



**slington college**  
(इस्लिङ्टन कलेज)

**Module Code & Module Title**

**CS5004NA Emerging Programming Platforms and Technologies**

**Assessment Weightage & Type**

**30% Group Coursework**

**Year and Semester**

**2019-20 Autumn**

Group Name:			
SN	Student Name	College ID	University ID
1	Urusha Shrestha	NP01CP4S190042	18030821
2	Punam Shrestha	NP01CP4S190053	18030833
3	Trishna BC	NP01CP4S190037	18030816

*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 mark of zero will be awarded.*

## Table of Contents

Task Division .....	1
1. Introduction .....	2
2. Body .....	3
2.1. Selection sort .....	3
2.2. Linear search .....	5
2.3. Binary search .....	7
2.4. Menu bar .....	10
2.5. Method Description: .....	11
2.6. Wireframe .....	13
3. Testing .....	14
3.1 Test 1 .....	14
3.2 Test 2 .....	15
3.3 Test 3 .....	17
3.4 Test 4 .....	19
3.5 Test 5 .....	22
3.6 Test 6 .....	24
3.7 Test 7 .....	25
3.8 Test 8 .....	26
3.9 Test 9 .....	27
3.10 Error Correction .....	29
3.11 Test 11 .....	30
4. Conclusion .....	31
Bibliography .....	32

## List of figures

Figure 1:Example for selection sort.....	3
Figure 2:Example for Linear Search.....	5
Figure 3:Example for binary search .....	7
Figure 4: Menu bar .....	10
Figure 5: Wireframe.....	13
Figure 6: Test 1: To check if the program runs in Netbeans.....	14
Figure 7:Test 2.1: To add item details to the table .....	16
Figure 8: Test 2.2: To add item details to the table .....	16
Figure 9: Test 3.1: To search for item based on price. ....	17
Figure 10: Test 3.2: To search for item based on price. ....	18
Figure 11: Test 4.1: To search for item based on category. ....	20
Figure 12: Test 4.2: To search for item based on category. ....	21
Figure 13: Test 4.3: To search for item based on category. ....	21
Figure 14: Test 5.1: To open a file from the menu. ....	22
Figure 15: Test 5.2: To open a file from the menu. ....	23
Figure 16: Test 6: To show dialog box appears when all the fields are left empty. ....	24
Figure 17: Test 7: To show that dialog box appears when any one of the fields is left empty. ....	25
Figure 18: Test 8: To show that dialog box appears when the user enters same Id number twice.....	26
Figure 19: Test 9.1: To show that dialog box appears when the user enters string value in ID number.....	28
Figure 20: Test 9.2: To show that dialog box appears when the user enters string value in ID number.....	28
Figure 21: Error Correction: To show that dialog box appears when the user enters string value in ID number. ....	29
Figure 22 Test 11: To show that dialog box appears when the user enters a string value for Price.....	30

## List of tables

Table 1: Task Division .....	1
Table 2: Test 1: To check if the program runs in Netbeans.....	14
Table 3: Test 2: To add item details to the table .....	15
Table 4: Test 3: To search for item based on price. ....	17
Table 5: Test 4: To search for item based on category. ....	19
Table 6: Test 5: To open a file from the menu.....	22
Table 7: Test 6: To show dialog box appears when all the fields are left empty.....	24
Table 8: Test 7: To show that dialog box appears when any one of the fields is left empty. ....	25
Table 9: Test 8: To show that dialog box appears when the user enters same Id number twice. ....	26
Table 10: Test 9: To show that dialog box appears when the user enters string value in ID number. ....	27
Table 11: Error Correction: To show that dialog box appears when the user enters string value in ID number. ....	29
Table 12: Test 11: To show that dialog box appears when the user enters a string value for Price.....	30

**Task Division**

Members name	Member Performed task
Trishna BC	Code: 1. Validation 2. Data entry in table Report: 1. Introduction 2. Conclusion
Urusha Shrestha	Code: 1. Binary Search 2. Selection sort Report: 1. Testing 2. Algorithm for binary and linear search
Punam Shrestha	Code: 1. GUI 2. Linear Search Report: 1. Selection sort, Binary and linear search explanation 2. Method description

*Table 1: Task Division*

## 1. Introduction

Java Technology is both a programming language and platform. The Java programming language provides a very clean implementation of most of the important object-oriented concepts and serves well as an introductory teaching language (David J. Barnes, 2003). The Java Platform is software only platform which runs on all the OS (Windows, Linux, Solaris, Mac OS) consisting of two components - The Java Virtual Machine (JVM) and The Java Application Programming Interface (API).

A java-swing based application was required to develop in this coursework. The software required to develop the application is **Netbeans**. The group task was assigned where the works were divided within the group members and present it with proper documentation. For this coursework, an application for “**Camera Accessories Details**” was developed for storing and displaying the camera accessories items. The GUI for this application consists of frame, panel, label, table, scroll pane, menu bar, text field, combo box and radio button.

The features of the application consist of a menu bar, tables, data entry and search button. The menu bar has two menus (file – to open file to open the existing file and exit for closing the system, help – opens a help guide in pdf format). Two tables are used, one is the main table and other is the search table. The main table is used for listing all the details of the camera and is updated every time a new data is inserted. The search table is used to display the list of products while searching. Clear button is used to clear the data in text fields. Binary search and linear search are used to search the items according to price and category. Selection sort is used to sort the price in ascending order.

## 2. Body

### 2.1. Selection sort

Selection sort is a simple sorting algorithm. In this sorting algorithm, the array is divided into two parts, the sorted part and the unsorted part. Initially, the sorted part is empty, and the unsorted part is the entire array. It sorts an array by repeatedly finding the smallest element of the unsorted tail region and moving it to the front, to the sorted part. The loop runs till the whole array is sorted. After the smallest value is found and is moved to the sorted part, the unsorted part's bound is increased by one. (tutorialspoint, 2020).

```
arr[] = 64 25 12 22 11

// Find the minimum element in arr[0...4]
// and place it at beginning
11 25 12 22 64

// Find the minimum element in arr[1...4]
// and place it at beginning of arr[1...4]
11 12 25 22 64

// Find the minimum element in arr[2...4]
// and place it at beginning of arr[2...4]
11 12 22 25 64

// Find the minimum element in arr[3...4]
// and place it at beginning of arr[3...4]
11 12 22 25 64
```

(geeksforgeeks, 2020)

*Figure 1: Example for selection sort*

In the above figure, an array is taken whose length is 5. The selection sort algorithm, at first, takes the element at the 0<sup>th</sup> index as a reference element.

Reference element (r)=64

In the first loop, the index of unsorted array starts from 0. The algorithm compares the reference element to all the other elements. If any other element is smaller than the reference element, then the value of reference element is changed to the value of the smaller element.

After comparing the value of the reference element with the value of the 1<sup>st</sup> index, the value of reference element is changed to 25.

Reference element (r)= 25

Now, the algorithm compares the value of reference element with the rest of the array. At the end of the first loop, the value of reference element is 11, which is the smallest number in the array. This value is swapped with the starting index of the unsorted array and the starting index of the unsorted array is increased by 1.

Now, the value in the array are as follows:

11 25 12 22 64

The loop runs again, and the same process is repeated. After each loop, the starting index of unsorted array is increased by one. This process continues until the last element of the whole array is compared or until the unsorted part of the array is empty.

In this project selection sort is used in order to implement binary search. It is used sort the accessories of the camera according to the price. Selection sort help to sort the data of price in ascending order and after sorting it keeps the data in an array list.



## 2.2. Linear search

A linear search is also called sequential search. Linear search is a method for finding an element within a list. It sequentially checks each element of the list until a match is found or the whole list has been searched. For an array of  $n$  elements in linear search the average search visits is  $n/2$  elements and the maximum visits is  $n$ . A linear search locates a value in an array in  $O(n)$  steps (wikipedia, 2020).

```
Input : arr[] = {10, 20, 80, 30, 60, 50,  
                110, 100, 130, 170}
```

```
        x = 110;
```

```
Output : 6
```

```
Element x is present at index 6
```

```
Input : arr[] = {10, 20, 80, 30, 60, 50,  
                110, 100, 130, 170}
```

```
        x = 175;
```

```
Output : -1
```

```
Element x is not present in arr[].
```

(geeksforgeeks, 2020)

*Figure 2:Example for Linear Search*

### Algorithm for Linear Search:

Linear Search (Array A, Value x)

Step 1: Set  $i=1$  and  $n=A.length-1$

Step 2: if  $i > n$  then go to step 7, else go to step 3

Step 3: if  $A[i] = x$  then go to step 6, else go to step 4

Step 4: Set  $i$  to  $i + 1$

Step 5: Go to Step 2

Step 6: Print Element  $x$  Found at index  $i$  and go to step 8

Step 7: Print element not found

Step 8: Exit

In this project linear search is used to search the accessories of camera according to the category. When the user clicks the category button then one Combo box with the list of categories and search button appears. After that when the user selects any category and click the search button then the selected category is compared with the list of values in table and the list of accessories selected according to the categories is shown in the next table at the bottom of the frame.

### 2.3. Binary search

Binary search is an efficient algorithm for finding an item from a sorted list of items. Binary search searches a sorted array by repeatedly dividing it in two halves. Begin with an interval covering the whole array and determining whether the values occurs in the first half or second half then repeating the search in one of the halves. If the value of the search key is less than the item in the middle of the interval, narrow the interval to the lower half. Otherwise narrow it to the upper half. Repeatedly check until the value is found, or the interval is empty (geeksforgeeks, 2020).



(geeksforgeeks, 2020)

Figure 3: Example for binary search

In the above figure showing an example of binary search, an array is taken of length 10. The number to be searched is 23. The lowest and highest index are also set.

The middle element of the array is found at the beginning using the formula:

Middle index = (left index + right index) / 2

Then, the algorithm compares the number to be searched with the middle element. If the number is equal to the middle element, then it displays it and the loop is terminated.

If the number is less than the middle element, then the right side of the array of the middle element is ignored and the process is repeated for the elements present in the left side. Otherwise, the left side is ignored, and the process is repeated for the right side.

Here, the middle element is 16 and number is 23. The number is greater than the middle element, so the algorithm now ignores the left side of the array and moves to the right part. The next array starts from the index (middle index+1). The loop is again run for the second array. This process continues until the required value is found.

**Algorithm for Binary search algorithm:**

Binary search (Array A, Value X)

Step1: Set  $l=0$  and  $r=A.length-1$

Step 2: If  $l \leq r$ , go to step 3, else go to step 12

Step 3:  $m=(l+r)/2$

Step 4: If  $A[m]==X$ , go to step 5, else go to step 7

Step 5: Return  $A[m]$

Step 6: Go to step 13

Step 7: If  $A[m]>X$ , then go to step 8, else go to step 10

Step 8:  $l=m+1$

Step 9: Go to step 2

Step 10:  $r=m-1$

Step 11: Go to step 2

Step 12: Print element not found

Step 13: End

In this project binary search is used to search the accessories of camera according to the price. When the user clicks the price button one combo box with the list of price and search button appears. At first to implement the binary search algorithm the data should be sorted, for sorting the data selection sort method has been used as described above. The selection sort help to sort the above data of price in ascending order and after sorting it keeps the data in array list. After that when the user selects the price and click the search button than the selected price is compared with the list of prices in the table and the list of accessories according to the selected price is shown in the table at the bottom.

## 2.4. Menu bar

The menu bar consists of two menus they are: File and Help. In file there are two menu items: File, and Exit. When Open is clicked, the group report will load in pdf and when Exit is clicked, the system will close. When the Help menu is clicked, the help guide for the user will open in pdf format.

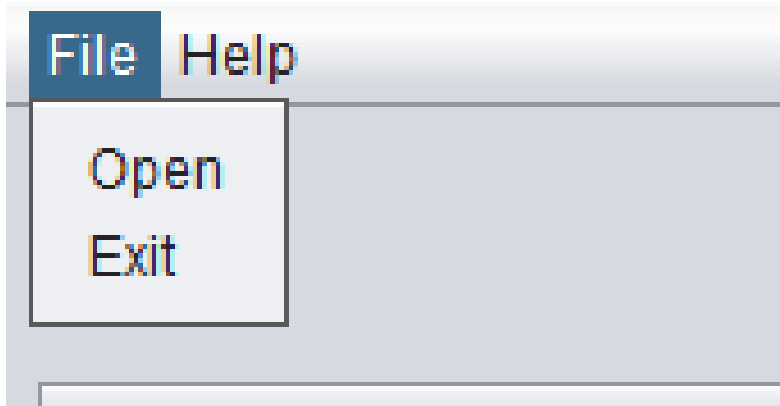


Figure 4: Menu bar

## 2.5. Method Description:

**public static void selectionSort ()** method is a void method which does not return any value. This method is called when an integer array needs to be sorted in ascending order. As the name suggests, it sorts the array using selection sort. In the program, it is called just before applying binary search to sort the array containing different price values.

**public int binarySearch ()** is a method which returns values. This method is called when a specific value in a sorted integer array needs to be searched. As the name suggests, it searches for the specific value in a sorted array using binary search. In the program, it is used to search the value of price according to user input and is called after applying selection sort to sort the array.

**public void jButton1ActionPerformed (java.awt.event.ActionEvent evt)** is a method for the “Add to Table” button. After the button is clicked, this method checks extracts all the values from the textfields, combobox and radiobutton. If all the criteria is met and no field is left empty, the values are added to the table.

**private void jButton4ActionPerformed(java.awt.event.ActionEvent evt)** is a method for “Clear” button. After the button is clicked, all the values in the textfields and are cleared.

**private void jButton2ActionPerformed(java.awt.event.ActionEvent evt)** is a method for “Price” button in the filter section of the application. After clicking the button, the combobox and search button for price are set visible.

**private void jButton3ActionPerformed(java.awt.event.ActionEvent evt)** is a method for “Category” button in the filter section of the application. After clicking the button, the combobox and search button for category are set visible.

**private void jButton5ActionPerformed(java.awt.event.ActionEvent evt)** is a method for the “Search” button of price. After clicking the button, the method checks if the user has selected any value from the combobox. If the user has not selected any value, then it displays a message stating to select a value. If the user has selected a value, it first calls the selection sort method to sort the price and then calls the binary search method. When the selected value is found, it is displayed in the “Search Results” table.

**private void jButton6ActionPerformed(java.awt.event.ActionEvent evt)** is a method for the “Search” button of category. After clicking the button, the method checks if the user has selected any value from the combobox. If the user has not selected any value, then it displays a message stating to select a value. If the user has selected a value, it first adds all the data from the main table into a 2D array. Then it iterated through the category column in the array to search for the selected value and displays in the “Search Results” table.

**private void jButton7ActionPerformed(java.awt.event.ActionEvent evt)** is a method for the “None” button in the filter section. After the button is clicked, the “Search Result” table is cleared.

**private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt)** is a method for the “Exit” menu item. It exits from the application.

**private void jMenuItem2MouseClicked(java.awt.event.MouseEvent evt)** is a method for the “File” menu item. When it is clicked a pdf file opens.

**private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt)** is a method for “Help” menu. When the button is clicked a help guide in pdf format is opened.



## 2.6. Wireframe

File

Help

### Camera Details

ID	Name	Brand	Category	Lens	Price

#### Filter

Search according to

Price

Category

None

Price 

ComboBox

Search

Category 

ComboBox

Search

#### New Entry

ID

Name

Brand 

ComboBox

Category 

☐ Radio Button ☐ Radio Button ☐ Radio Button

☐ Radio Button ☐ Radio Button

Lens

Price

Add to Table

Clear

### Search Results

ID	Name	Brand	Category	Lens	Price

Figure 5: Wireframe

### 3. Testing

#### 3.1 Test 1

Test Number	1
Objective	To check if the program runs in NetBeans
Action Performed	The program was run in NetBeans.
Expected Result	A fully functioning GUI should be opened.
Actual Result	A fully functioning GUI was opened.
Conclusion	Test was successful.

Table 2: Test 1: To check if the program runs in Netbeans.

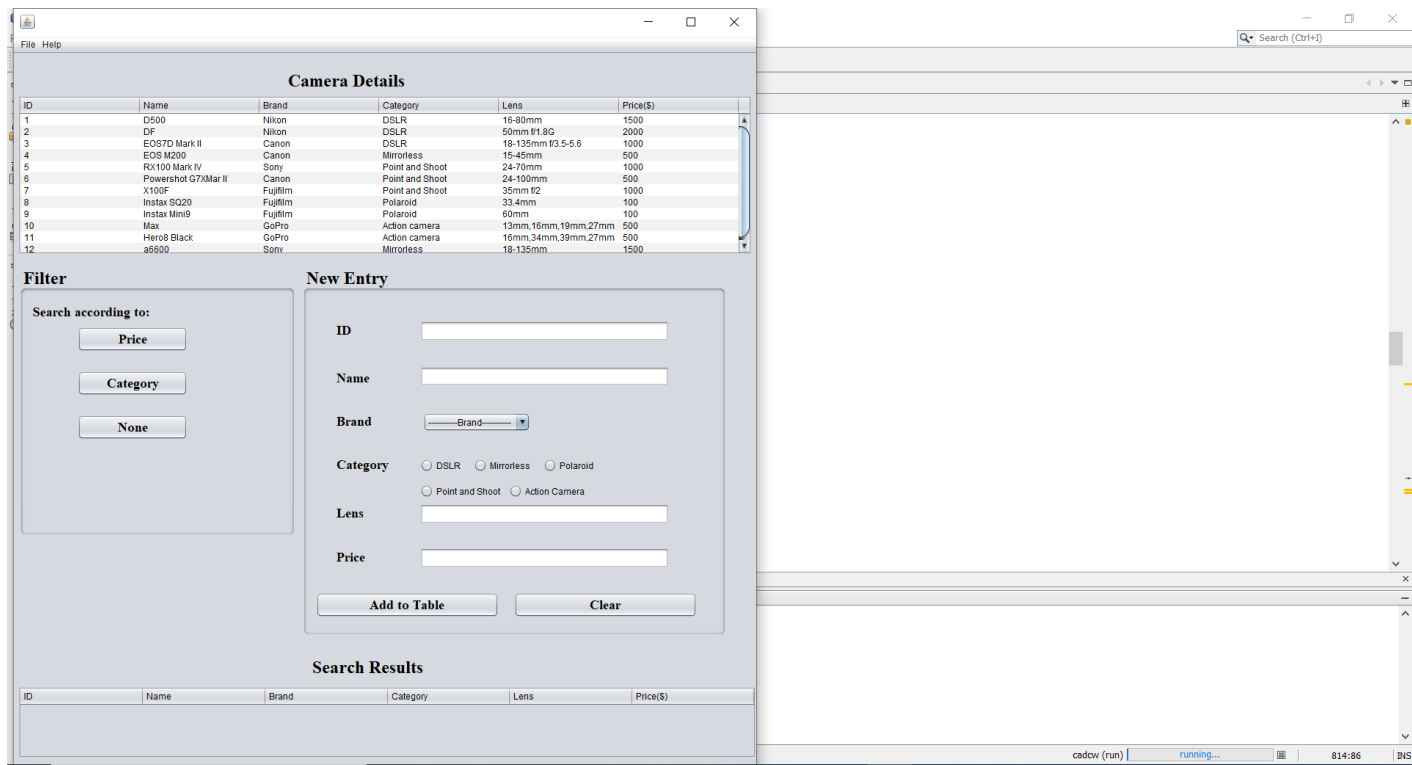


Figure 6: Test 1: To check if the program runs in Netbeans.

**3.2 Test 2**

Test Number	2
Objective	To add item details to the table
Action Performed	All the required fields for the entry of data were filled with appropriate data and “Add to Table” button was clicked.
Expected Result	A new row will be created in the table where the new data will be inserted.
Actual Result	A new row was created in the table and the data was inserted.
Conclusion	Test was successful.

*Table 3: Test 2: To add item details to the table*

**Camera Details**

ID	Name	Brand	Category	Lens	Price(\$)
1	D500	Nikon	DSLR	16-80mm	1500
2	DF	Nikon	DSLR	50mm f/1.8G	2000
3	EOS7D Mark II	Canon	DSLR	18-135mm f/3.5-5.6	1000
4	EOS M200	Canon	Mirrorless	15-45mm	500
5	RX100 Mark IV	Sony	Point and Shoot	24-70mm	1000
6	Powershot G7XMar II	Canon	Point and Shoot	24-100mm	500
7	X100F	Fujifilm	Point and Shoot	35mm f/2	1000
8	Instax SQ20	Fujifilm	Polaroid	33.4mm	100
9	Instax Mini9	Fujifilm	Polaroid	60mm	100
10	Max	GoPro	Action camera	13mm,16mm,19mm,27mm	500
11	Hero8 Black	GoPro	Action camera	16mm,34mm,39mm,27mm	500
12	a6600	Sony	Mirrorless	18-135mm	1500

**Filter**

Search according to:

**New Entry**

ID:

Name:

Brand:

Category: ☐ DSLR ☒ Mirrorless ☐ Polaroid

☐ Point and Shoot ☐ Action Camera

Lens:

Price:

**Search Results**

ID	Name	Brand	Category	Lens	Price(\$)
----	------	-------	----------	------	-----------

Figure 7: Test 2.1: To add item details to the table

**Camera Details**

ID	Name	Brand	Category	Lens	Price(\$)
2	DF	Nikon	DSLR	50mm f/1.8G	2000
3	EOS7D Mark II	Canon	DSLR	18-135mm f/3.5-5.6	1000
4	EOS M200	Canon	Mirrorless	15-45mm	500
5	RX100 Mark IV	Sony	Point and Shoot	24-70mm	1000
6	Powershot G7XMar II	Canon	Point and Shoot	24-100mm	500
7	X100F	Fujifilm	Point and Shoot	35mm f/2	1000
8	Instax SQ20	Fujifilm	Polaroid	33.4mm	100
9	Instax Mini9	Fujifilm	Polaroid	60mm	100
10	Max	GoPro	Action camera	13mm,16mm,19mm,27mm	500
11	Hero8 Black	GoPro	Action camera	16mm,34mm,39mm,27mm	500
12	a6600	Sony	Mirrorless	18-135mm	1500
13	D800	Canon	Mirrorless	35mm	500

**Filter**

**New Entry**

ID:

Name:

Brand:

Category: ☐ DSLR ☐ Mirrorless ☐ Polaroid

☐ Point and Shoot ☐ Action Camera

Lens:

Price:

**Search Results**

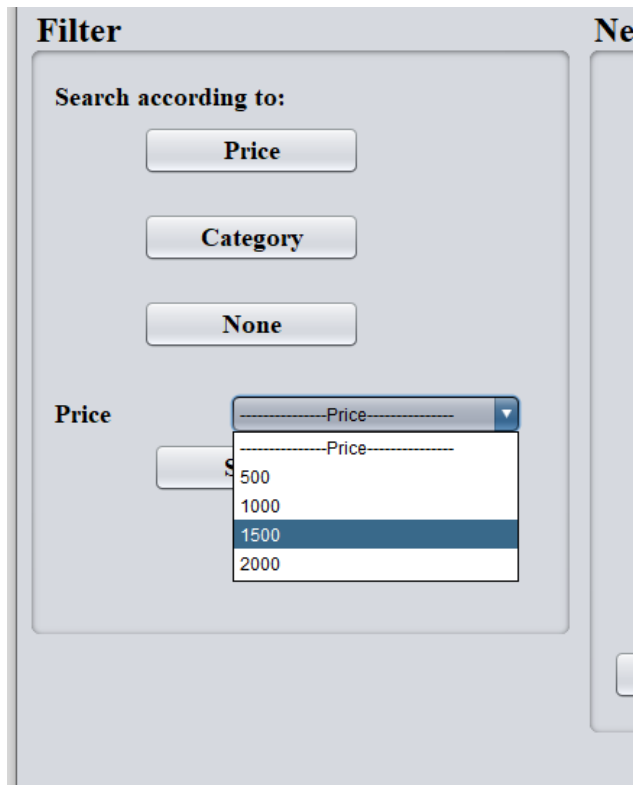
ID	Name	Brand	Category	Lens	Price(\$)
----	------	-------	----------	------	-----------

Figure 8: Test 2.2: To add item details to the table

### 3.3 Test 3

Test Number	3
Objective	To search for item based on price.
Action Performed	The “Price” button was clicked, and the price was selected from the combo box and “Search” button was clicked.
Expected Result	The data with the selected price will be shown in the “Search Results” table.
Actual Result	The data with the selected price was shown in the “Search Results” table.
Conclusion	The test was successful.

Table 4: Test 3: To search for item based on price.



The screenshot shows a web application interface with a 'Filter' section. It contains three buttons: 'Price', 'Category', and 'None'. The 'Price' button is selected. Below it, a dropdown menu is open, showing a list of prices: 500, 1000, 1500 (highlighted), and 2000. The dropdown menu is titled 'Price' and has a search bar with the text 'Price'.

Figure 9: Test 3.1: To search for item based on price.

Search Results					
ID	Name	Brand	Category	Lens	Price(\$)
1	D500	Nikon	DSLR	16-80mm	1500
12	a6600	Sony	Mirrorless	18-135mm	1500

Figure 10: Test 3.2: To search for item based on price.

**3.4 Test 4**

Test Number	4
Objective	To search for item based on category.
Action Performed	The “Category” button was clicked, and the category was selected from the combo box and “Search” button was clicked.
Expected Result	The data with the selected category will be shown in the “Search Results” table.
Actual Result	The data with the selected category was shown in the “Search Results” table.
Conclusion	The test was successful.

*Table 5: Test 4: To search for item based on category.*

**Camera Details**

ID	Name	Brand	Category	Lens	Price(\$)
2	DF	Nikon	DSLR	50mm f/1.8G	2000
3	EOS7D Mark II	Canon	DSLR	18-135mm f/3.5-5.6	1000
4	EOS M200	Canon	Mirrorless	15-45mm	500
5	RX100 Mark IV	Sony	Point and Shoot	24-70mm	1000
6	Powershot G7XMar II	Canon	Point and Shoot	24-100mm	500
7	X100F	Fujifilm	Point and Shoot	35mm f/2	1000
8	Instax SQ20	Fujifilm	Polaroid	33.4mm	100
9	Instax Mini9	Fujifilm	Polaroid	60mm	100
10	Max	GoPro	Action camera	13mm,16mm,19mm,27mm	500
11	Hero8 Black	GoPro	Action camera	16mm,34mm,39mm,27mm	500
12	a6600	Sony	Mirrorless	18-135mm	1500
13	D800	Canon	Mirrorless	35mm	500

**Filter**

Search according to:

Category:

**New Entry**

ID:

Name:

Brand:

Category: ☐ DSLR ☐ Mirrorless ☐ Polaroid  
☐ Point and Shoot ☐ Action Camera

Lens:

Price:

**Search Results**

Figure 11: Test 4.1: To search for item based on category.



The image shows a 'Filter' panel with a section titled 'Search according to:'. It contains three buttons: 'Price', 'Category', and 'None'. Below this, there is a 'Category' label and a dropdown menu. The dropdown menu is open, showing a list of categories: 'DSLR', 'Mirrorless', 'Point and Shoot' (which is highlighted), 'Polaroid', and 'Action camera'. The dropdown menu has a search bar with the text 'S' and a 'Category' label.

Figure 12: Test 4.2: To search for item based on category.

Search Results					
ID	Name	Brand	Category	Lens	Price(\$)
5	RX100 Mark IV	Sony	Point and Shoot	24-70mm	1000
6	Powershot G7XMar II	Canon	Point and Shoot	24-100mm	500
7	X100F	Fujifilm	Point and Shoot	35mm f/2	1000

Figure 13: Test 4.3: To search for item based on category.

### 3.5 Test 5

Test Number	5
Objective	To open a file from the menu.
Action Performed	The file menu was clicked, and the Open button was clicked.
Expected Result	A pdf file will be opened.
Actual Result	A pdf file was opened.
Conclusion	Test was successful.

Table 6: Test 5: To open a file from the menu.

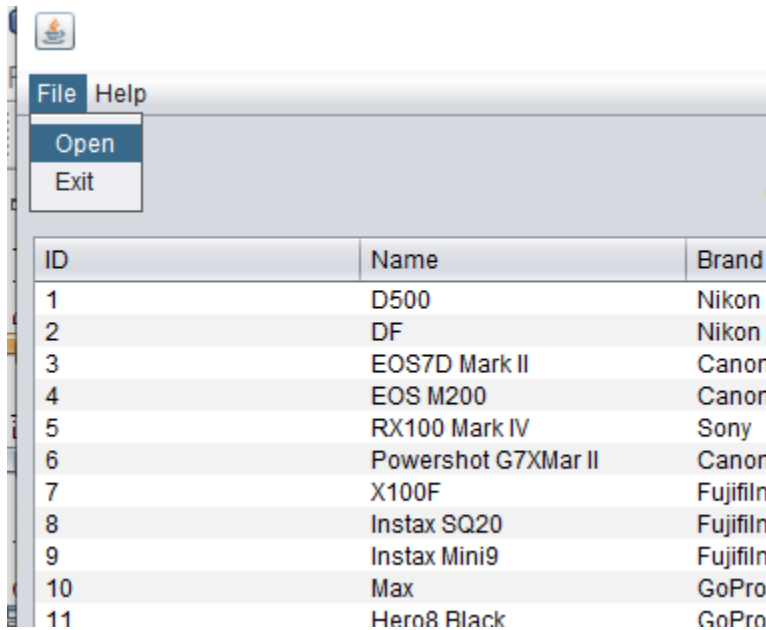


Figure 14: Test 5.1: To open a file from the menu.

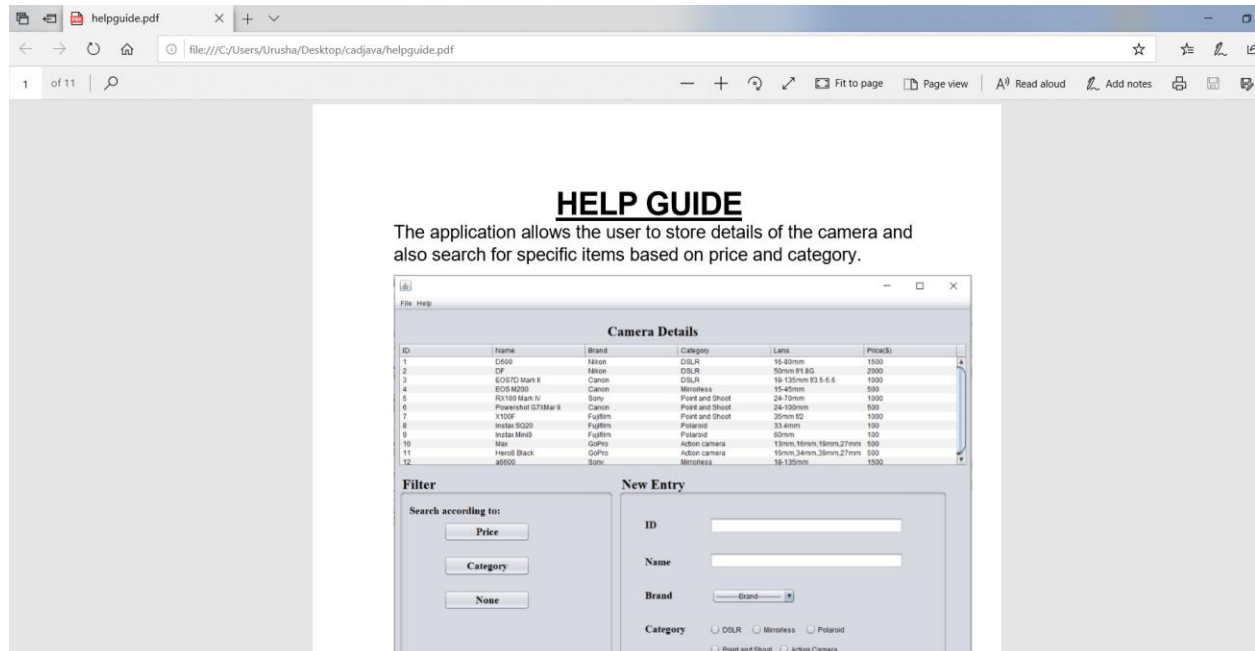


Figure 15: Test 5.2: To open a file from the menu.

### 3.6 Test 6

Test Number	6
Objective	To show dialog box appears when all the fields are left empty.
Action Performed	All the fields were left empty and “Add to Table” button was clicked.
Expected Result	A dialog box will appear stating to fill the required fields.
Actual Result	A dialog box appeared stating to fill the required fields.
Conclusion	The test was successful.

Table 7: Test 6: To show dialog box appears when all the fields are left empty.

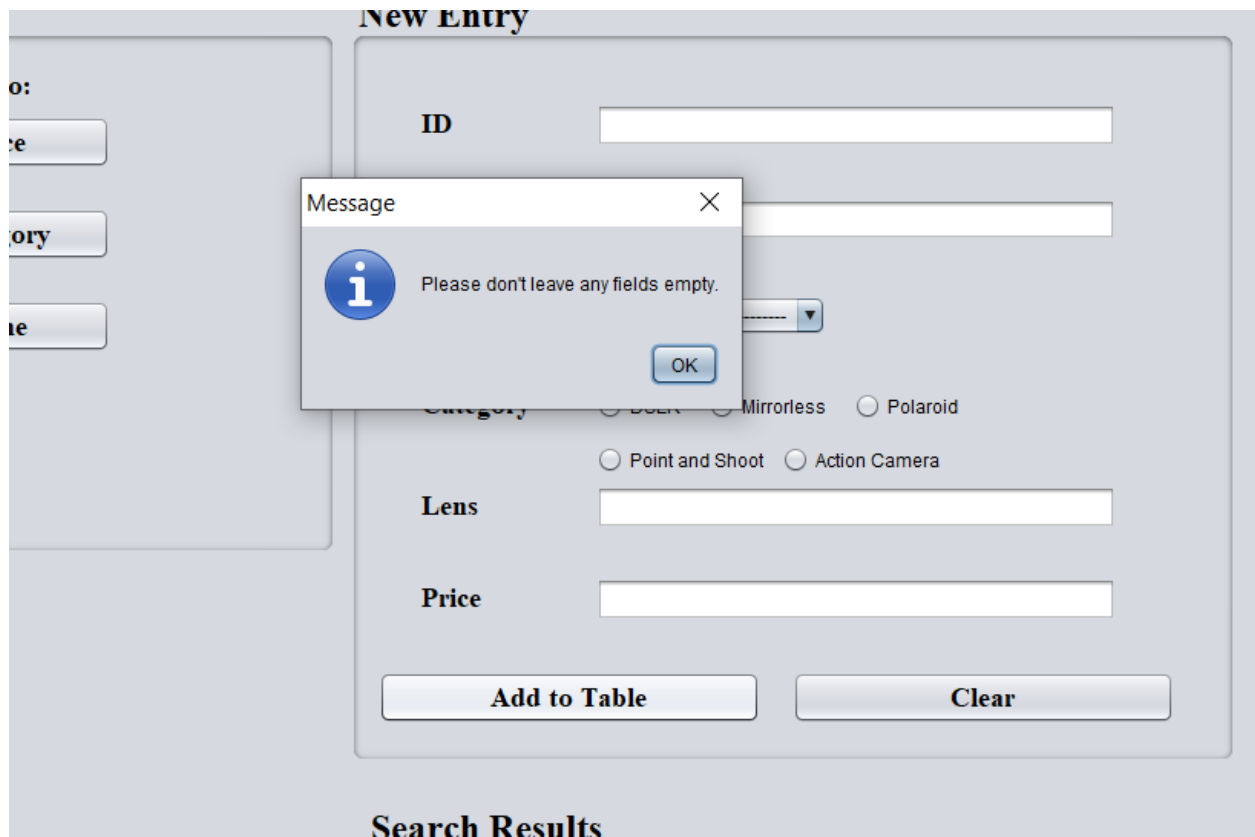


Figure 16: Test 6: To show dialog box appears when all the fields are left empty.

### 3.7 Test 7

Test Number	7
Objective	To show that dialog box appears when any one of the fields is left empty.
Action Performed	All the fields were filled except the “Lens” field and “Add to Table” button was clicked.
Expected Result	A dialog box will appear stating to fill the Lens field.
Actual Result	A dialog box appeared stating to fill the Lens field.
Conclusion	The test was successful.

Table 8: Test 7: To show that dialog box appears when any one of the fields is left empty.

The screenshot shows a web form titled "New Entry" for adding a new camera entry. The form contains several input fields: "ID" (containing "14"), "Category" (a dropdown menu), "Lens" (an empty text field), and "Price" (containing "1000"). There are also radio buttons for "Mirrorless" (selected), "Point and Shoot", and "Action Camera". At the bottom, there are two buttons: "Add to Table" and "Clear". A message dialog box is overlaid on the form, titled "Message" with a close button (X). It contains an information icon (i) and the text "Please enter the Lens". An "OK" button is at the bottom of the dialog box.

Figure 17: Test 7: To show that dialog box appears when any one of the fields is left empty.

### 3.8 Test 8

Test Number	8
Objective	To show that dialog box appears when the user enters same Id number twice.
Action Performed	The ID field was inserted with repeated ID number and “Add to Table” button was clicked.
Expected Result	A dialog box will appear stating to enter accurate ID number.
Actual Result	A dialog box appeared stating to enter accurate ID number.
Conclusion	The test was successful.

Table 9: Test 8: To show that dialog box appears when the user enters same Id number twice.

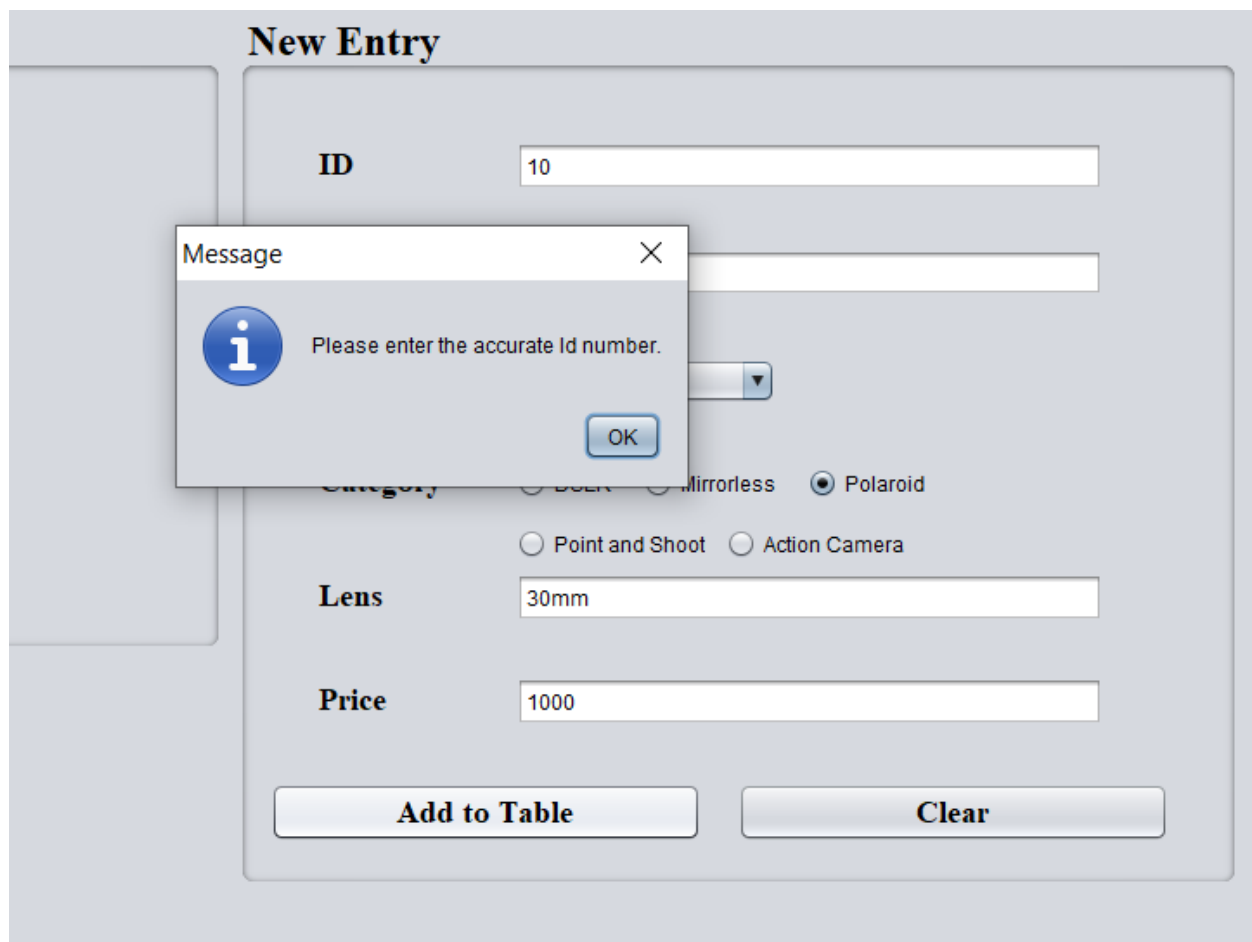


Figure 18: Test 8: To show that dialog box appears when the user enters same Id number twice.

**3.9 Test 9**

Test Number	9
Objective	To show that dialog box appears when the user enters string value in ID number.
Action Performed	String value was inserted in the ID number field and “Add to Table” button was clicked.
Expected Result	A dialog box will appear stating to enter numeric value for ID.
Actual Result	A dialog box did not appear stating to enter a numeric value for ID and data was entered in the table.
Conclusion	The test was unsuccessful.

*Table 10: Test 9: To show that dialog box appears when the user enters string value in ID number.*

File Help

### Camera Details

ID	Name	Brand	Category	Lens	Price(\$)
2	DF	Nikon	DSLR	50mm f/1.8G	2000
3	EOS7D Mark II	Canon	DSLR	18-135mm f/3.5-5.6	1000
4	EOS M200	Canon	Mirrorless	15-45mm	500
5	RX100 Mark IV	Sony	Point and Shoot	24-70mm	1000
6	Powershot G7XMar II	Canon	Point and Shoot	24-100mm	500
7	X100F	Fujifilm	Point and Shoot	35mm f/2	1000
8	Instax SQ20	Fujifilm	Polaroid	33.4mm	100
9	Instax Mini9	Fujifilm	Polaroid	60mm	100
10	Max	GoPro	Action camera	13mm,16mm,19mm,27mm	500
11	Hero8 Black	GoPro	Action camera	16mm,34mm,39mm,27mm	500
12	a6600	Sony	Mirrorless	18-135mm	1500
assddfd	D800	Nikon	Mirrorless	80mm	700

#### Filter

Search according to:

Price

Category

None

#### New Entry

ID:

Name:

Brand:

Category: ☐ DSLR ☒ Mirrorless ☐ Polaroid  
☐ Point and Shoot ☐ Action Camera

Lens:

Price:

Add to Table Clear

### Search Results

Figure 19: Test 9.1: To show that dialog box appears when the user enters string value in ID number.

### Camera Details

ID	Name	Brand	Category	Lens	Price(\$)
3	EOS7D Mark II	Canon	DSLR	18-135mm f/3.5-5.6	1000
4	EOS M200	Canon	Mirrorless	15-45mm	500
5	RX100 Mark IV	Sony	Point and Shoot	24-70mm	1000
6	Powershot G7XMar II	Canon	Point and Shoot	24-100mm	500
7	X100F	Fujifilm	Point and Shoot	35mm f/2	1000
8	Instax SQ20	Fujifilm	Polaroid	33.4mm	100
9	Instax Mini9	Fujifilm	Polaroid	60mm	100
10	Max	GoPro	Action camera	13mm,16mm,19mm,27mm	500
11	Hero8 Black	GoPro	Action camera	16mm,34mm,39mm,27mm	500
12	a6600	Sony	Mirrorless	18-135mm	1500
assddfd	D800	Nikon	Mirrorless	80mm	700
abcd	D600	Canon	Mirrorless	90mm	600

Figure 20: Test 9.2: To show that dialog box appears when the user enters string value in ID number.



### 3.10 Error Correction

Test Number	10
Objective	To show that dialog box appears when the user enters string value in ID number.
Action Performed	String value was inserted in the ID number field and “Add to Table” button was clicked.
Expected Result	A dialog box will appear stating to enter numeric value for ID.
Actual Result	A dialog box appeared stating to enter a numeric value for ID.
Conclusion	The test was successful.

Table 11: Error Correction: To show that dialog box appears when the user enters string value in ID number.

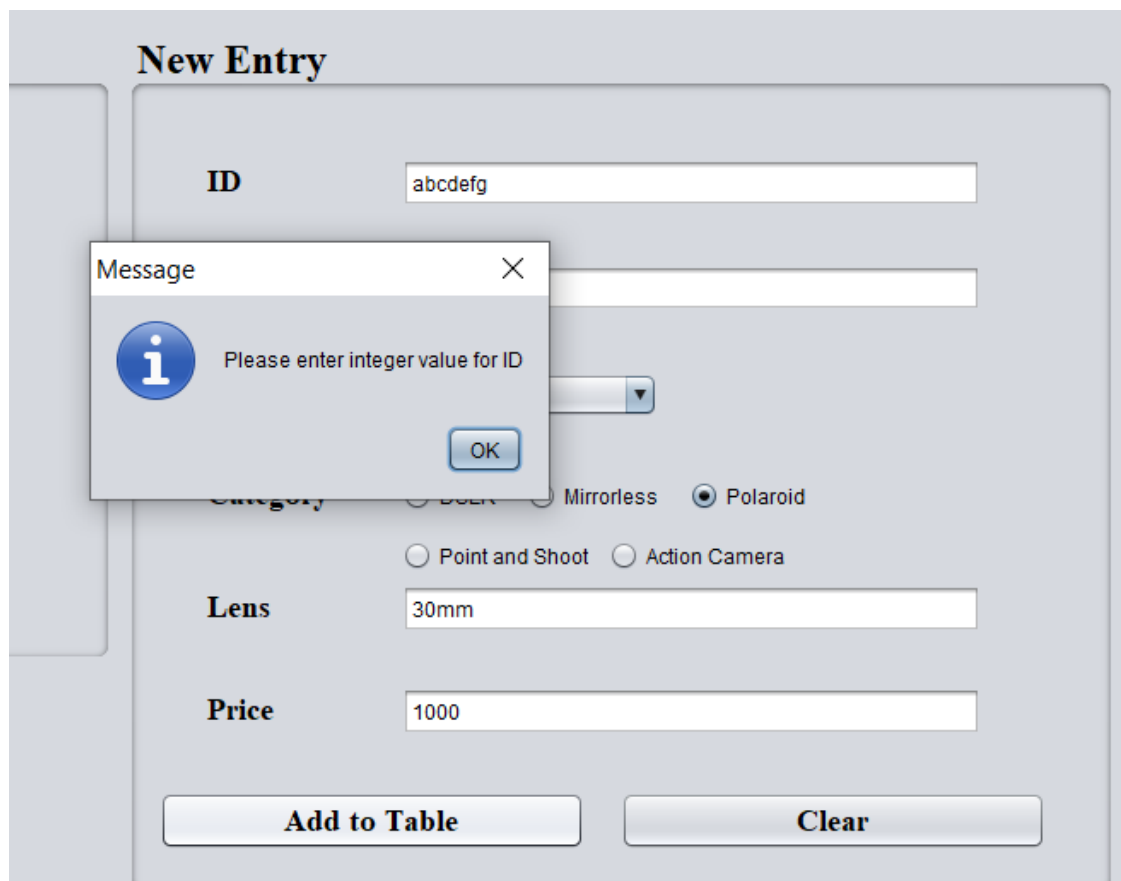


Figure 21: Error Correction: To show that dialog box appears when the user enters string value in ID number.

### 3.11 Test 11

Test Number	11
Objective	To show that dialog box appears when the user enters a string value for Price.
Action Performed	A string value was entered for Price and “Add to Table” button was clicked.
Expected Result	A dialog box will appear stating to enter a numeric value for price.
Actual Result	A dialog box appeared stating to enter a numeric value for price.
Conclusion	The test was successful.

Table 12: Test 11: To show that dialog box appears when the user enters a string value for Price.

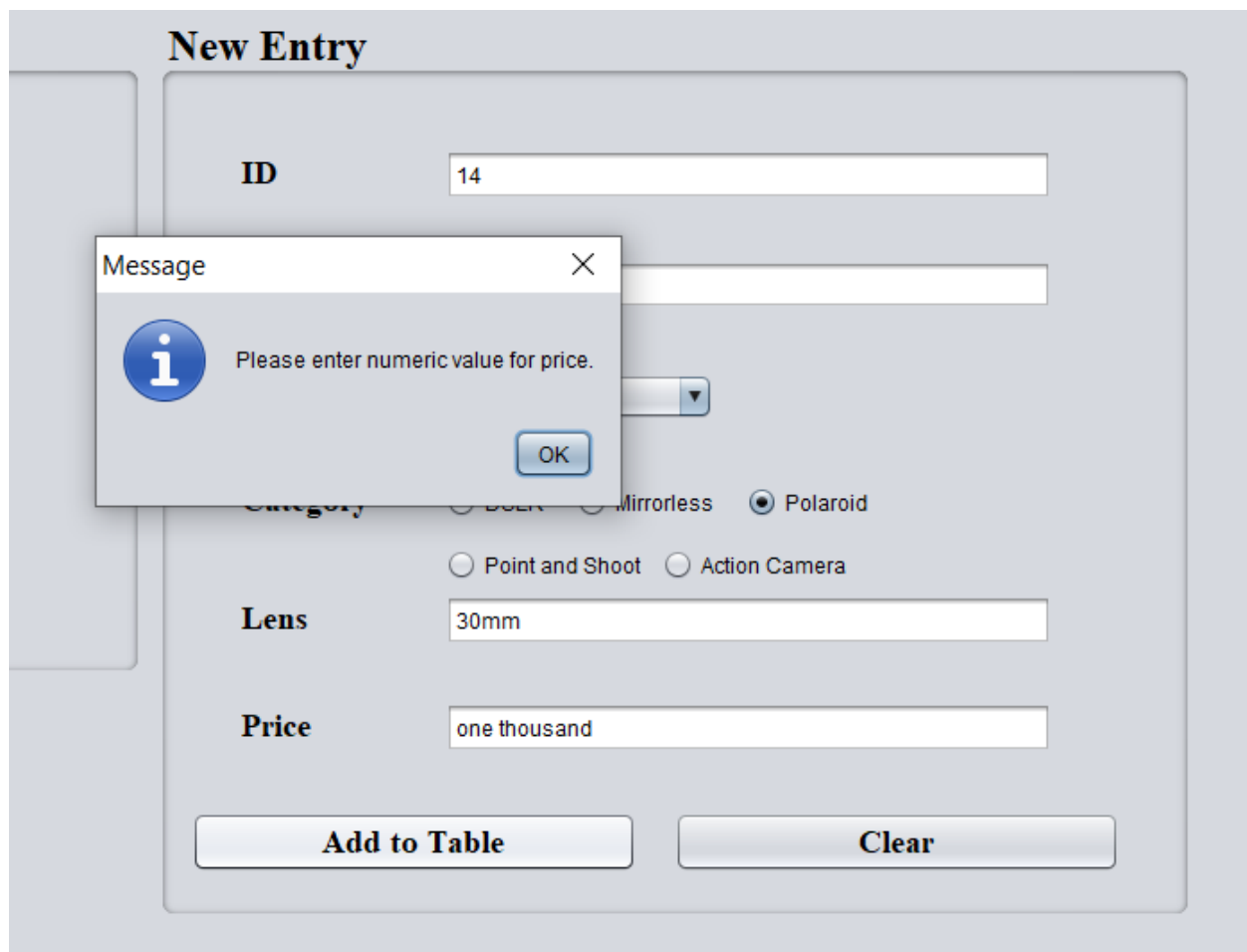


Figure 22 Test 11: To show that dialog box appears when the user enters a string value for Price.

## 4. Conclusion

The main aim of this coursework was to develop an application for sorting and displaying the details of the product. The application has many features and functions, so it was very difficult at first because of some confusion. The works were divided within the members of the group in order to reduce the problems that occur during the coursework & present it with proper documentation and complete it in time. The coursework was able to complete in time because of the group members, help from the module teachers and research on the various related topics.

During this coursework, many confusions occurred regarding searching, sorting and validations. To overcome these confusions, help from module teachers and some research were required. The coursework would not have been completed with only learning through classes, practice and research was required. All the members effort was required in order to complete the coursework. The coursework has helped a lot to gain the ideas and knowledges regarding the relevant topics. This will help in the future to develop other application systems.

## Bibliography

David J. Barnes, M.K. (2003) In David J. Barnes, M.K. *Objects First with Java*. New Jersey: Pearson Education.

geeksforgeeks. (2020) *binary-search* [Online]. Available from:  
<https://www.geeksforgeeks.org/binary-search/>.

geeksforgeeks. (2020) *linear-search* [Online]. Available from:  
<https://www.geeksforgeeks.org/linear-search/>.

geeksforgeeks. (2020) *selection-sort* [Online]. Available from:  
<https://www.geeksforgeeks.org/selection-sort/>.

tutorialspoint. (2020) *linear\_search\_algorithm* [Online]. Available from:  
[https://www.tutorialspoint.com/data\\_structures\\_algorithms/linear\\_search\\_algorithm.htm](https://www.tutorialspoint.com/data_structures_algorithms/linear_search_algorithm.htm).

tutorialspoint. (2020) *selection\_sort\_algorithm* [Online]. Available from:  
[https://www.tutorialspoint.com/data\\_structures\\_algorithms/selection\\_sort\\_algorithm.htm](https://www.tutorialspoint.com/data_structures_algorithms/selection_sort_algorithm.htm).

wikipedia. (2020) *Linear\_search* [Online]. Available from:  
[https://en.wikipedia.org/wiki/Linear\\_search](https://en.wikipedia.org/wiki/Linear_search).