, the admin is redirected to the panel to add details of new instrument while the user is provided an access to view the instruments available in the store and sort or search them according to price or category.

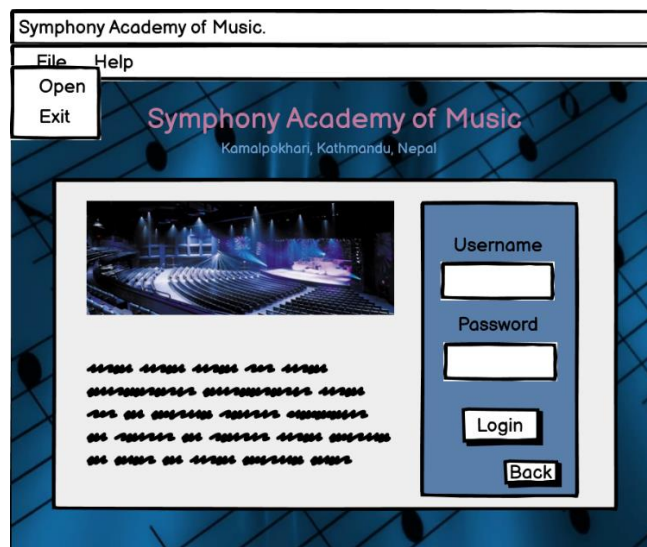


Figure 2: UI for Login

This wireframe resembles the UI for Registration of a new Member. It provides users with a form with some personal details to fill into. The member's info is temporarily stored in the JTable but permanently stored in the .csv file.

Symphony Academy of Music.

File Help

Open
Exit

Symphony Academy of Music
Kamalpokhari, Kathmandu, Nepal

Registration

Name Username

Specialization Password

Age Re-type Password

Name	Specialization	Age	Username	Password

Figure 3: UI for Registration

This wireframe resembles the Instrument addition UI for the Admin's side. It provides Admin with a form with some instrument details to fill into. The instrument's info is temporarily stored in the JTable but permanently stored in the .csv file.

The details stored by the Admin onto the table can be viewed from the Member's side.

Symphony Academy of Music.

File Help

Open
Exit

Add Instruments

Instrument ID Instrument

Model No. Brand

Price Warranty

Instrument ID	Instrument	Model No.	Brand	Price	Warranty

Figure 4: UI for Admin to input instrument details

This wireframe resembles the Instrument viewing UI for the Member's side. It provides members with an option to view and search instrument according to their choice of category or a desired price range. The data displayed into the JTable is imported from the .csv file.

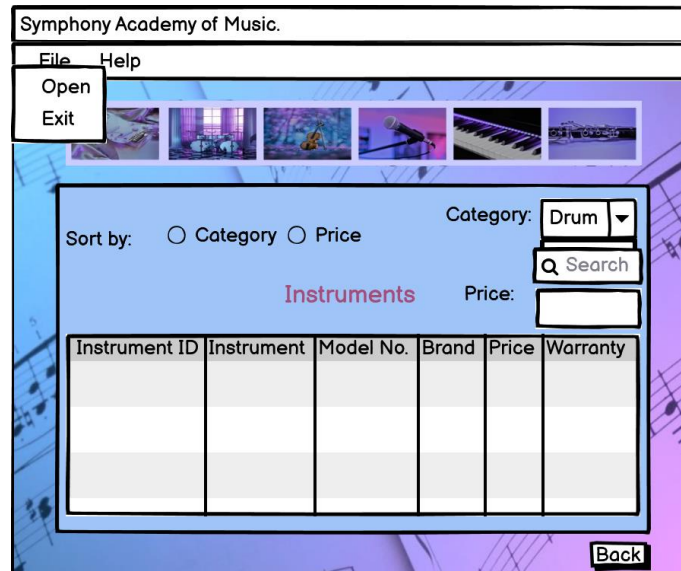


Figure 5: UI for Members to view instrument details

Justification of the tools to be used for the system development.

- Apache Netbeans – The IDE has been used to create the GUI using java swing components and assign respective functionalities.
- MS Word – It has been used to create the documentation of the project.
- Balsamiq – A software used to create wireframes for the system.
- Draw.io – A software used to create flowcharts.

Individual Tasks

Group Member	Responsibilities
Aashna Shrestha	<ul style="list-style-type: none">➤ Worked on the code for Searching and Sorting.➤ Worked on Login Validation.➤ Worked on Flowcharts and Algorithms explanation.
Rabina Shrestha	<ul style="list-style-type: none">➤ Worked on GUI of the system.➤ Worked on validation of inputs.➤ Worked on documentation of the report.
Subriti Aryal	<ul style="list-style-type: none">➤ Worked on Import and Export of data into JTable and .csv file.➤ Worked on GUI of the system.➤ Worked on the documentation of the report.

Table 3: Individual Tasks

Introduction.

One of the modules that the Computing students' study in Year 2 is "Emerging Programming Platforms and Technologies". This coursework was assigned to the students on the 6th Week and it accounts for 30% of the overall module grade. It is a three-student group coursework in which, the work is divided among the members and is presented with a proper documentation. The report is evidence based with proper description and diagrams. It follows a standard format and has a transparent body of work including evaluation and reflection along with the difficulties encountered.

Based on the scenario given, students were needed to create and develop an information system with an acceptable title and a table for storing and displaying the necessary information. The NetBeans IDE has been used to construct a Java-based Information System called Symphony Academy of Music. The users can log in, register, and view the products featured through this application. It also enables the administrator to add new goods. A menu bar with file and help options is also included in the system's features.

The functions and data are linked together using the Java's object-oriented programming concept. File handling has been used to read and write the data into the table and the csv file. To allow users to search for products depending on their category or price, searching algorithms and sorting algorithms has also been included into the program. Various java swing components like frame, panels, labels, text fields, radio buttons, combo boxes, tables, and many more are used to make the user interface more friendly.

Binary Search.

Elements are searched in a list using certain searching techniques. Binary Search is one such popular techniques to search the required element in a given list.

To perform binary search, it must be ensured that the list to be searched is sorted which means it must be present in either ascending order or descending order. The sorted list is divided in half and the element to be searched is compared with the element in the middle. If the values match, the index of the element is returned. If the middle element is greater than the search element, then the search is performed in the left half of the list. This is done by shifting the index of the rightmost element to the index exactly less than middle element's index. Instead, if the middle element is smaller, then the search is performed in the right half of the list. This is done by shifting the index of the leftmost element to the index exactly greater than middle element's index. (javaTpoint, 2021)

In this coursework, binary search has been performed to search the instrument having the price entered by the members in the jTextField. The price input by the members is the search element. An arraylist has been initialized to read the data from the file Instrument.csv which has the details of the instruments. The arraylist has then been sorted according to the price using merge sort algorithm. The index of the middle element is calculated by dividing the sum of index of the leftmost and the rightmost element by two. The middle element is compared to the search element. If the values are identical, the object of that element is returned. If the price entered is less than the middle element, the search is done in the left half of the arraylist otherwise the search is done in the right half of the arraylist. This process keeps on repeating until the required price is found. Once, the instrument with the entered price is found, its details are displayed in a JOptionPane to the member. If there are no instruments having the price entered by the member, they will be notified with an alert.

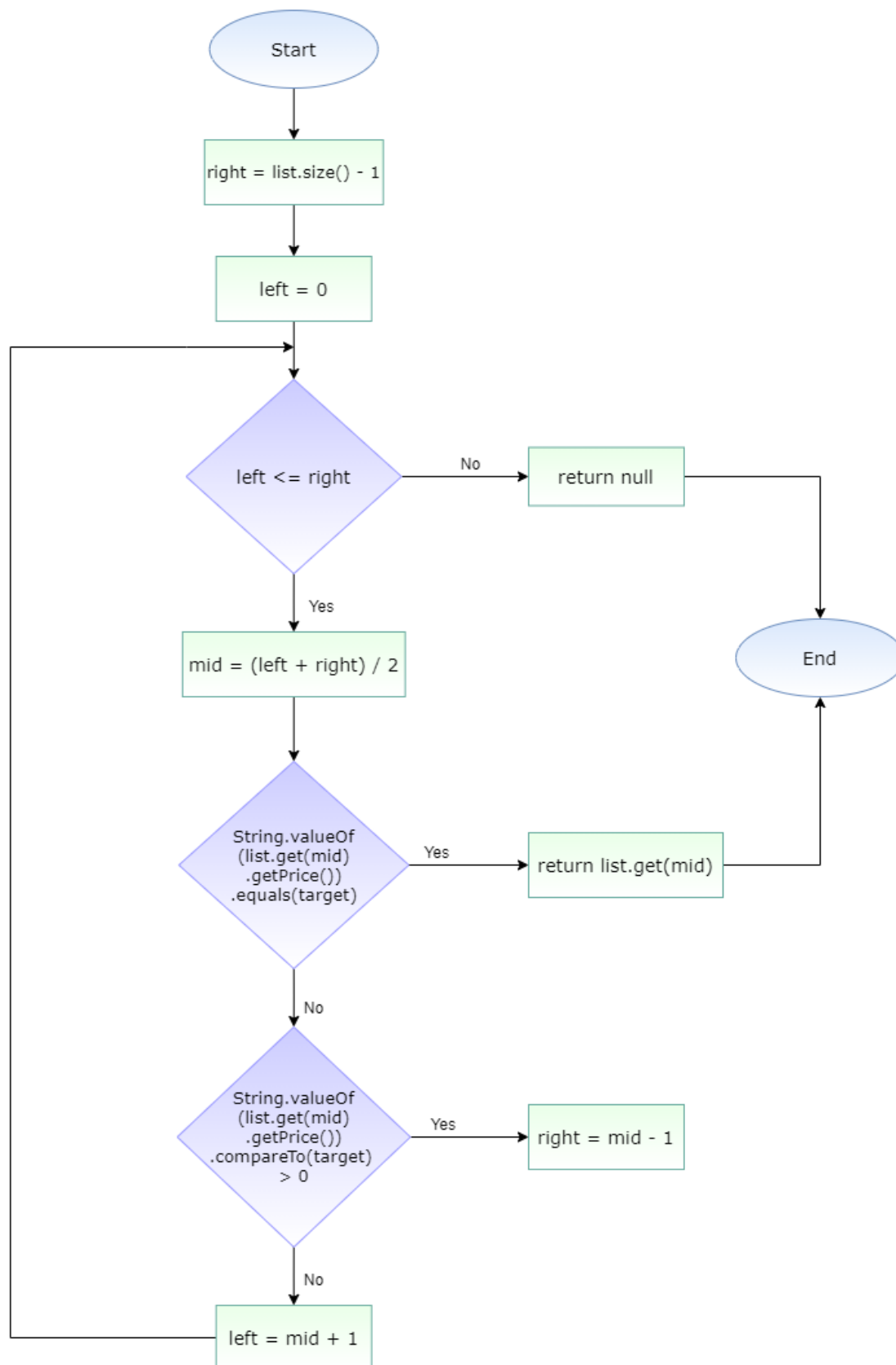


Figure 6: Flowchart of Binary Search.

Merge Sort.

There are various algorithms used for sorting lists. Merge Sort is an efficient way of arranging any lists in a particular order.

To perform merge sort, the unsorted list is divided into two halves. The function to implement merge sort calls itself for the divided halves and further division is done until the list cannot be broken down. The elements are then compared and merged in the same order that it had been broken down. Finally, a sorted list is returned. (javaTpoint, 2021)

In this coursework, merge sort has been performed in order to sort an arraylist with the details of instruments according to the order required by the user. The arraylist is broken down into two more arraylists: left and right. These arraylists are again broken down recursively until they cannot be separated. The elements are then compared. If the member wants to sort the instruments by category, the instrument names are compared and merged in alphabetical order. Instead, if the member wants to sort the instruments by price, the price of each instrument will be compared and sorted in ascending order. Each element is compared and merged step by step according to the order that they were broken down. Once the list has been sorted, it is returned and displayed in the JTable to be viewed by the user.

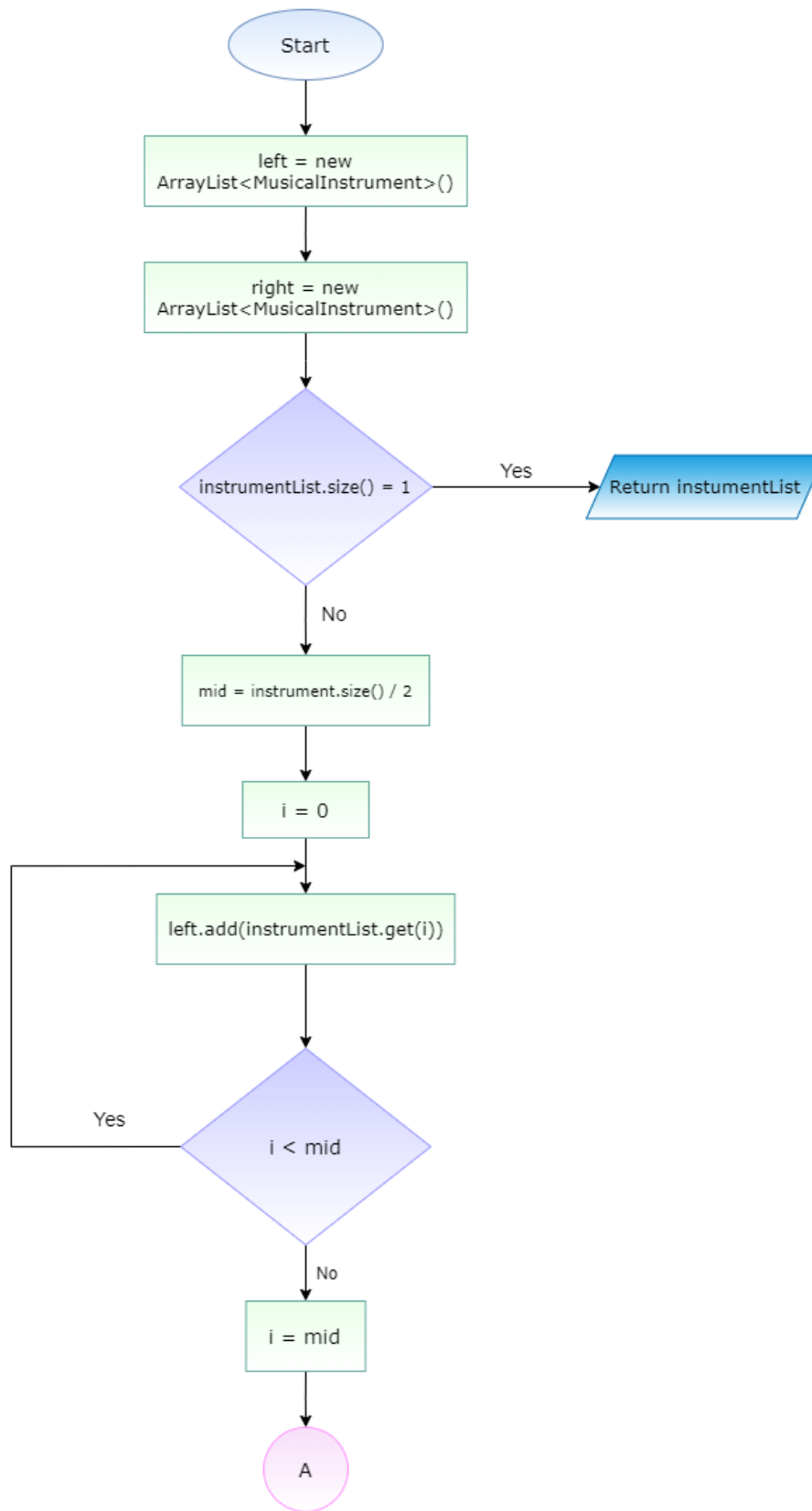


Figure 7: Merge Sort Part 1.

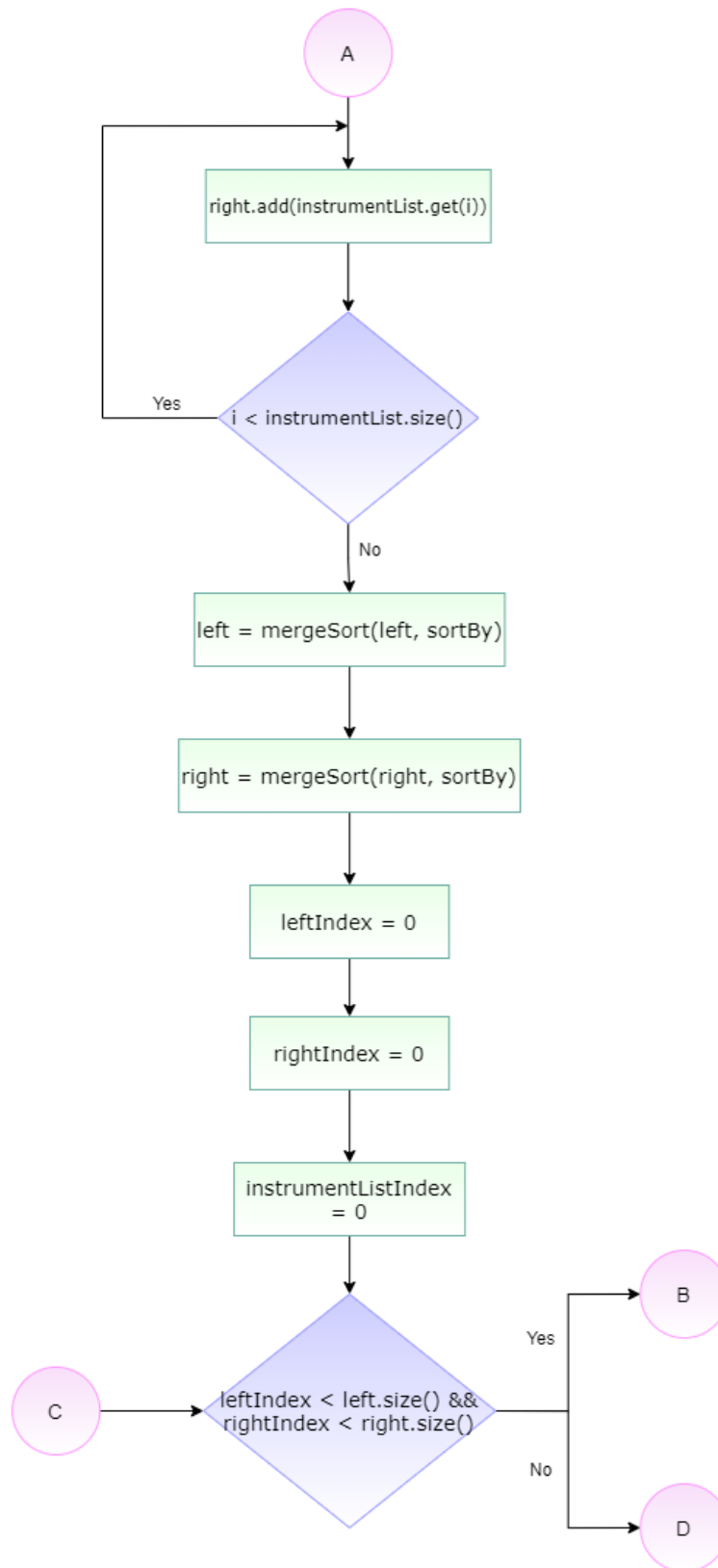
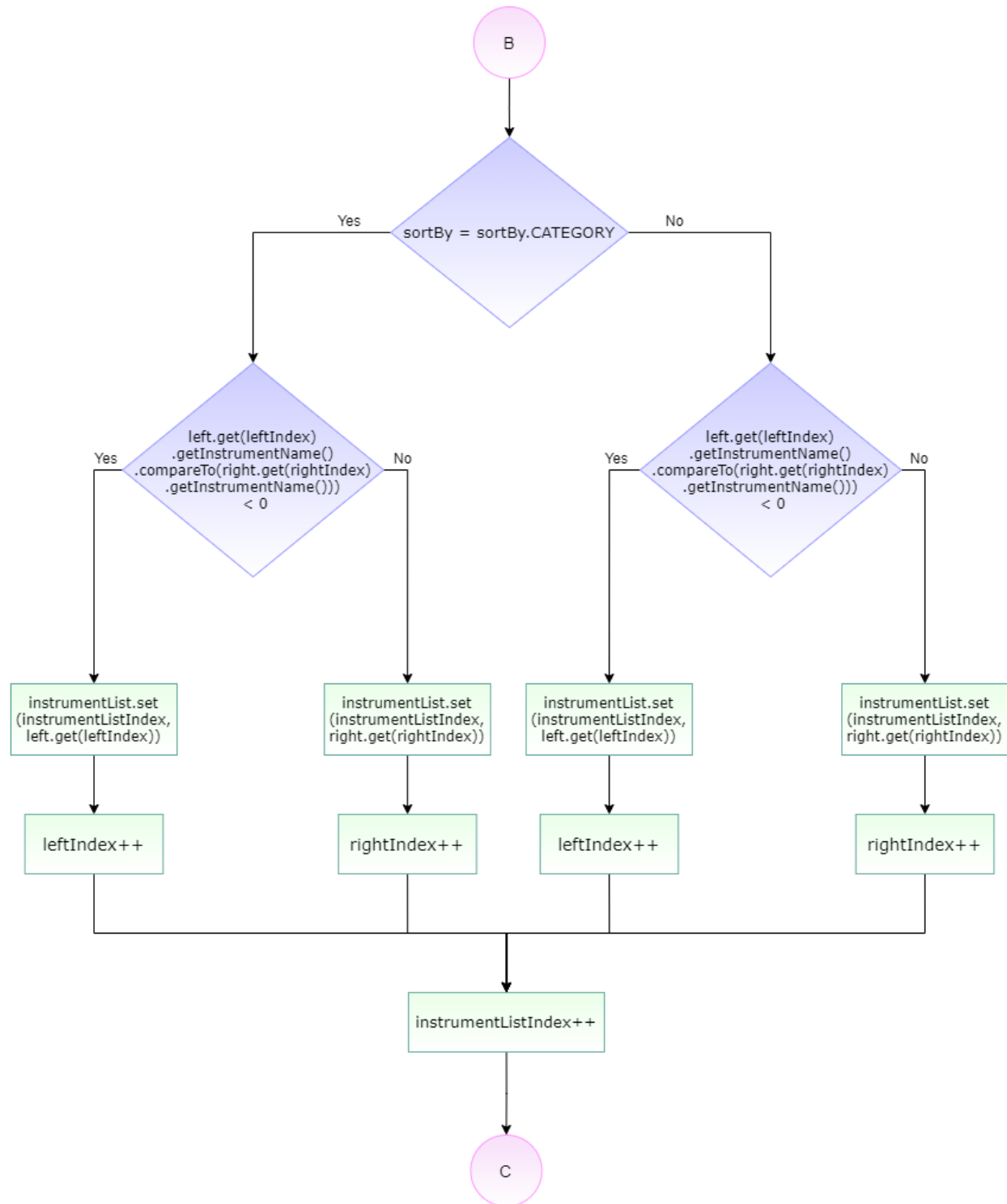


Figure 8: Merge Sort Part 2.

*Figure 9: Merge Sort Part 3.*

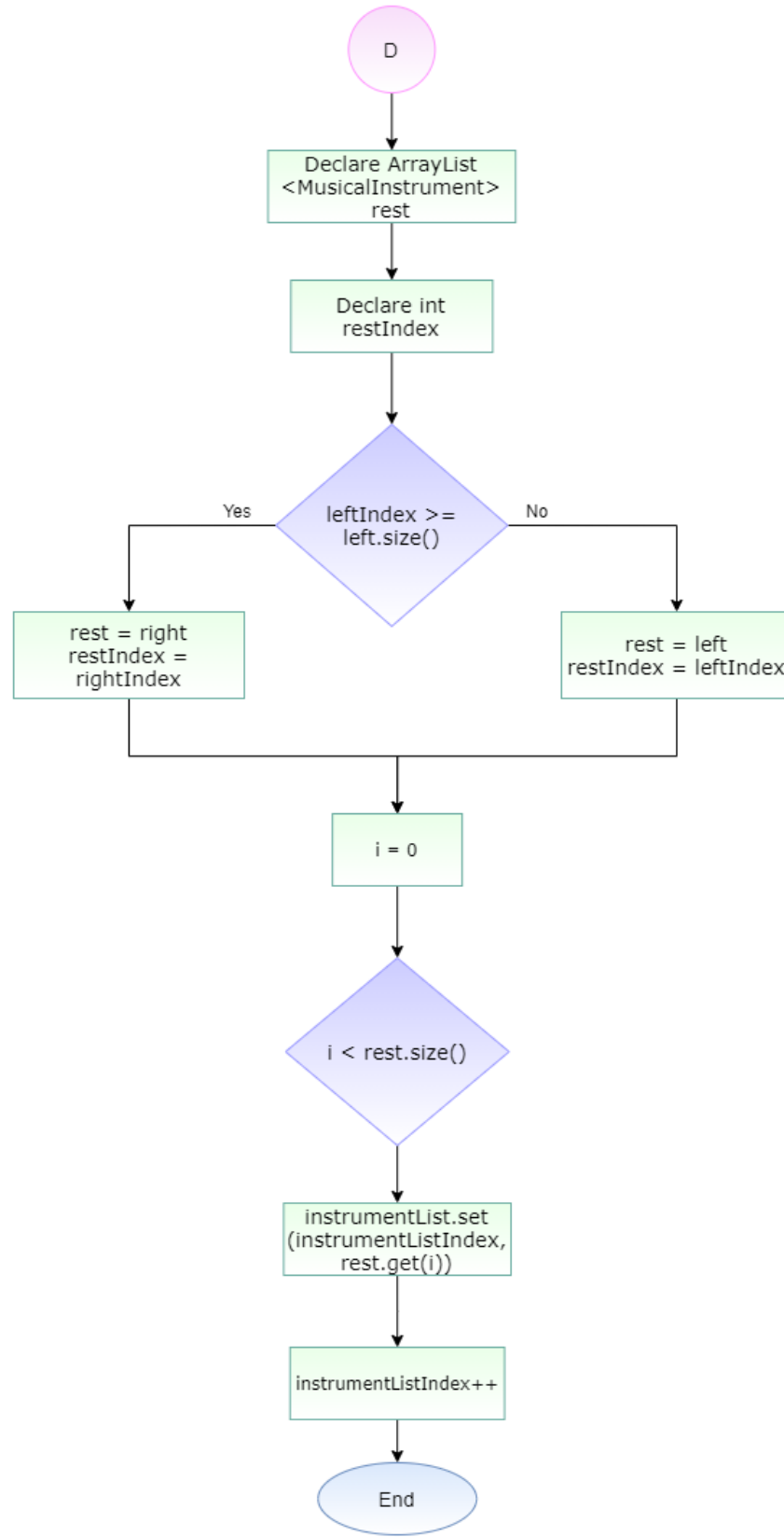


Figure 10: Merge Sort Part 4.

Method Descriptions.**MusicAcademyInfo.java**

Method	Description
void readFile ()	It is responsible for reading the file-Instruments.csv. This method has no return value since it is void.
void displayInstruments()	It is responsible for checking the empty status of the rows and displaying the details of instruments into the JTable. This method has no return value since it is void.
Void clearTable (final JTable table)	(final JTable table) is the parameter passed to the method It is responsible for clearing all the data present in the JTable. This method has no return value since it is void.
void admin_jButtonAction Performed (java.awt.event.ActionEvent evt)	(java.awt.event.ActionEvent evt) is the parameter passed to the method. On being pressed, this method opens the panel for the Admin Login by changing the visibility of other components.
void registerMember_jButtonAction Performed (java.awt.event.ActionEvent evt)	(java.awt.event.ActionEvent evt) is the parameter passed to the method.

	<p>On being pressed, this method opens the panel for the Member Registration by changing the visibility of other components.</p>
void member_jButtonActionPerformed (java.awt.event.ActionEvent evt)	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method.</p> <p>On being pressed, this method opens the panel for the Member Login by changing the visibility of other components.</p>
void adminLogin_jButtonActionPerformed (java.awt.event.ActionEvent evt)	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method.</p> <p>On being pressed, this validates the username and password of the Admin and enables the admin to login and access the panel for adding instruments (if validation is correct).</p>
void memberLogin_jButtonActionPerformed (java.awt.event.ActionEvent evt)	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method.</p> <p>On being pressed, this validates the username and password of the Member using a HashMap and enables the member to login and access the panel containing details of the instruments (if validation is correct).</p>

<p>void backAdmin_jButtonActionPerformed</p> <p>(java.awt.event.ActionEvent evt)</p> <p>void backMember_jButtonActionPerformed</p> <p>(java.awt.event.ActionEvent evt)</p> <p>void</p> <p>backRegistration_jButtonActionPerformed</p> <p>(java.awt.event.ActionEvent evt)</p> <p>void back_jButtonActionPerformed</p> <p>(java.awt.event.ActionEvent evt)</p> <p>void adminBack_jButtonActionPerformed</p> <p>(java.awt.event.ActionEvent evt)</p>	<p>(java.awt.event.ActionEvent evt) is the parameter passed to these method.</p> <p>On being pressed, these buttons enable the returning to the previous panel from the current panel while also clearing data present in the JTable, JTextFields or selections in JRadioButtons/ ButtonGroups.</p>
<p>void submit_jButtonActionPerformed</p> <p>(java.awt.event.ActionEvent evt)</p>	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method.</p> <p>On being pressed, this validates the data entered by the user and adds to the JTable. Also, it writes to the file</p>

	<p>Members.csv with the member's details.</p> <p>(if validation is correct).</p>
void category_jRadioButtonActionPerformed (java.awt.event.ActionEvent evt)	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method.</p> <p>On being pressed, this button enables the sorting of data from the file Instruments.csv in an alphabetical order according to the sort criteria (category). It also displays the sorted data into the JTable.</p>
void exit_jMenuItemActionPerformed (java.awt.event.ActionEvent evt)	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method.</p> <p>On being pressed, this button exits the system.</p>
void add_jButtonActionPerformed (java.awt.event.ActionEvent evt)	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method.</p> <p>On being pressed, this validates the data entered by the Admin and adds to the JTable. Also, it writes to the file Instruments.csv with the instrument's details. (if validation is correct).</p>
void price_jRadioButtonActionPerformed (java.awt.event.ActionEvent evt)	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method.</p>

	<p>On being pressed, this button enables the sorting of data from the file Instruments.csv in an ascending order according to the sort criteria (price). It also displays the sorted data into the JTable.</p>
<p>void clearRegistration_jButtonActionPerformed (java.awt.event.ActionEvent evt)</p> <p>void clearAddInstruments_jButtonActionPerformed (java.awt.event.ActionEvent evt)</p>	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method.</p> <p>On the press of this button, all the JTextFields are made to reset its stored value to an empty string (""), hence clearing all data from the panel.</p>
<p>void searchCategory_jButtonActionPerformed (java.awt.event.ActionEvent evt)</p>	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method.</p> <p>On being pressed, this method</p> <ul style="list-style-type: none"> • Searches the arraylist with the instrument details to find all the instruments selected by the user. • Counts the number of instruments of the selected category.

	<ul style="list-style-type: none"> Displays the total number of selected instruments with their respective model numbers in a JOptionPane.
void searchPrice_jButtonActionPerformed (java.awt.event.ActionEvent evt)	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method. On being pressed, this method</p> <ul style="list-style-type: none"> Searches the instrument with the price input by the user. Displays details of only one instrument with the price in JOptionPane.
void Instruments_jMenuItemActionPerformed (java.awt.event.ActionEvent evt)	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method. On being pressed, this method displays/imports the data from Instruments.csv into the JTable</p>
void External_jMenuItemActionPerformed (java.awt.event.ActionEvent evt)	<p>(java.awt.event.ActionEvent evt) is the parameter passed to the method. On being pressed, this method opens a file chooser dialog box for the user to choose a file and open it immediately without having to surf multiple folders outside the system.</p>

void User_jMenuItemActionPerformed (java.awt.event.ActionEvent evt)	(java.awt.event.ActionEvent evt) is the parameter passed to the method. On being pressed, this method displays a user manual (pdf) consisting of all the basic information and the working flow of our system.
void main(String args[])	void main (String args[]) is the main method of our class "GUI.java" It is used to create an object of HashMap, read from the Members.csv file and validate the login of members.

Table 4: Method Description for GUI.java

MusicalInstrument.java

Method	Description
int getInstrumentId()	Returns the instrument ID.
int getInstrumentName()	Returns the name of the instrument.
int getModelNum()	Returns the model number of the instrument.
int getBrand()	Returns the brand of the instrument.
int getPrice()	Returns the price of the instrument.
int getWarranty()	Returns the warranty of the instrument.

Table 5: Method Description for MusicalInstrument.java

MergeSort.java

Method	Description
<pre>ArrayList<MusicalInstrument> mergeSort(ArrayList<MusicalInstrument> instrumentList, SortBy sortBy)</pre>	<ul style="list-style-type: none"> Sorts the arraylist passed in the parameter using merge sort algorithm. Returns the sorted arraylist.
<pre>void merge(ArrayList<MusicalInstrument> left, ArrayList<MusicalInstrument> right, ArrayList<MusicalInstrument> instrumentList, SortBy sortBy)</pre>	<ul style="list-style-type: none"> Checks if the lists have to be sorted by category or price. If the list has to be sorted by category, then the instrument names are compared and sorted alphabetically. If the list has to be sorted by price, then the price of each instrument is compared and sorted in ascending order. The sorted lists are merged.

*Table 6: Method Description for MergeSort.java***BinarySearch.java**

Method	Description
<pre>MusicalInstrument binarySearch(ArrayList<MusicalInstrument> list, String target)</pre>	<p>The value of searchElement passed in the parameter is searched in the given arraylist.</p> <p>If the required value is found its object gets returned.</p>

Table 7: Method Description for BinarySearch.java

Testing.

Test 1: To test if the program can run in NetBeans.

Test No.	1.
Objective:	To test if the program can run in NetBeans.
Action:	<ul style="list-style-type: none"> ➤ Go to file and open the project in Apache NetBeans. ➤ Run the MusicAcademyInfo.java file.
Expected Result:	The GUI should run properly without any errors.
Actual Result:	The program was compiled and run successfully.
Conclusion:	The test was successful.

Table 8: To test if the program can run in NetBeans.

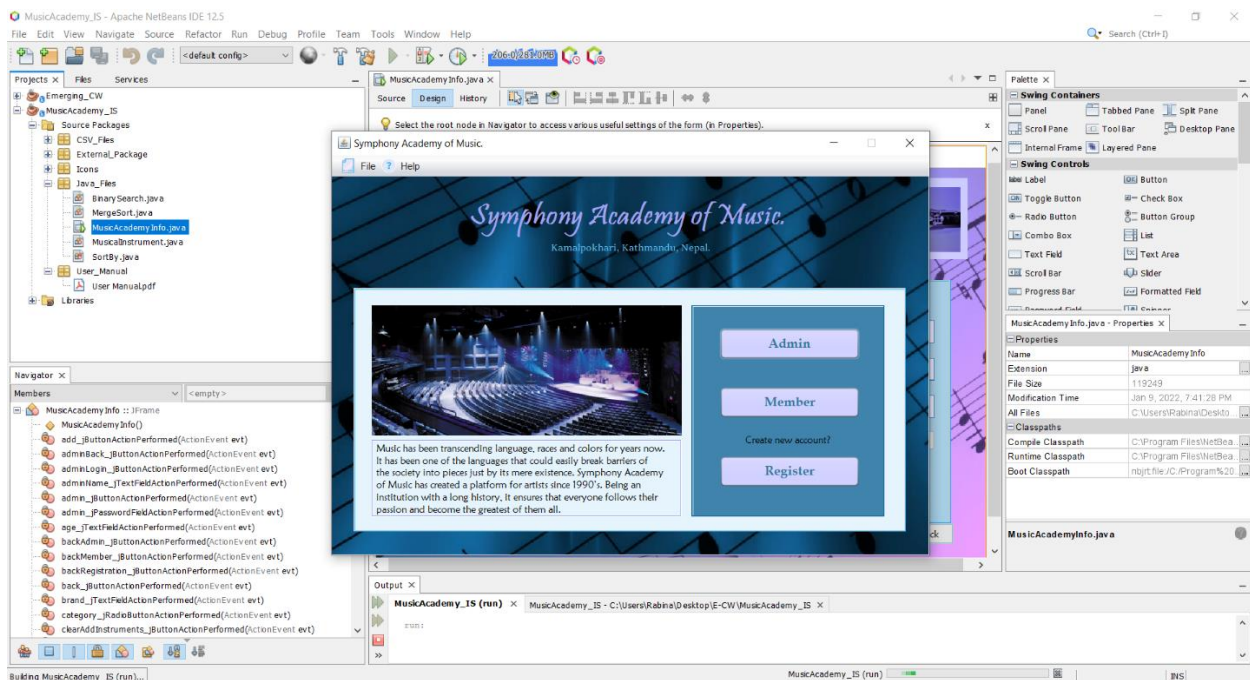
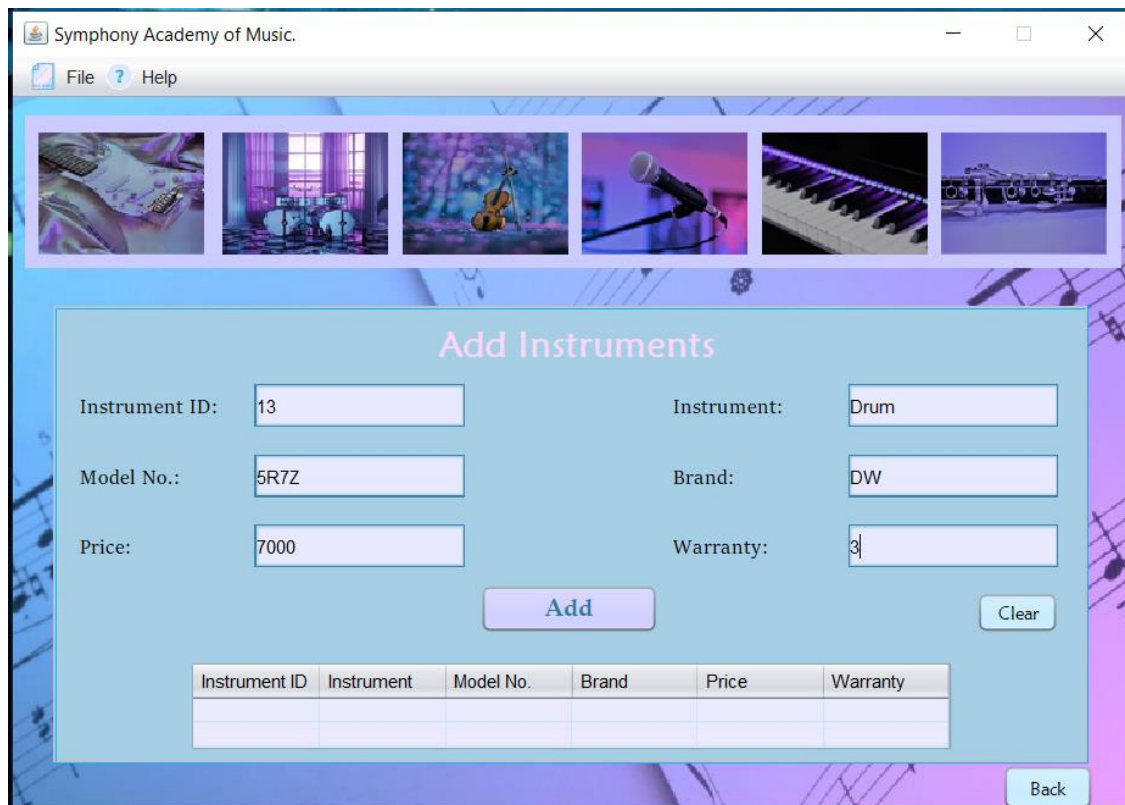


Figure 11: Program can run in NetBeans: Test Successful.

Test 2: To add item details in the table.

Test No.	2.
Objective:	To add item details in the table.
Action:	<p>Admin: Add Instruments.</p> <ul style="list-style-type: none"> ➤ Click on the Admin button and enter the username / password. ➤ After a successful log in: ➤ Enter the instrument details. ➤ Click on the Add button. <p>Registration.</p> <ul style="list-style-type: none"> ➤ Click on the Register button. ➤ Enter the registration details. ➤ Click on the Submit button.
Expected Result:	<p>The JTable and Instruments.csv should be updated with the entered values.</p> <p>The JTable and Members.csv should also be updated.</p>
Actual Result:	The instruments and members csv file along with the JTable was updated with the added / submitted details.
Conclusion:	The test was successful.

Table 9: To add item details in the table.



Symphony Academy of Music.

File ? Help

Add Instruments

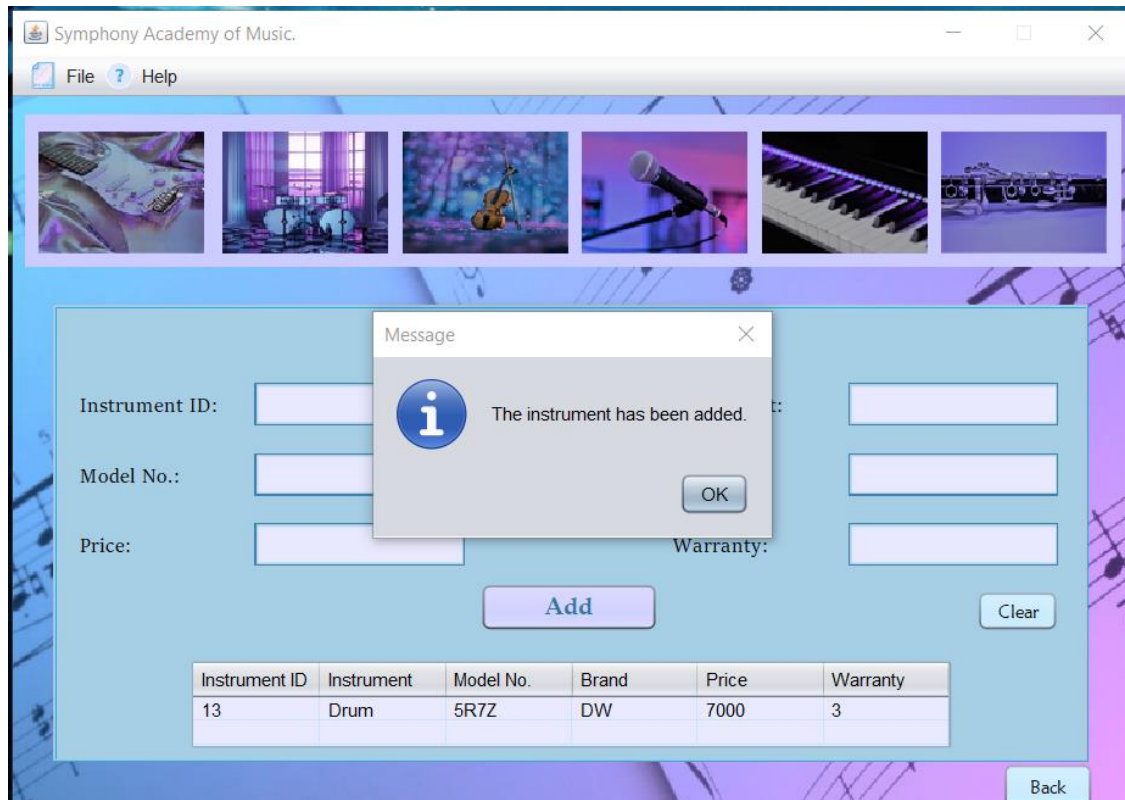
Instrument ID: Instrument:

Model No.: Brand:

Price: Warranty:

Instrument ID	Instrument	Model No.	Brand	Price	Warranty


Figure 12: Adding Instruments.



Symphony Academy of Music.

File ? Help

Message

 The instrument has been added.

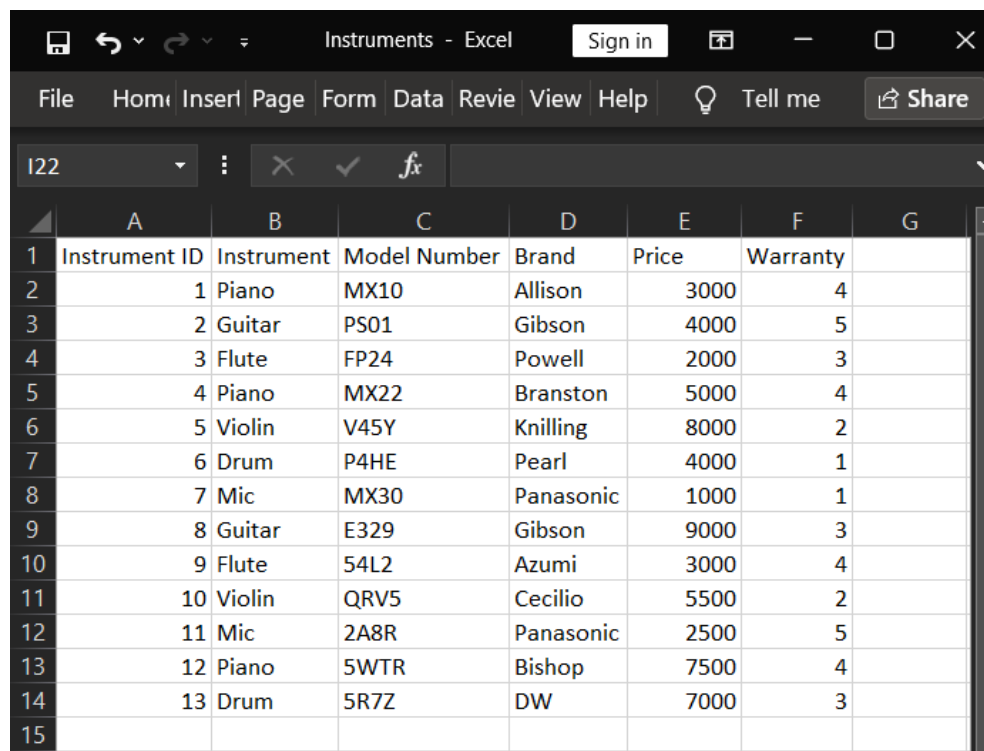
Instrument ID: Instrument:

Model No.: Brand:

Price: Warranty:

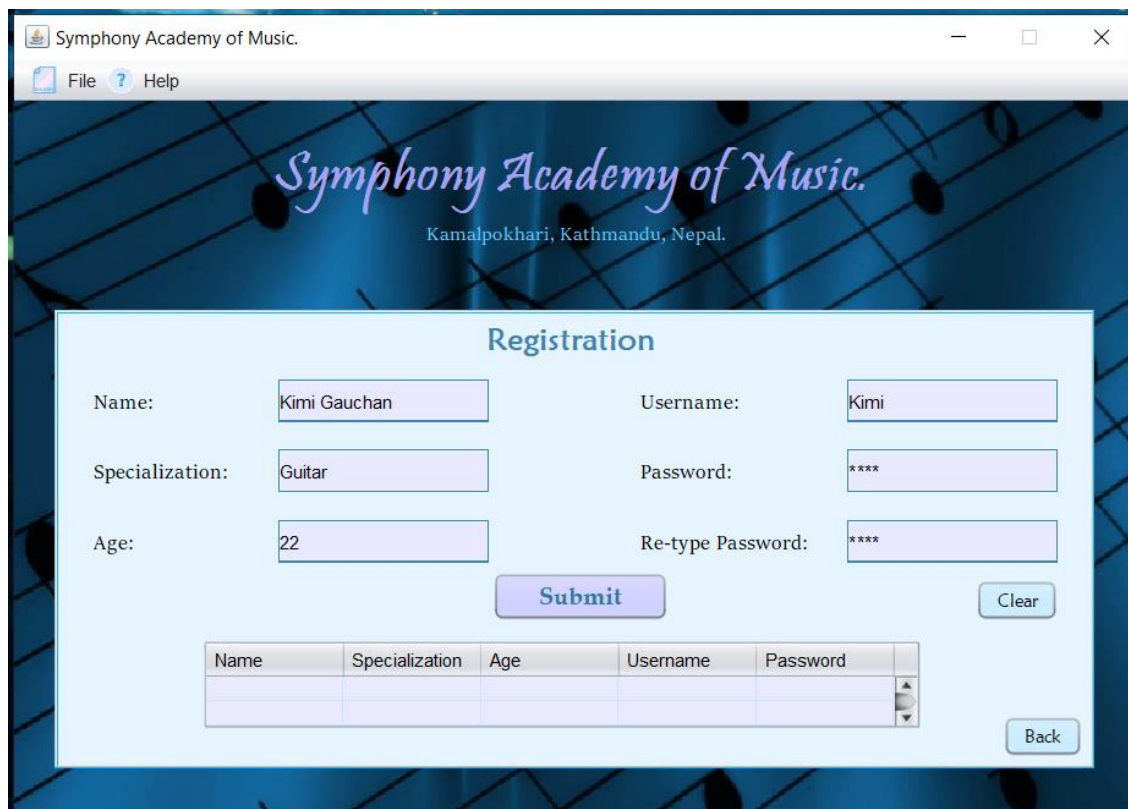
Instrument ID	Instrument	Model No.	Brand	Price	Warranty
13	Drum	5R7Z	DW	7000	3

Figure 13: Instruments has been added.



Instrument ID	Instrument	Model Number	Brand	Price	Warranty
1	Piano	MX10	Allison	3000	4
2	Guitar	PS01	Gibson	4000	5
3	Flute	FP24	Powell	2000	3
4	Piano	MX22	Branston	5000	4
5	Violin	V45Y	Knilling	8000	2
6	Drum	P4HE	Pearl	4000	1
7	Mic	MX30	Panasonic	1000	1
8	Guitar	E329	Gibson	9000	3
9	Flute	54L2	Azumi	3000	4
10	Violin	QRV5	Cecilio	5500	2
11	Mic	2A8R	Panasonic	2500	5
12	Piano	5WTR	Bishop	7500	4
13	Drum	5R7Z	DW	7000	3

Figure 14: Add Instruments: Test Successful.



Symphony Academy of Music.
Kamalpokhari, Kathmandu, Nepal.

Registration

Name: Username:

Specialization: Password:

Age: Re-type Password:

Name	Specialization	Age	Username	Password

Figure 15: Registration.

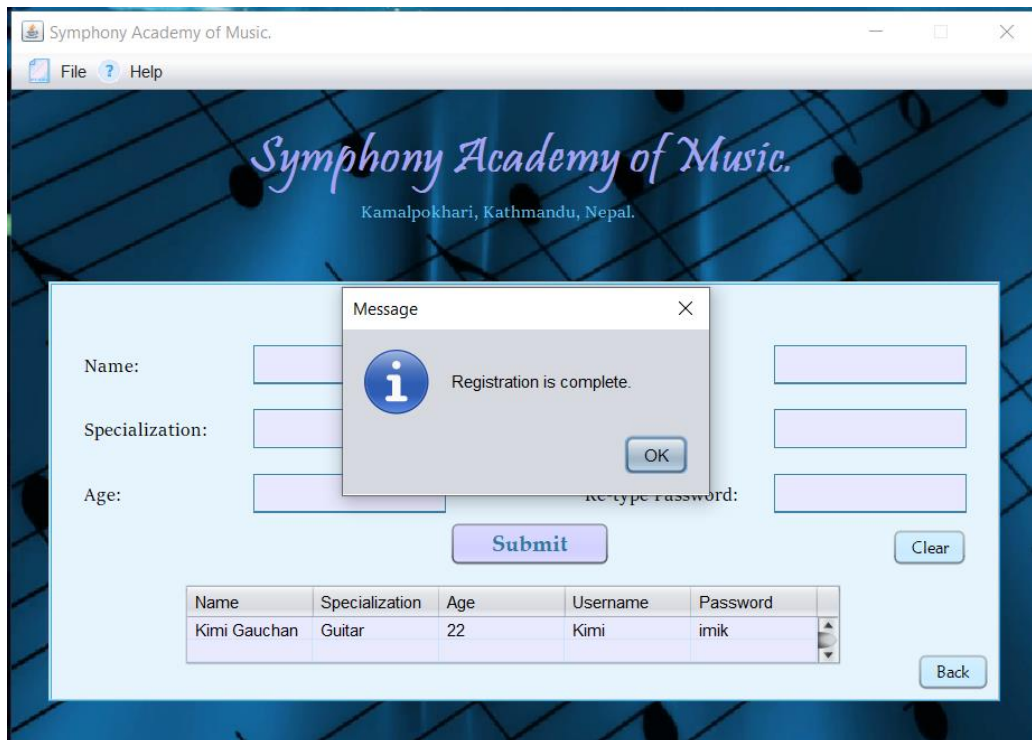


Figure 16: Registration Complete.

Members - Ex... Sign in

File Home Insert Page Layout Formulas Data Review View Help Tell me

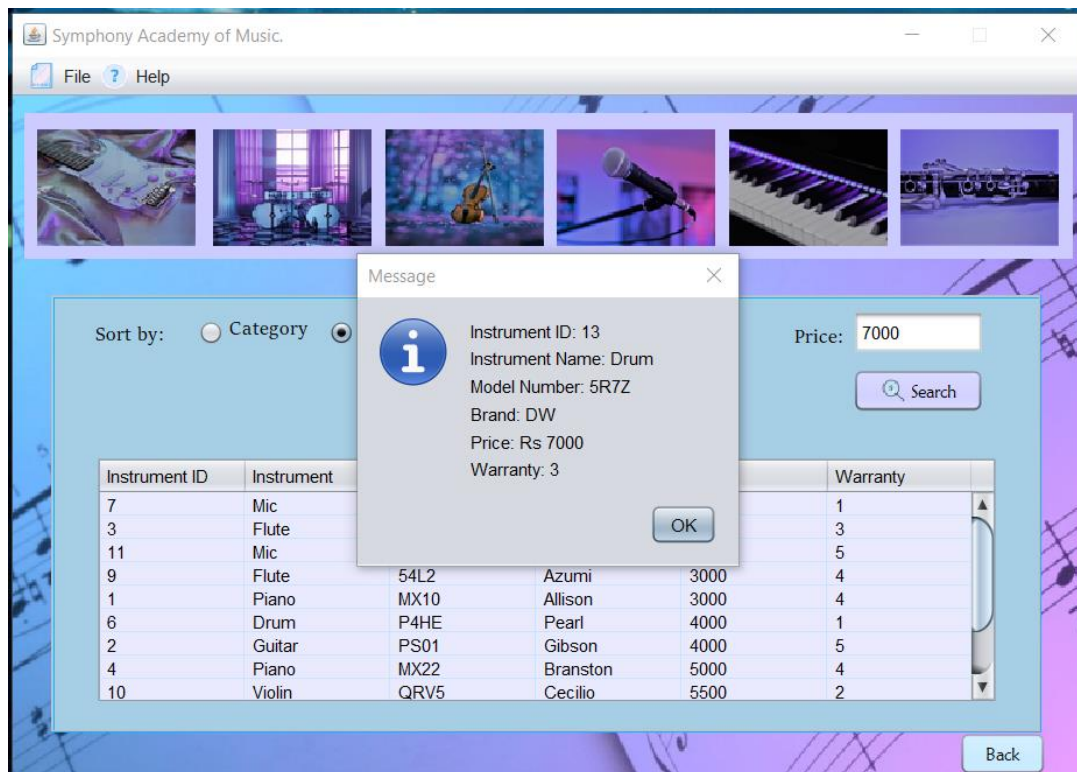
I28

	A	B	C	D	E	F
1	Name	Specialization	Age	Username	Password	
2	Aadesh Shrestha	Piano	20	Aadesh	hsedaa	
3	Aditya Amatya	Violin	23	Aditya	aytida	
4	Ameesha Thapa	Vocal	19	Ameesha	ahseema	
5	Ramesh Basnet	Guitar	32	Ramesh	hsemar	
6	Abhinav Sharma	Flute	28	Abhinav	vanihba	
7	Rabin Acharya	Piano	16	Rabin	nibar	
8	Namrata Sitaula	Guitar	27	Namrata	atarman	
9	Ishani Karki	Drum	18	Ishani	inahsi	
10	Ankur Tamang	Violin	29	Ankur	rukna	
11	Ujwal Khadka	Flute	21	Ujwal	lawju	
12	Anish Shah	Vocal	30	Anish	hsina	
13	Presha Bhattarai	Piano	22	Presha	ahserp	
14	Mubson Karki	Vocal	20	Mubson	nosbum	
15	Shreesu Thapa	Guitar	20	Shreesu	useerhs	
16	Aakash Shrestha	Violin	20	Aakash	hsakaa	
17	Subriti Aryal	Guitar	20	Subriti	itirbus	
18	Rabina Shrestha	Drum	20	Rabina	anibar	
19	Aashna Shrestha	Piano	20	Aashna	anhsaa	
20	Kimi Gauchan	Guitar	22	Kimi	imik	
21						

Figure 17: Registration: Test Successful.

Test 3: To search for an instrument based on price

Test No.	3.
Objective:	To search for an instrument based on price.
Action:	<ul style="list-style-type: none"> ➤ Click on the Members button and log in to the account. ➤ Click on file > Open > Import Instruments. ➤ Click on Sort by: Price. ➤ Enter the price and click the Search button.
Expected Result:	A JOptionPane should pop up and give detail about one product available in that specific price. If there is no instrument of that price an appropriate message should pop up.
Actual Result:	A JOptionPane popped up and gave details about one product in the given price.
Conclusion:	The test was successful.

Table 10: To search for an instrument based on price.*Figure 18: Search an Instrument using Price: Test Successful.*

Test 4: To search for the number of Instruments available in a category.

Test No.	4.
Objective:	To search for the number of Instruments available in a category.
Action:	<ul style="list-style-type: none"> ➤ Click on the Members button and log in to the account. ➤ Click on file > Open > Import Instruments. ➤ Click on Sort by: Category. ➤ Choose the Instrument Category and click the Search button.
Expected Result:	A JOptionPane should pop up and give the number of Instruments available in the chosen category along with the model number.
Actual Result:	A JOptionPane popped up and gave the number of Instruments available in the chosen category along with the model number.
Conclusion:	The test was successful.

Table 11: To search the number of Instruments available in a Category.

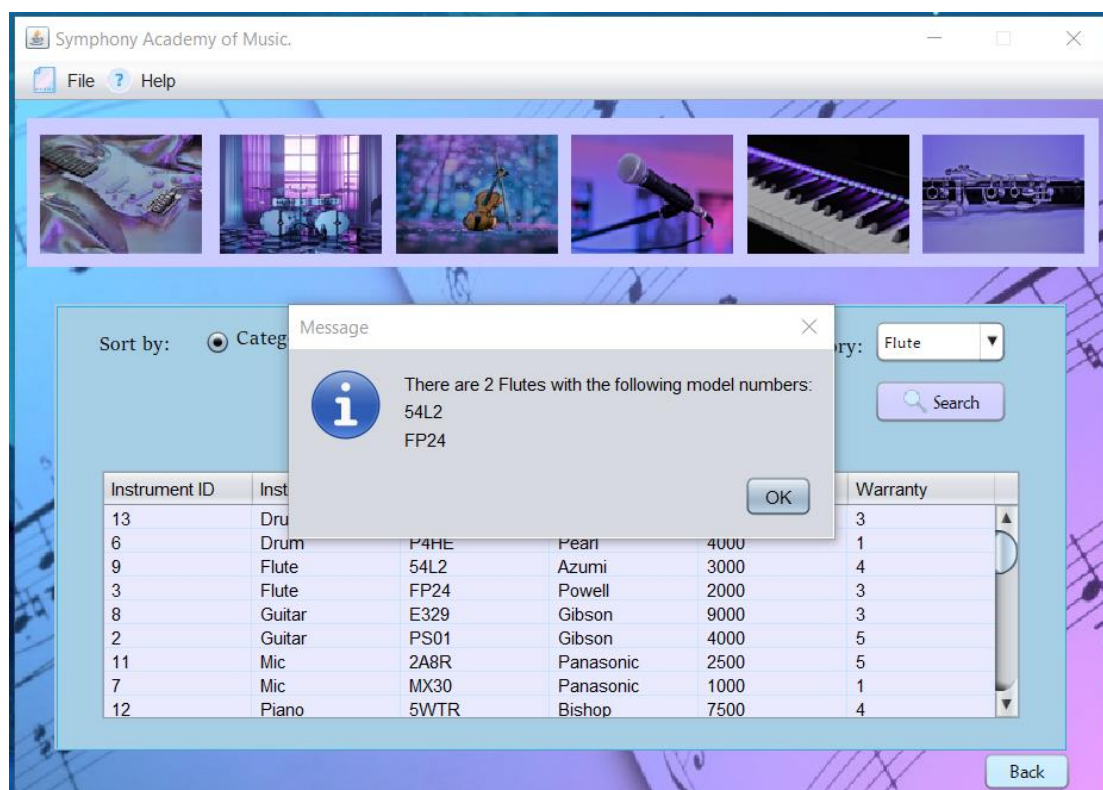


Figure 19: Number of Instruments Available in a Category: Test Successful.

Test 5: To open a file from menu.

Test No.	5.
Objective:	To open a file from menu.
Action:	<p>File: Open: Import Instruments.</p> <ul style="list-style-type: none"> ➤ Click on the Member button and log in to the account. ➤ Click on File > Open > Import Instruments. <p>File: Open: Open External Files.</p> <ul style="list-style-type: none"> ➤ Click on File > Open > External Files. <p>Help > User Manual.</p> <ul style="list-style-type: none"> ➤ Click on Help > User Manual.
Expected Result:	<p>Import Instruments: The Instrument details should get imported from the csv file.</p> <p>Open External Files: Any file chosen should open up.</p> <p>User Manual: The user manual pdf should open up.</p>
Actual Result:	<p>Import Instruments: The Instrument details was imported from the csv file and was displayed in the instruments table.</p> <p>Open External Files: The chosen file opened up.</p> <p>User Manual: The user manual pdf opened up.</p>
Conclusion:	The test was successful.

Table 12: To open a file from menu.

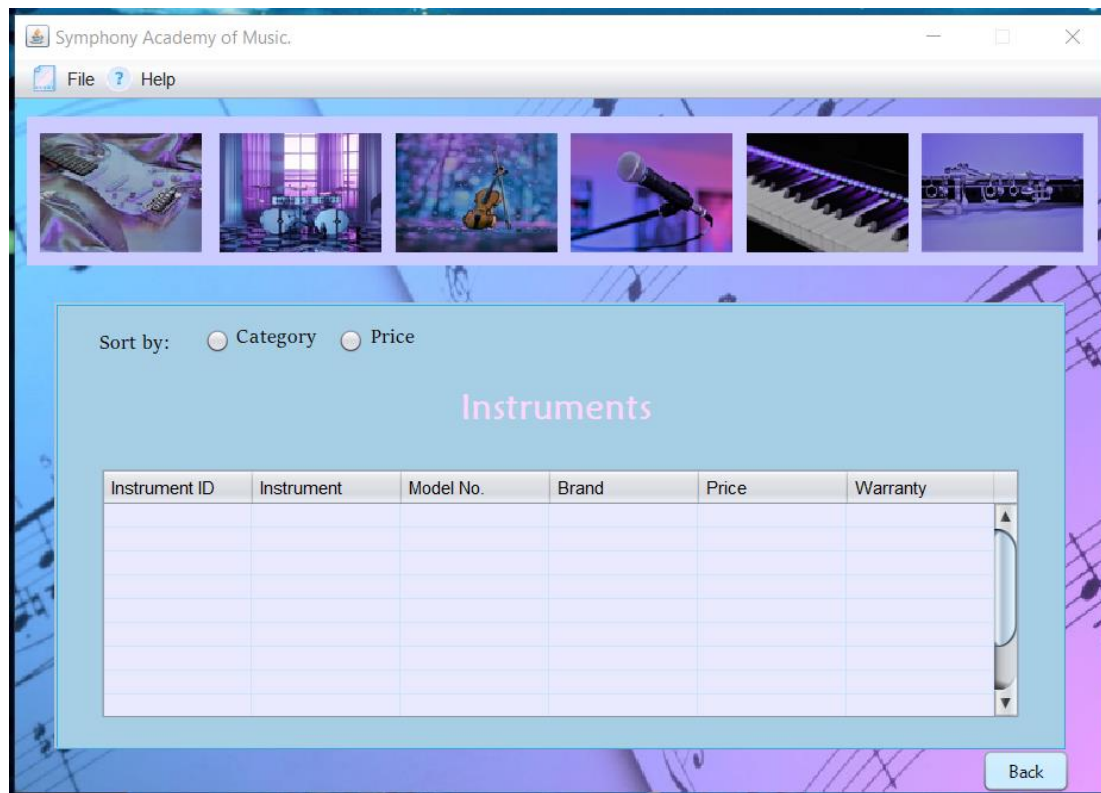


Figure 20: Open: Import Instruments.

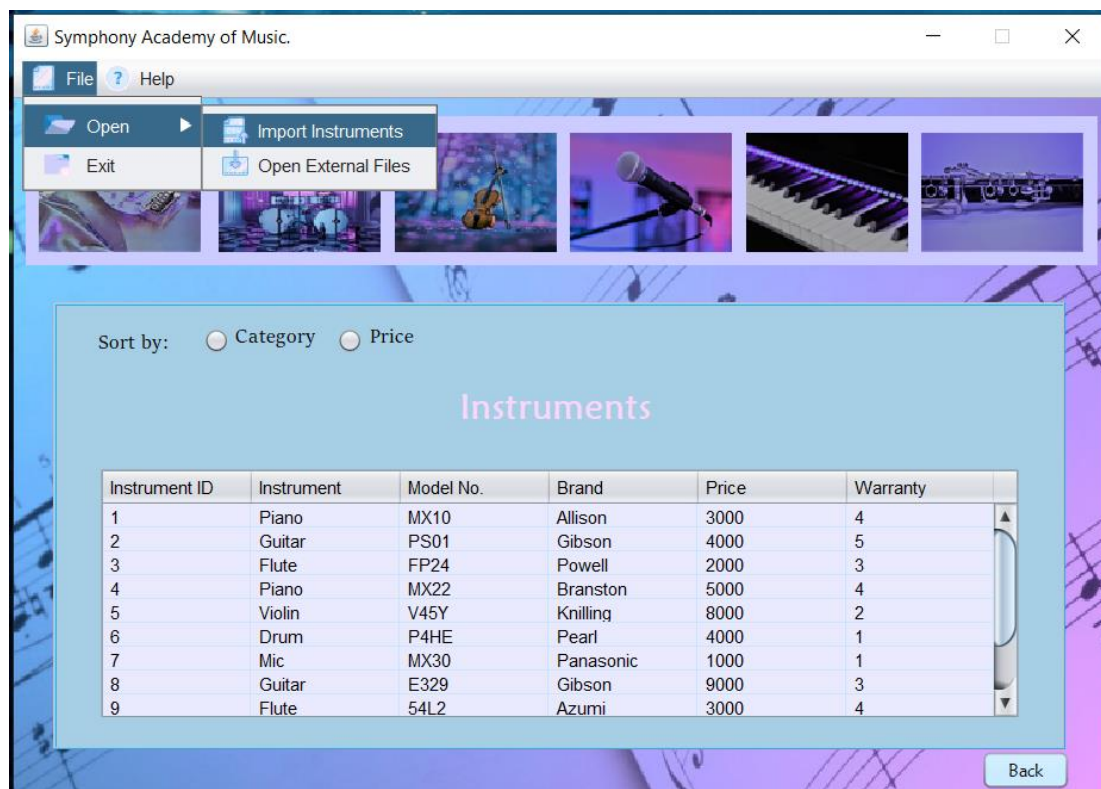


Figure 21: Open: Import Instruments: Successful.

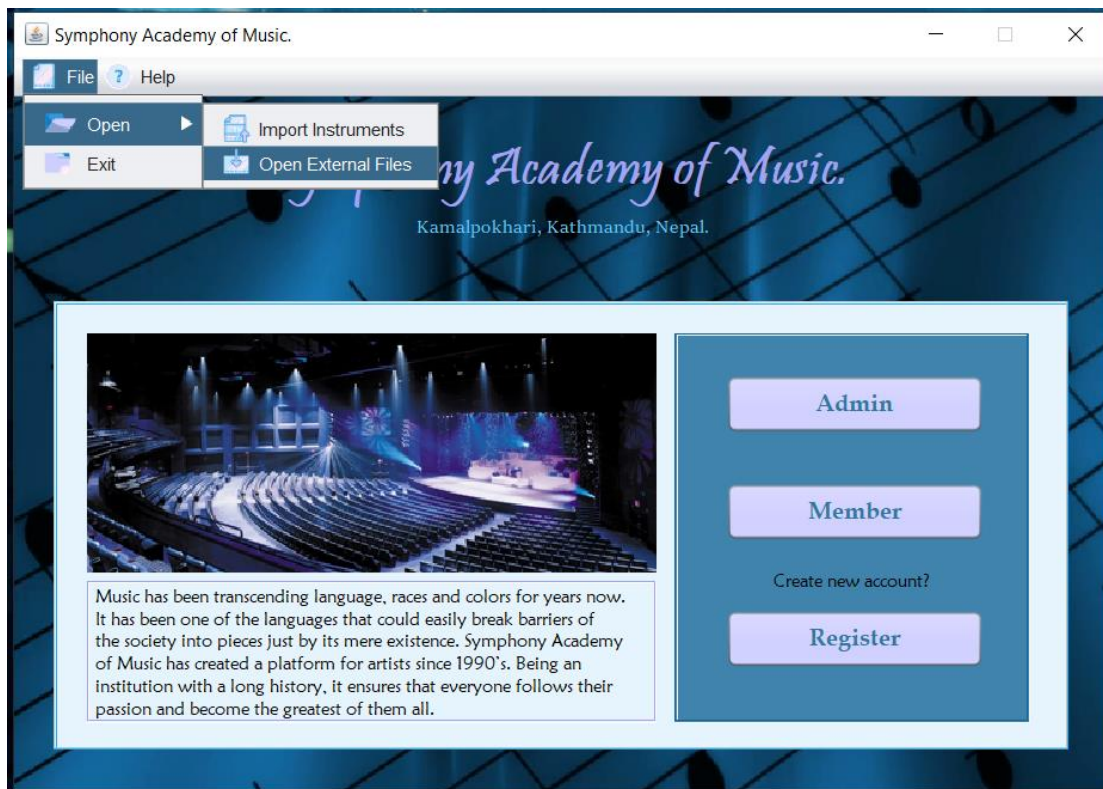


Figure 22: Open: External Files.

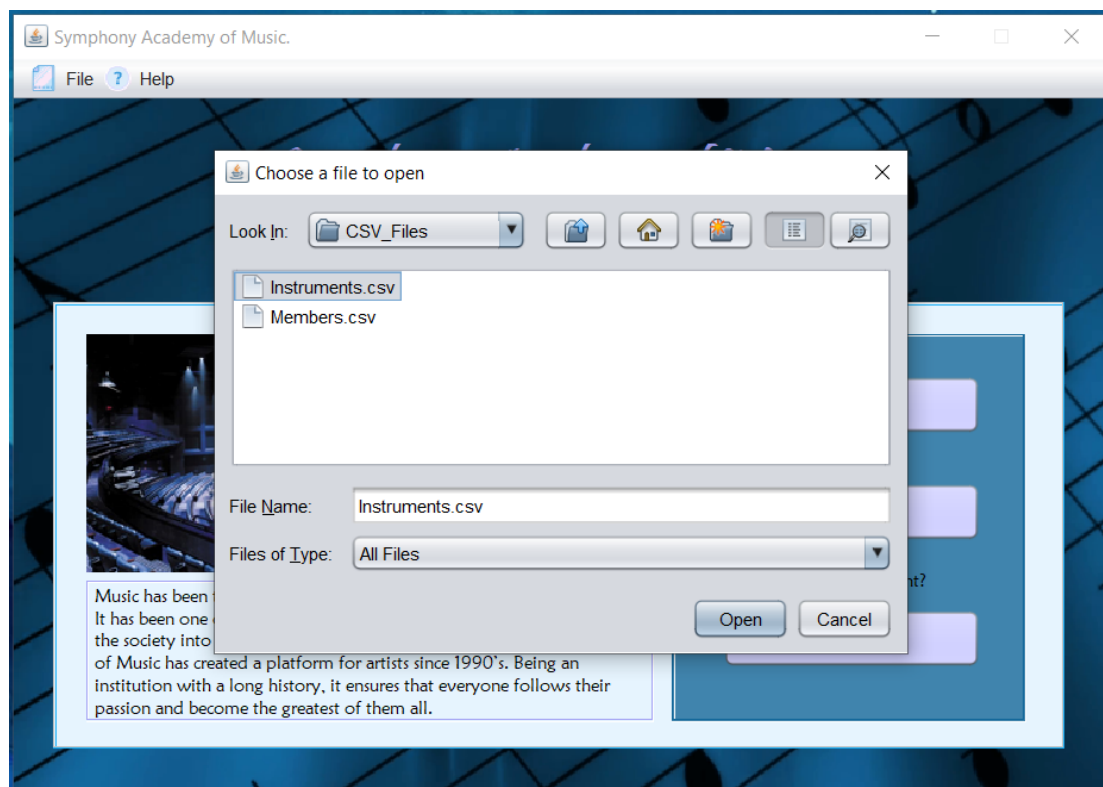
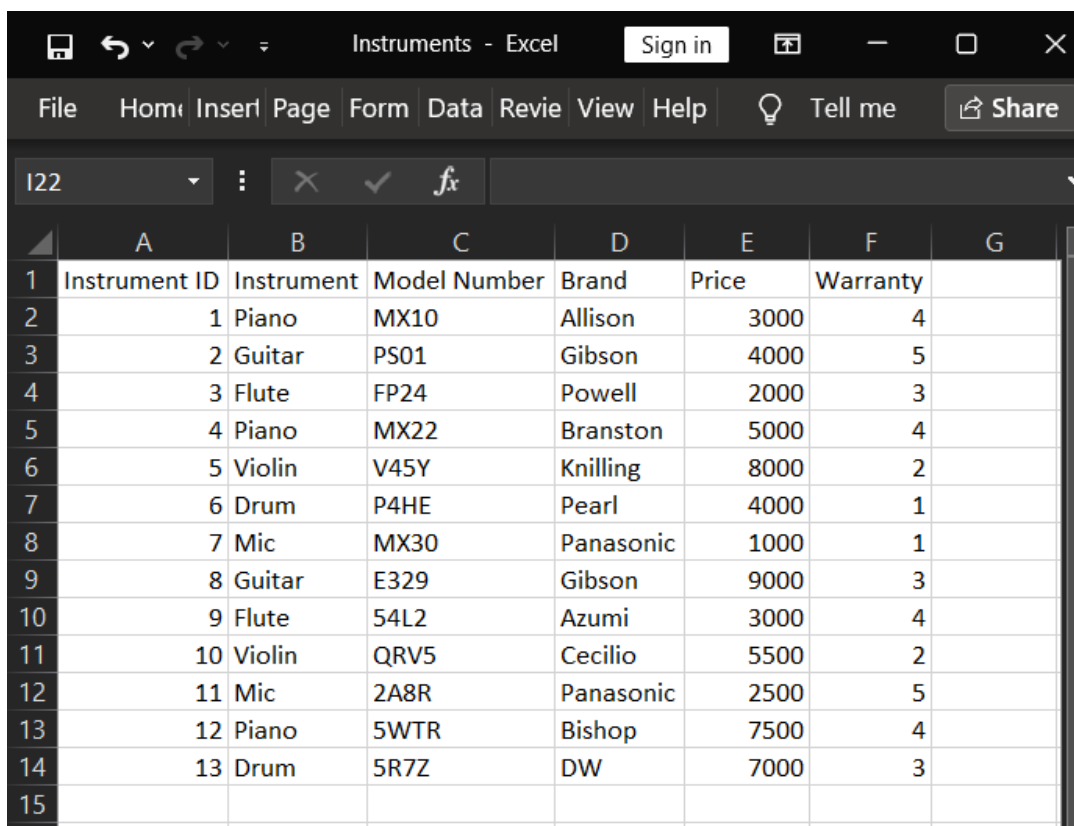


Figure 23: Open: External Files: Choosing the File.



The screenshot shows an Excel spreadsheet with the following data:

	A	B	C	D	E	F	G
1	Instrument ID	Instrument	Model Number	Brand	Price	Warranty	
2	1	Piano	MX10	Allison	3000	4	
3	2	Guitar	PS01	Gibson	4000	5	
4	3	Flute	FP24	Powell	2000	3	
5	4	Piano	MX22	Branston	5000	4	
6	5	Violin	V45Y	Knilling	8000	2	
7	6	Drum	P4HE	Pearl	4000	1	
8	7	Mic	MX30	Panasonic	1000	1	
9	8	Guitar	E329	Gibson	9000	3	
10	9	Flute	54L2	Azumi	3000	4	
11	10	Violin	QRV5	Cecilio	5500	2	
12	11	Mic	2A8R	Panasonic	2500	5	
13	12	Piano	5WTR	Bishop	7500	4	
14	13	Drum	5R7Z	DW	7000	3	
15							

Figure 24: Open: External Files: Test Successful.



Figure 25: Help: User Manual.

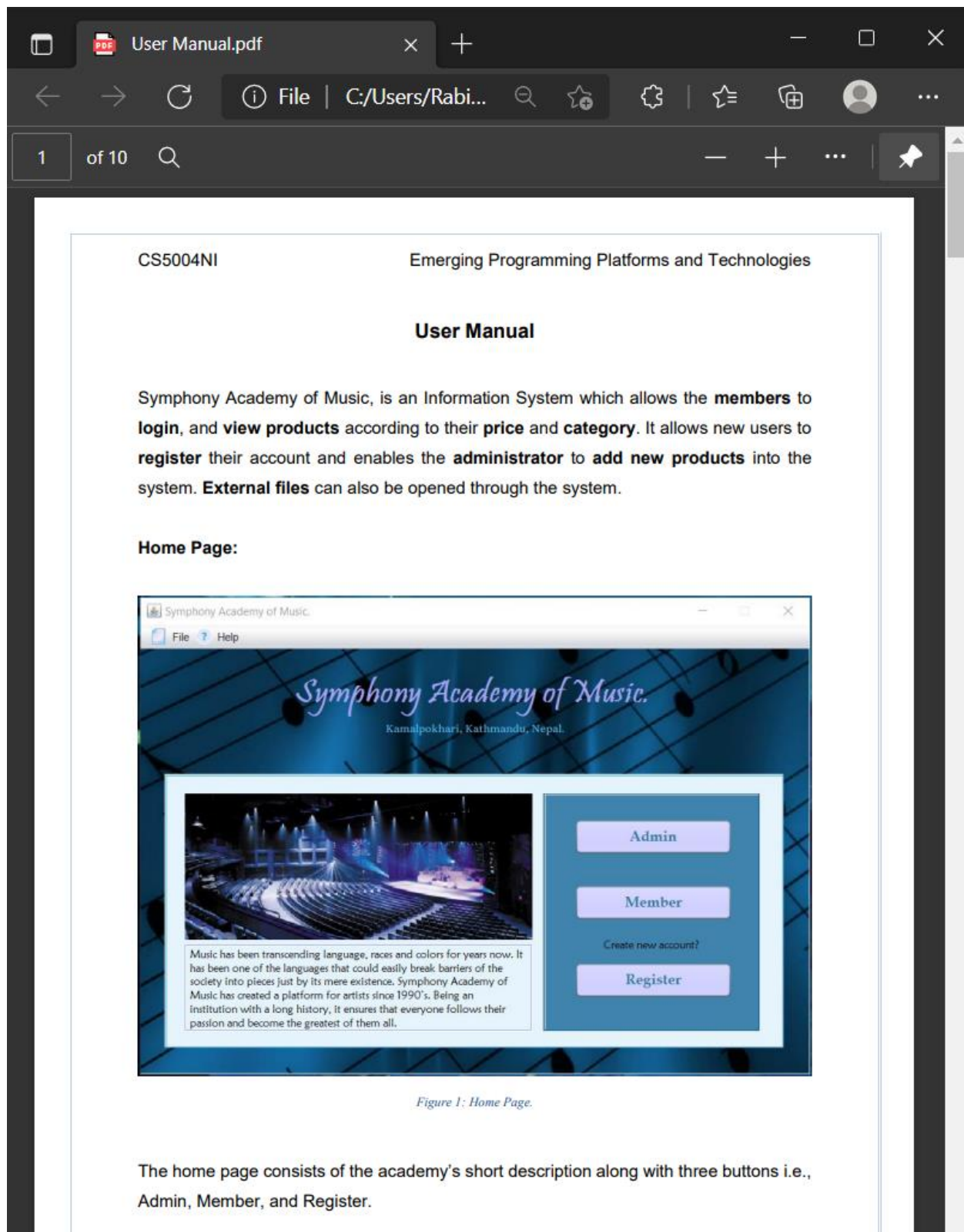


Figure 1: Home Page.

Figure 26: Help: Open User Manual: Test Successful.

Test 6: System Validation.

Test No.	6.
Objective:	System Validation.
Action:	<p>Admin:</p> <ul style="list-style-type: none"> ➤ Enter a wrong username or wrong password. ➤ Enter the price in string. <p>Member:</p> <ul style="list-style-type: none"> ➤ Enter the wrong username or wrong password. ➤ Leave the fields empty. ➤ Leave the price text field empty while searching. <p>Registration:</p> <ul style="list-style-type: none"> ➤ Leave the fields empty. ➤ Enter the age in string. ➤ Enter two different passwords in password and re-password.
Expected Result:	Appropriate dialog boxes should pop up when unsuitable / no values are entered.
Actual Result:	Appropriate dialog boxes popped up when unsuitable / no values are entered.
Conclusion:	The test was successful.

Table 13: To open a file from menu.



Figure 27: Admin: Wrong Username: Test Successful.

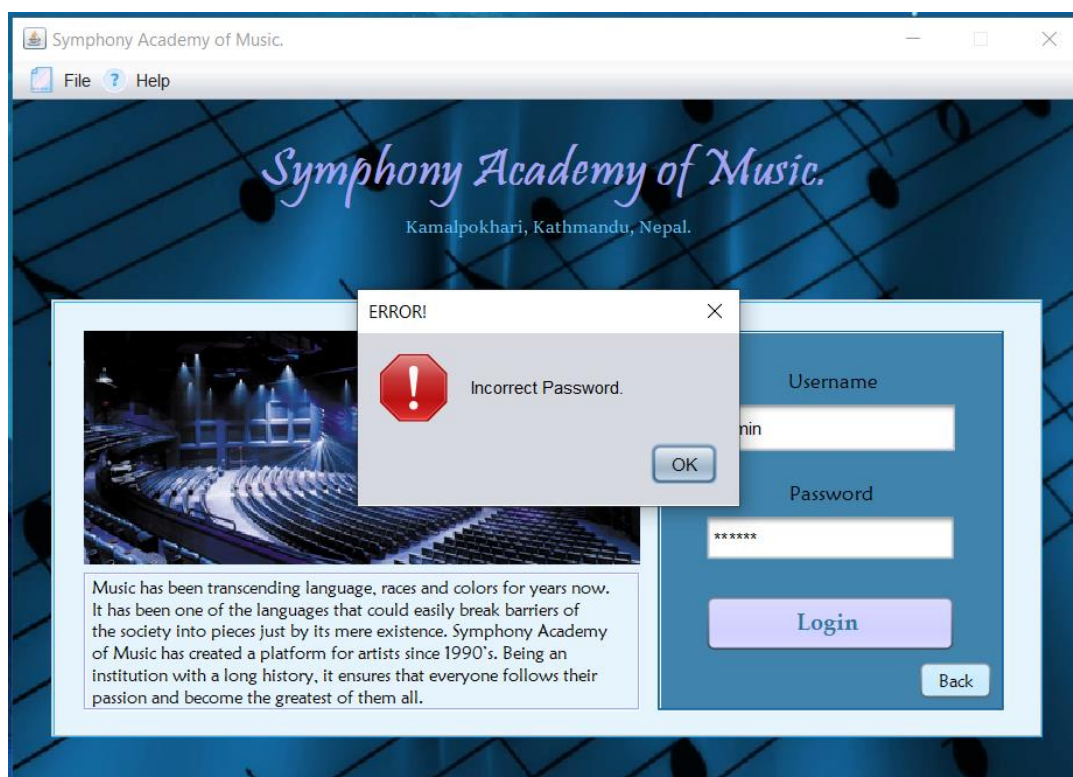


Figure 28: Admin: Incorrect Password: Text Successful.

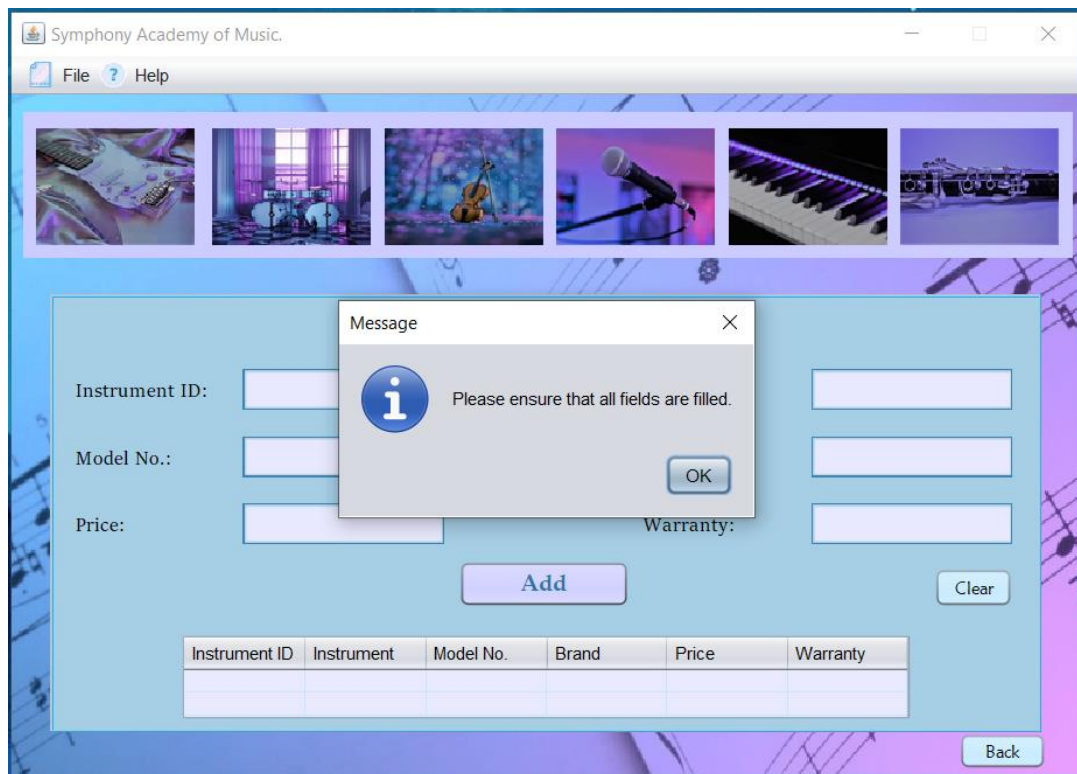


Figure 29: Admin: Empty Fields: Test Successful.

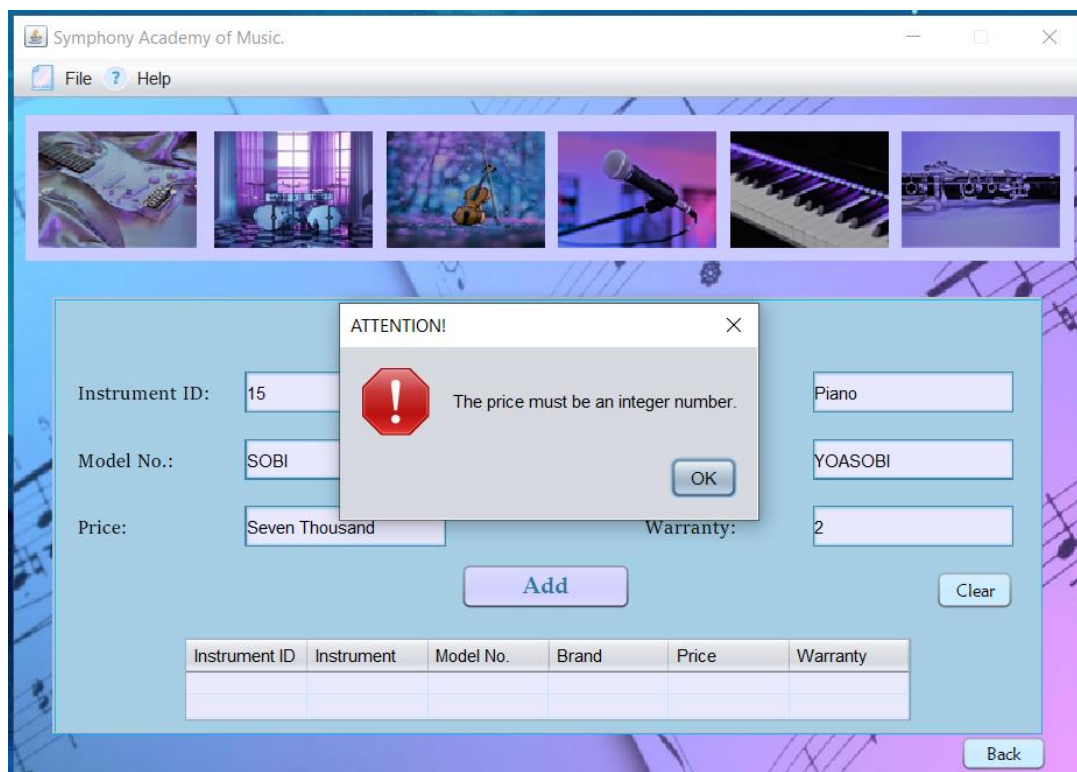


Figure 30: Admin: Price Validation: Test Successful.



Figure 31: Member: Wrong Username: Test Successful.



Figure 32: Member: Invalid Password: Test Successful.

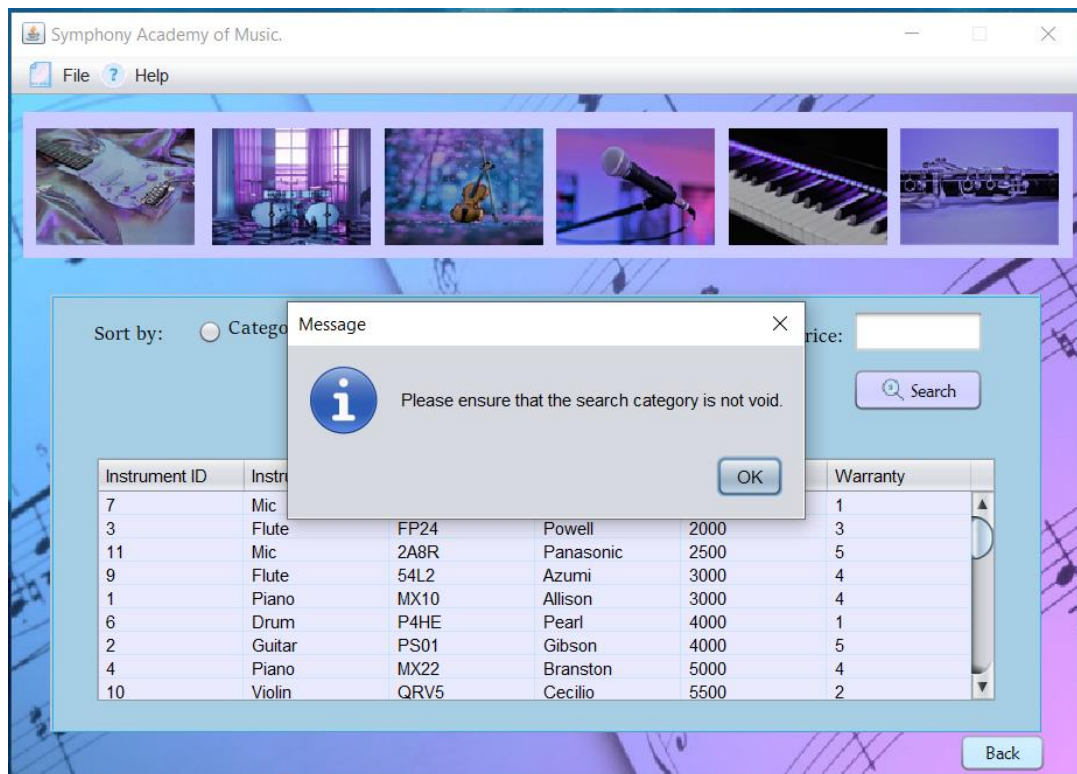


Figure 33: Member: Search Validation: Test Successful.

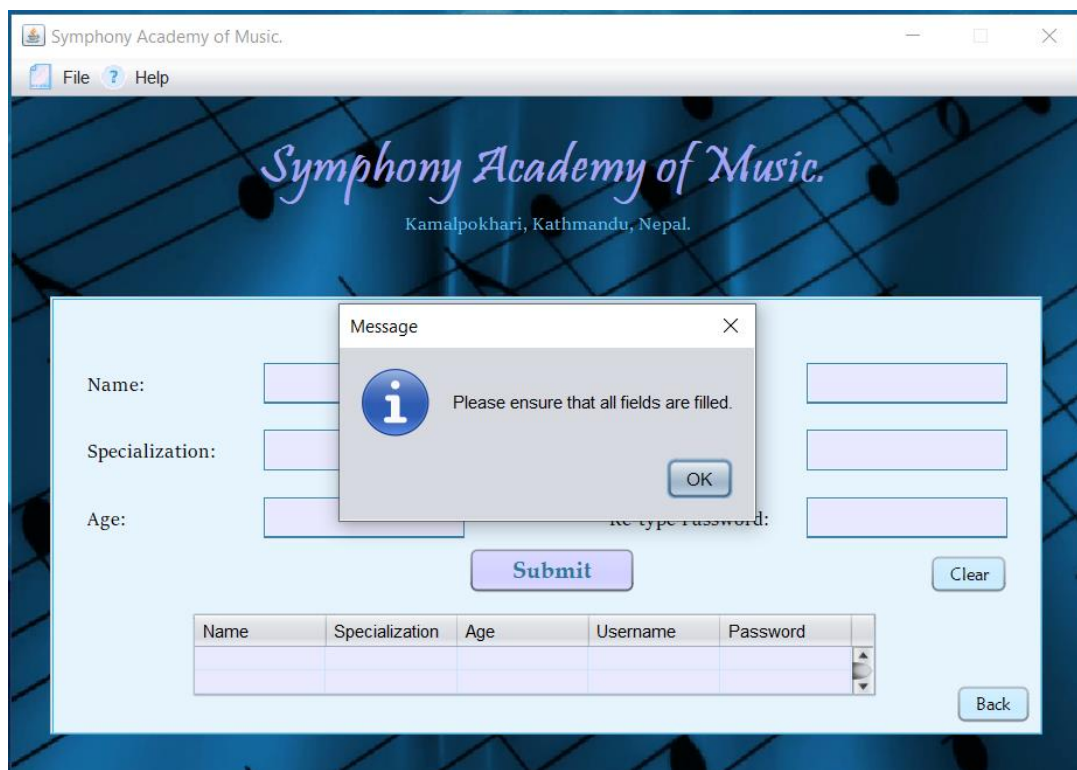


Figure 34: Registration: Empty Fields: Test Successful.

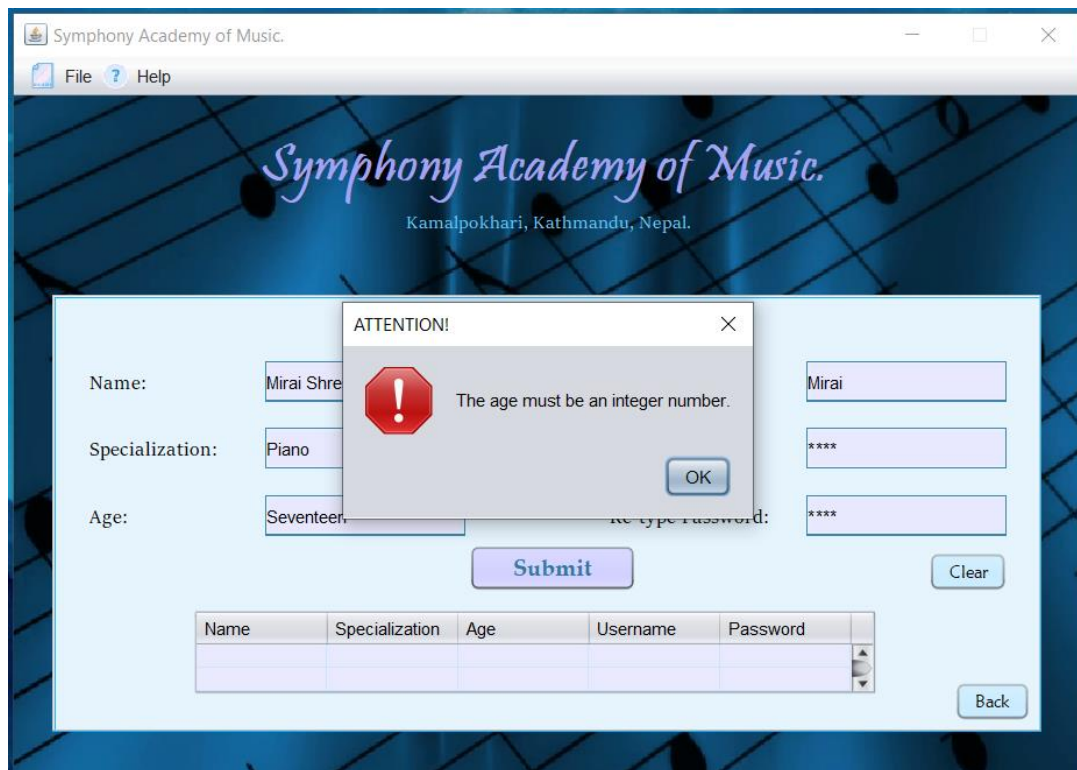


Figure 35: Registration: Age Validation: Test Successful.

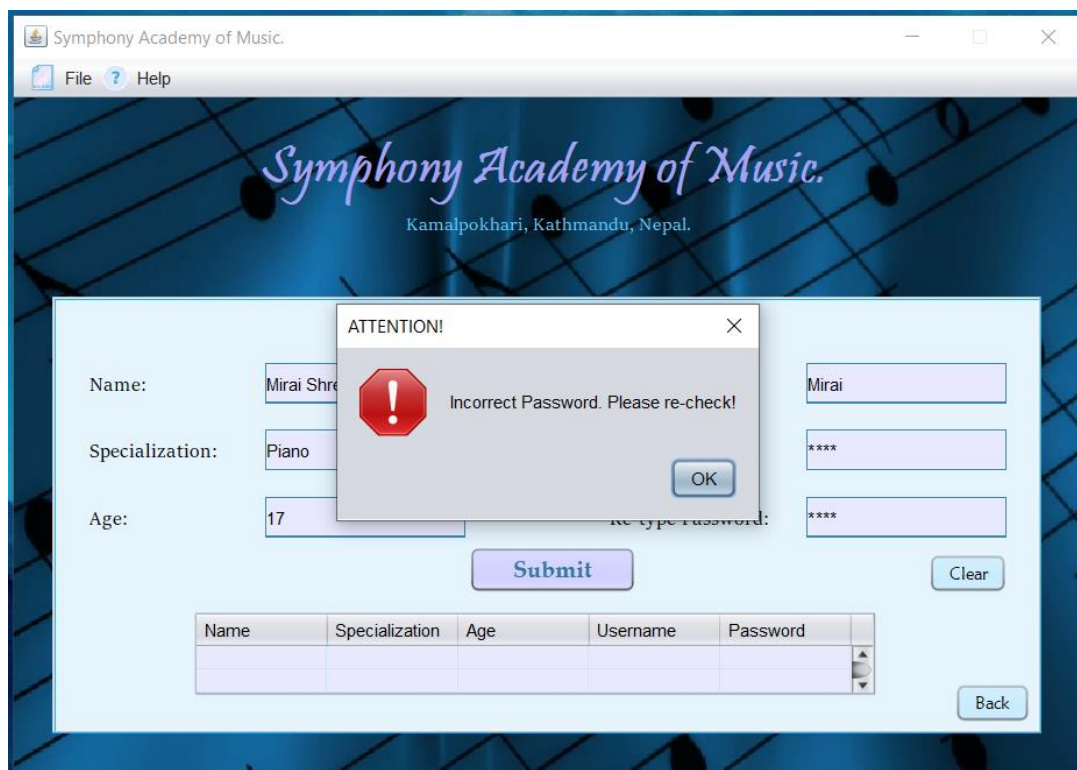


Figure 36: Registration: Password Validation: Test Successful.

Conclusion

In the end, this coursework proved to be informative as well as resourceful as it was completed with many trials, errors, and a lot of research. It mostly benefited the students in understanding the working mechanism behind the binary search and sorting algorithms. A proposal was prepared along with the wireframe to create a rough sketch of how the actual system would look like and the necessary features were noted down.

NetBeans was used to create the graphic user interface since it has a drag and drop functionality it made it much easier to design according to our own aesthetic. However, the main objective of this coursework, was to develop a working system which sorts and displays the details of each product in a table based on the user's input. Binary search was used to display the instrument according to the price whereas merge sort was used to sort out the instruments along with its price. The sorted data will be presented in the table, or a JOptionPane will appear to notify the user.

Key-value pairs are essential and are utilized everywhere. In this coursework as well, HashMap in java is implemented that enables the member to login to their account. If the data matches, the key username can retrieve the corresponding data i.e., the password, making the login successful. If the user types in the incorrect password or username, a JOptionPane will pop up to alert the user.

Although it is a group coursework, it was not possible for the members to meet up every day and work on the project together. Despite the fact that the work was split, the compilation of the programs was a concern, and GitHub came to our rescue. It allowed us to collaborate, merge the code into the main file, fix any errors, make adjustments, and work efficiently.

Programs and errors come hand in hand, and one single error could prohibit the program from running. While completing the assessment, came the difficulties which were rigid. Complications arose regarding searching, sorting, and registering, but with the help of group members along with lots of research, support from the teachers, and the materials provided to us we were able to see the end of it.

Bibliography.

Codexpedia, 2016. *Codexpedia*. [Online]

Available at: <https://www.codexpedia.com/java/java-merge-sort-implementation/>

[Accessed 5 January 2022].

javaTpoint, 2021. *javaTpoint*. [Online]

Available at: <https://www.javatpoint.com/java-filereader-class>

[Accessed 5 January 2022].

javaTpoint, 2021. *javaTpoint*. [Online]

Available at: <https://www.javatpoint.com/merge-sort>

[Accessed 5 January 2022].

javaTpoint, 2021. *javaTpoint*. [Online]

Available at: <https://www.javatpoint.com/binary-search>

[Accessed 5 January 2022].

Kumarsen, A., 2021. *GeeksforGeeks*. [Online]

Available at: <https://www.geeksforgeeks.org/java-program-to-perform-binary-search-on-arraylist/>

[Accessed 7 January 2022].

Meinecke, L., 2016. *study.com*. [Online]

Available at: <https://study.com/academy/lesson/what-is-an-algorithm-in-programming-definition-examples-analysis.html>

[Accessed 5 December 2021].

Ofni Systems, 2018. *Ofni Systems*. [Online]

Available at: <http://www.ofnisystems.com/services/validation/design-specification/>

[Accessed 20 December 2021].

Singh, A., 2021. *GeeksforGeeks*. [Online]

Available at: <https://www.geeksforgeeks.org/writing-a-csv-file-in-java-using-opencsv/>

[Accessed 7 December 2021].

Appendix

For the Teachers:

Given below are few login credentials for accessing the system as it is password-protected.

Admin Login

Username- Admin

Password- @admin

Member Login

Username	Password
Aashna	anhsaa
Rabina	anibar
Subriti	itirbus

User Manual

Symphony Academy of Music, is an Information System which allows the members to login, and view products according to their price and category. It allows new users to register their account and enables the administrator to add new products into the system. External files can also be opened through the system.

Home Page:



Figure 37: Home Page.

The home page consists of the academy's short description along with three buttons i.e., Admin, Member, and Register.

Admin:

If the user is a person responsible for adding new products click on the “Admin” button.



Figure 38: Admin Button.

A login form with a blue background. It contains three input fields: "Username" at the top, "Password" in the middle, and a "Login" button at the bottom. A "Back" button is located in the bottom right corner.

Figure 39: Admin Login Details.

Once the button named admin is clicked, the administrator will be able to see the Figure 39 in the right-hand side given above. Now, the administrator has to enter the username and password in the respective fields and click on the login button. After the login is successful, the admin will be able to see the Figure 4 given down below.

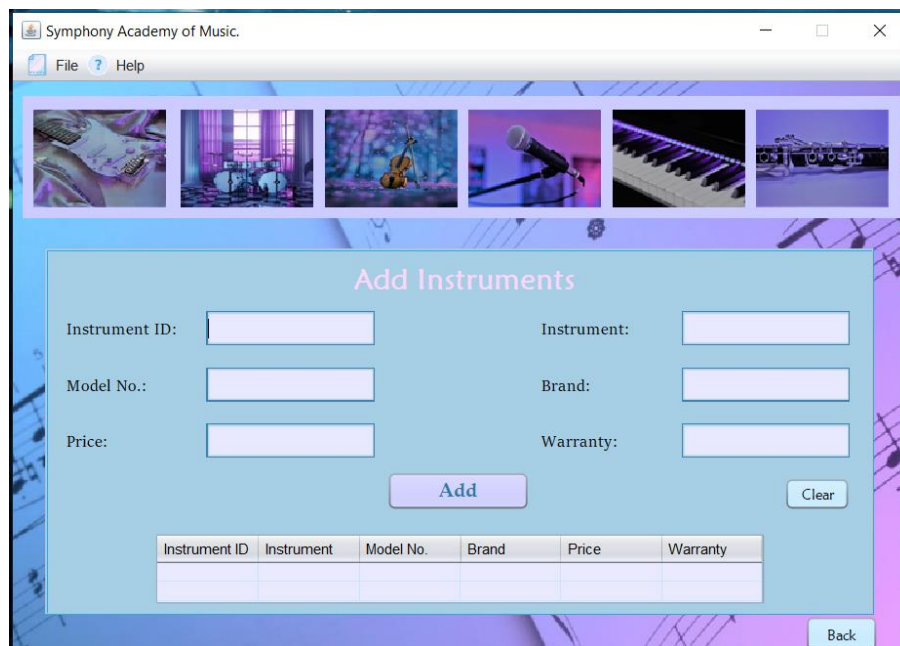
A screenshot of a web application window titled "Symphony Academy of Music." The window shows a form titled "Add Instruments" with a light blue background. The form has six input fields: "Instrument ID:", "Instrument:", "Model No.:", "Brand:", "Price:", and "Warranty:". Below the fields are "Add" and "Clear" buttons. At the bottom, there is a table with six columns: "Instrument ID", "Instrument", "Model No.", "Brand", "Price", and "Warranty". The table is currently empty. A "Back" button is located in the bottom right corner of the window.

Figure 40: Admin: Add Instruments.

The admin can now fill in the details of the instruments that has to be added into the system. After all the details are filled, one must click on the add button to successfully add the product into the existing list. If the instrument has been successfully added, a message box will pop up to notify indicating that the instrument has been added and the details of the added instrument will be shown in the table. However, if it pops an error, one must check the values according to the message box.

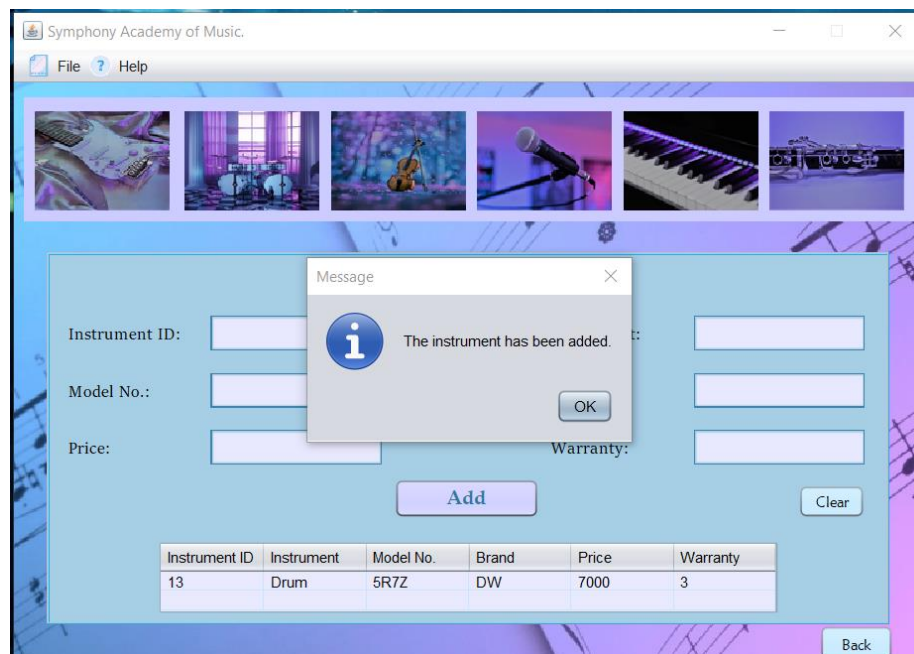
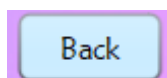
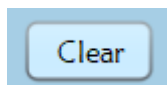


Figure 41: Admin: Instrument Added.



The admin can go back to the admin login and the home page using the back button.



One can also clear all the text fields using the clear button.

Member:

If the user is an already registered member click on the “Member” button.

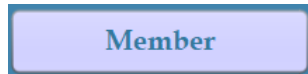


Figure 42: Member Button.

Figure 43: Member Login Details.

Once the button named member is clicked, Figure 43 in the right-hand side given above will be shown. Now, the member has to enter the username and password in the respective fields and click on the login button. After the login is successful, they will be able to see the Figure 8 given down below.

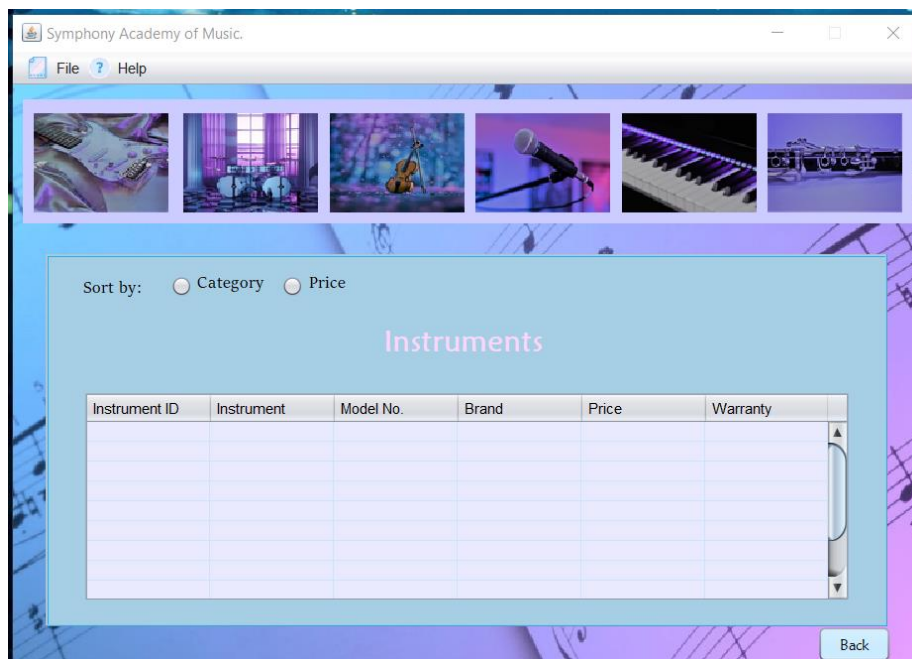


Figure 44: Member: Instruments.

As one can see in the above figure, the table of instruments is empty. The member has to now click on the file menu in the menu bar, hover over the open, and import the instruments. Once, the “Import Instruments” is clicked, the table will be filled with all the instruments available in the academy.

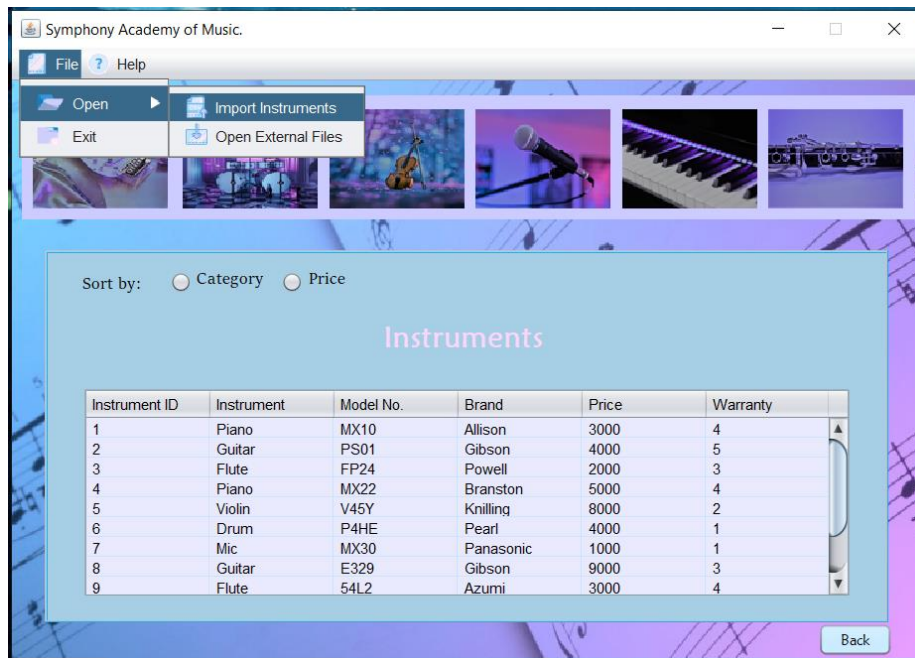


Figure 45: Member: Instruments Part 2.

Sort by: ☒ Category ☐ Price

The member can now sort out the instruments by either category or price.

Category Search:

When the category radio button is clicked, the table is sorted by the instrument's category, and the member can also search for a specific instrument using the drop-down box and the search button. When a specific instrument is searched a message box pops up showing the model number of the instruments available which can be seen in figure 47.

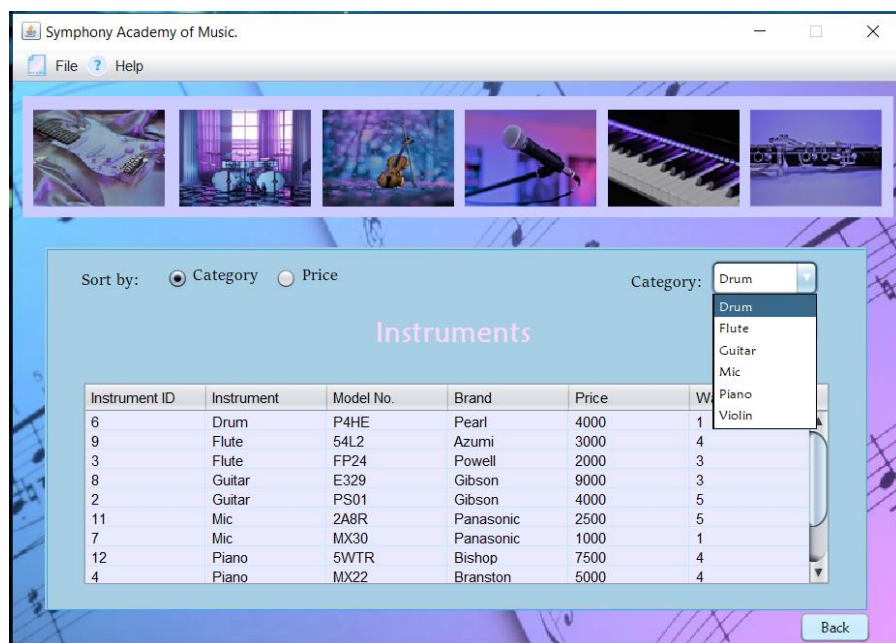


Figure 46: Member: Instruments: Sort By: Category.

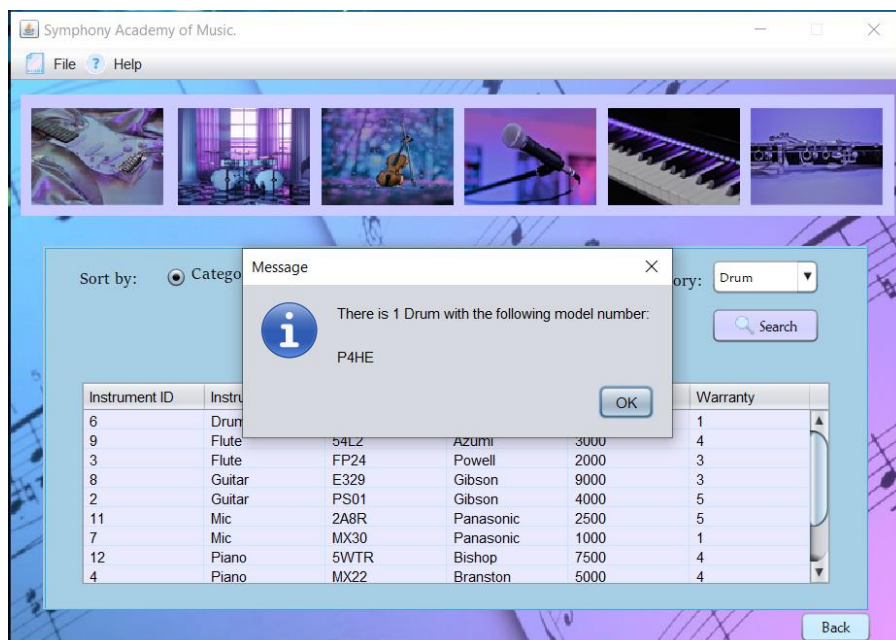


Figure 47: Member: Instruments: Specific Category Search.

Price Search:

When the price radio button is clicked, the table is sorted by the instrument's price, and the member can also search for a specific price using the text field and the search button.

When a specific price is searched a message box pops up showing the instrument details available which can be seen in figure 49.



Figure 48: Member: Instruments: Sort By: Price.

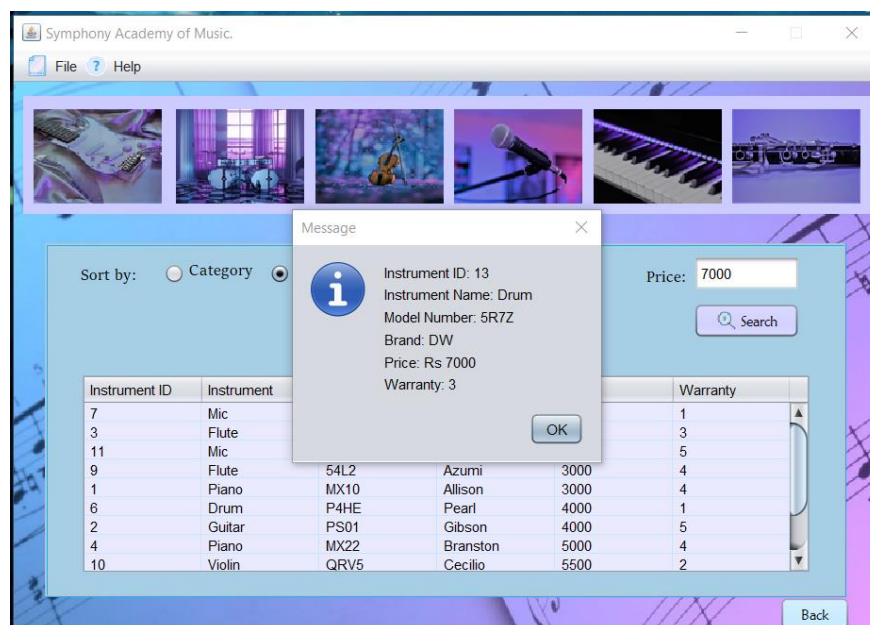
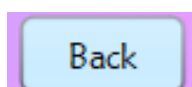


Figure 49: Member: Instruments: Specific Price Search.



The member can also go back to the member login and the home page using the back button.

Register:

If the user does not have an account, click on the register button to register and create an account.

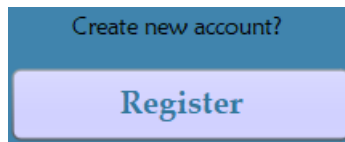


Figure 50: Register Button.

Registration

Name: Username:

Specialization: Password:

Age: Re-type Password:

Name	Specialization	Age	Username	Password

Figure 51: Registration.

Once the button named register is clicked. Figure 51 will be shown in which the new user has to fill in the details. After all the details are filled, one must click on the submit button to successfully complete the registration. If the registration is successful, a message box will pop up to notify indicating that the registration is complete and the details of the registration will be shown in the table. Do not worry as the registration detail will only be visible in the user's device.

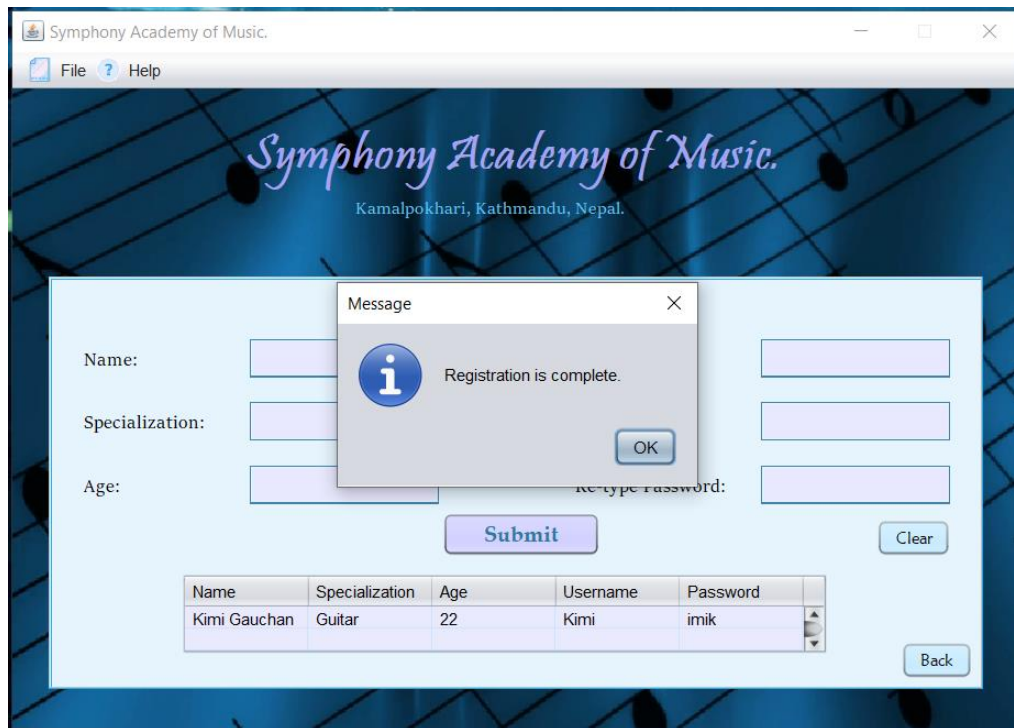
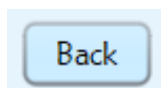
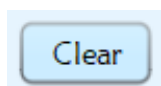


Figure 52: Registration Complete.



The user can go back to the home page using the back button.



One can also clear all the text fields using the clear button.

External Files:

The user can open external files through this application.



Figure 53: Open External Files.

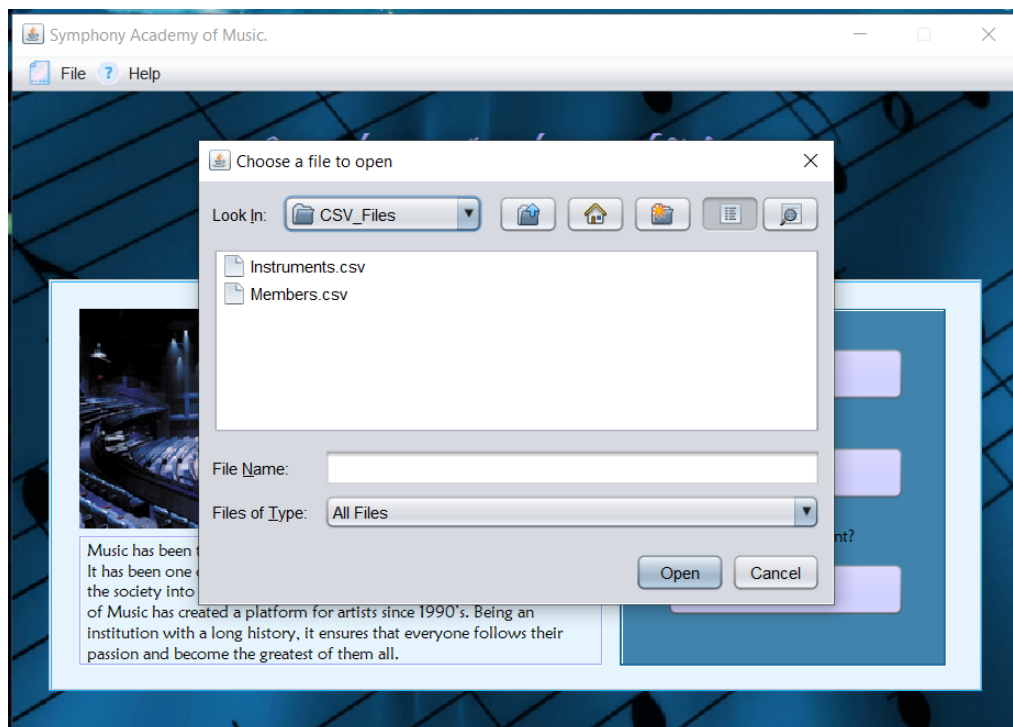


Figure 54: Opening External Files.