Module Code & Module Title

CS5004NA Emerging Programming Platforms and Technologies

Assessment Weightage & Type

30% Group Coursework

Title: Restaurant Menu Info System

Year and Semester

2019-20 Autumn / 2019-20 Spring

Group Name:			
SN	**Student Name**	**College ID**	**University ID**
1	**Abhishek Gupta**	**NP01CP4A180067**	**18029801**
2	**Animesh Gautam**	**NP01CP4A180083**	**18029830**
3	**Birendra Bista**	**NP01CP4A180076**	**18029818**

1) Proposal

This proposal is written to address the coursework of Emerging Programming Platform and Technologies, given to us as a group task. We have decided to develop a java GUI information system as required by the coursework. We have chosen to make a Restaurant Menu Info System, where the user can order food and drinks according to their need and search for food later specifically also. This application gives the user ability to efficiently order food, drinks and see the prices of the respective food and drinks. The main goal is to make this application look pretty, user-friendly and fun to use. Today’s market requires sophisticated products that can rapidly adapt to changing customer needs. This Project certainly cannot be made by isolated participants; thus, our group members will have some meetings, understand the application and build it together to make it a success.

The point of this project is to outline and make a restaurant menu info system using most of the systems and techniques from the field guaranteeing that no common slip-ups are repeated.

1.1) List of Data:

The Restaurant Menu Info System will have a form to input the restaurant’s menu data and add it to the system. The form will have following type of inputs:

- Dish ID: TextField to input unique ID for each Dish. (Integer)
- Dish Name: TextField to insert the Dish’s name. (String)
- Type: The Type of the Dish selectable from the Combo Box. (String)

- Diet Preference: Radio button to choose the type of dish the user wants to eat. (String)
- Price: TextField to insert the selling price of the dish. (Integer)

The Restaurant Menu Info System will have another form to get personal information from the user once he/she chooses the watch they want to process order for. The form will have following type of inputs:

• First Name: TextField to input First name of the customer. (String)

• Last Name: TextField to input Last name of the customer. (String)

• Total Amount: TextField to input total amount for the customer's order. (Double)

1.2) List of features:

• This simple software will have the capability of adding new dishes to the system by giving the specific information via the user's input.

• The saved data will be shown in Table and is changed every time a new dish is added.

• Dishes can be searched according to the price as well as by the type of the dish.

• The table will be made with JTable from Swing Package.

• Users can place order and receive a bill of what they ate.

• The bill will be similar as that of an actual restaurant.

• There will also be a menu bar on top.

• Users can open existing data by clicking open in the menu bar.

• Users can also exit the system by clicking the exit option from the menu bar.

• Users can get help on using the system by clicking the help option in the menu bar.

• Each item in the menu will have shortcut keys for ease of access.

1.3) Tools Used:

Here are some of the tools that is to be used for the making and completion of this coursework:

1.3.1) Apache NetBeans IDE 11.2:

NetBeans IDE and NetBeans Platform are based on Apache NetBeans from the Apache Software Foundation. NetBeans IDE is a free and open source integrated development environment for application development and is cross platform. It has many features like to drag and drop whilst making the GUI of an application. It gives us the ability to input long codes with shortcuts. It highlights the source code syntactically and lets us easily restructure our code, with a range of handy and powerful tools. It provides us with many editors to create applications in Java, PHP, HTML, CSS and many other languages.

1.3.2) Balsamiq Mock-ups (For GUI Prototyping):

Balsamiq Mock-ups is a UI wireframing tool which enables us to develop GUI prototypes of our projects. It gives us the option to use different shapes and tools that are required to create a wireframe for our GUI prototype. Balsamiq also gives us the ability to drag and drop our necessary shapes to create the wireframe. It reproduces the experience of sketching on a notepad or whiteboard but using a computer.

1.3.3) Java:

Java Platform lets us develop and deploy Java applications on desktops and servers. Java offers the user interface, performance, versatility, portability, and security that today's applications require. We used Java SE (Standard Edition) 13 and

JDK (Java Development Kit) 13 for developing, testing, prototyping and demonstrating our Java application. Java 13 was used in NetBeans IDE whilst developing our application for our coursework.

Table of Contents

1) Proposal ....

1.1) List of Data ....

1.2) List of features ....

1.3) Tools Used ....

2) Individual Task *1*

3) Introduction *3*

4) Binary Search *5*

4.1) Implementation of Binary Search 8

4.2) Flowchart of Binary Search 9

4.3) Sorting 10

5) Method Description *11*

6) Testing *14*

6.1) Testing in NetBeans: 14

6.2) Opening a CSV file to add data to the table: 15

6.3) Adding an item to the table: 16

6.4) Searching items by Dish type: 17

6.5) Search items by Dish Price: 18

6.6) System Validation and Function of the Program: 19

7) Conclusion: *23*

8) Bibliography *25*

9) Appendix: *26*

Table of Figures

Figure 1: Binary Search 1 6
Figure 2: Binary Search 2 6
Figure 3: Binary Search 3 7
Figure 4: Flowchart of Binary Search 9
Figure 5: Testing 1 14
Figure 6: Testing 2 15
Figure 7: Testing 3 16
Figure 8: Testing 4 17
Figure 9: Testing 5 18
Figure 10: Testing 6 19
Figure 11: Testing 7 20
Figure 12: Testing 9 21
Figure 13: Testing 10 21
Figure 14: Testing 11 22
Figure 15: Testing 12 22

2) Individual Task

The task given to us was not any easy task to do, a lot of dedication research was necessary to complete this task. As being a group task somehow the load was decreased to individual as it would be done by the combine effort, skills of three student. To complete the following project, every group member has contributed equally while designing, developing, testing and documenting the entire process. To complete this project the following steps are essential to be done:

1. Analysing the requirement of the project,
2. Creating a design prototype consisting of GUI elements,
3. Making the components function,
4. Using a data structure to store the data,
5. Sorting the data stored,
6. Implementing Binary Search on the sorted data.

The group consists of three members and the task load was divided equally among three members by the agreeable discussion of group members themselves. The detailed description of the responsibilities of this task have been divided amongst each individual while working on this project is given below in tabular form:

Member Name	Work done
Birendra Bista	Application development part: ❖ Research on the Binary search and made it function in the application. ❖ Made the insert, clear, exit buttons functioning. Documentation part: ❖ Introduction of the coursework. ❖ Description of the functionality of the binary search and flowchart for more detailed information
Animesh Gautam	Application development part: ❖ Made appropriate GUI Design of the application. ❖ Opening the csv file from the menu bar and appending the opened file in the menu table. Documentation part: ❖ Description of each method created in menu Info class ❖ Conclusion of the coursework.
Abhishek Gupta	Application development part: ❖ System Validation. ❖ Sorting of the data. ❖ Array list to store the data. Documentation part: ❖ Proposal of the coursework ❖ Testing of the program with the screenshots for evidence. ❖ Formatting of the document.

3) Introduction

This documentation clarifies the information System developed which deals with a restaurant's menu in which a user can add, store or find the desired dish or drink. This program contains various elements and programming techniques to function properly and is created in a modular manner so that it can be configured or debugged easily. The modularity of this program will allow this program to reuse the same code at multiple instances, which is much more efficient. The program will also store the information for each dish as an object in array lists for storing, sorting and searching. The GUI is created by implementing included java library dealing with all sorts of GUI component called "Java Swing". The selection sorting algorithm is used to sort the data individually while being added to the program. The sorted information will be crucial to search the product with respect to its price. Also, the dishes can be searched by their type.

Object-Oriented Programming is a methodology to design a program using classes and objects. This make the code cleaner, reusable, maintainable and scalable It provides many concepts, such as inheritance, data binding, polymorphism etc. OOP concepts in Java are the main ideas behind Java's OOP. OOP concepts let us create working methods and variables, then re-use all or part of them without compromising security. The main aim of object-oriented programming is to implement real-world entities, for example, object, classes, abstraction, inheritance, polymorphism, etc. It simplifies software development and maintenance by providing some concepts like Object, Class, Inheritance, Polymorphism, Abstraction, Encapsulation (JavaTpoint, 2011-2018)

A graphical user interface (GUI) is an interface through which a user interacts with electronic gadgets such as computers, hand-held gadgets and other machines. This interface uses symbols, menus and other visual pointer (design) representations to show data and related client controls, not at all like text-based interfacing, where information and commands are in content. GUI representations are controlled by an indicating gadget such as a mouse, trackball, stylus, or a finger on a touch screen. The require for GUI got to be clear since the primary human/computer content interface was through console content creation by what is called a prompt (or DOS prompt). Commands were written on a console at the DOS prompt to start responses from a computer. The utilization of these commands and the requirement for correct spelling made an awkward and wasteful interface. (Techopedia, 2020)

The overall GUI of menu information system made as our coursework is interactive to the user. Here proper and understandable icon and button are made so that there would be no problem to the user to use it. The dish details box is shown where dish available in the restaurant are shown and the gif is used to make GUI impressive and the buttons text fields are managed properly. The table box is kept where the dish details are shown in the table format clearly.

4) Binary Search

Binary search is an efficient and fast search algorithm for finding an item from a sorted list of items. It is a divide and conquer algorithm. It works by repeatedly diving in half the portion of the list that could contain the item, until we have narrowed down the possible locations to just one. For this algorithm to work properly, the data collection should be in the sorted form. Begin with an interval covering the whole array. First the particular item is compared to the middle most item of the collection. If they got matched, then the index of item is returned. If the middle item is greater than the item then the item is searched in the sub-array to the left of the middle item else if the middle item is smaller then the item is searched in the sub-array to the right of the middle item. We basically ignore half of the element just after one comparison. (tutorialspoint, 2020)

Generally binary search works as follows:

Let’s consider the particular item as x.

1. Compare x with the middle element.
2. If x matches with middle element, we return the mid index.
3. Else if x is greater than the mid element, then x can only lie in right half subarray after the mid element. So, we recur for right half.
4. Else (x is smaller) recur for the left half.

When binary search is used to perform operations on a sorted set, the number of iterations can always be reduced on the basis of the value that is being searched. Binary Search basically reduces the search space to half at each step. By search space we mean sub-array of given array where the target value is located (if present in the array). It discards one sub-array and continue on second sub-array. This decision of discarding one sub-array is made in just one comparison. Let’s understand this by taking an example.

Let value be the array and target be the particular item searching for where

Value= [2, 3, 5, 7, 8, 10, 12, 15, 18, 20]

Target= 7

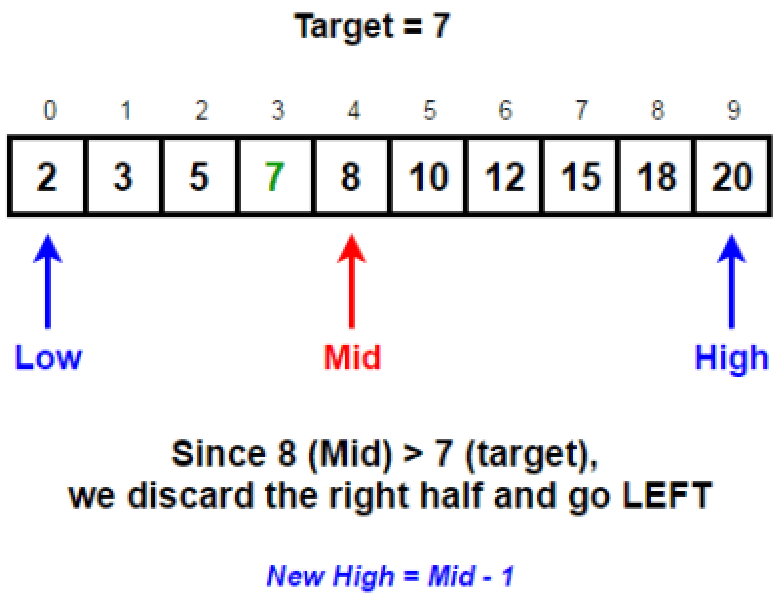


Figure 1: Binary Search 1

First the mid, low and high is determined. The target is compared with the value of mid which is not equal to the target and then compared with value in low and high. After comparing one sub-array is discarded which is the right one and the left one is taken.

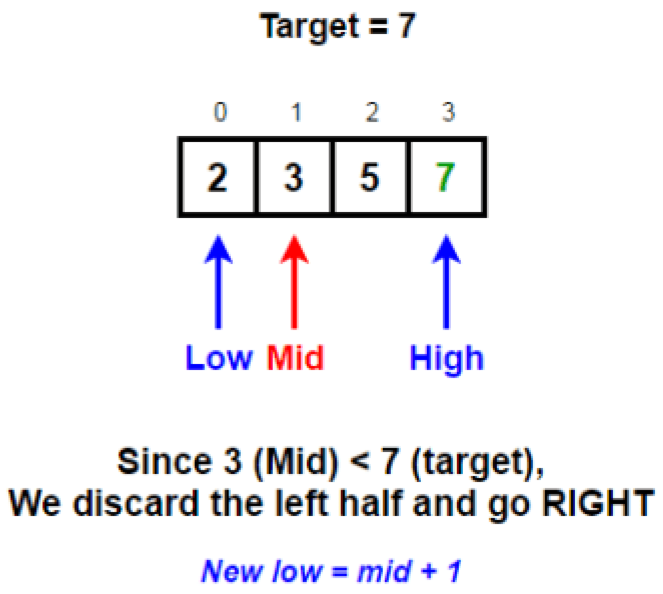

Target = 7
0 1 2 3
2 3 5 7
Low Mid
High
Since 3 (Mid) < 7 (target),
We discard the left half and go RIGHT
New low = mid + 1

Figure 2: Binary Search 2

In the left sub-array again the mid, high and low is determine and same process is followed as explained earlier. Here, left sub-array is discarded and right one is taken.

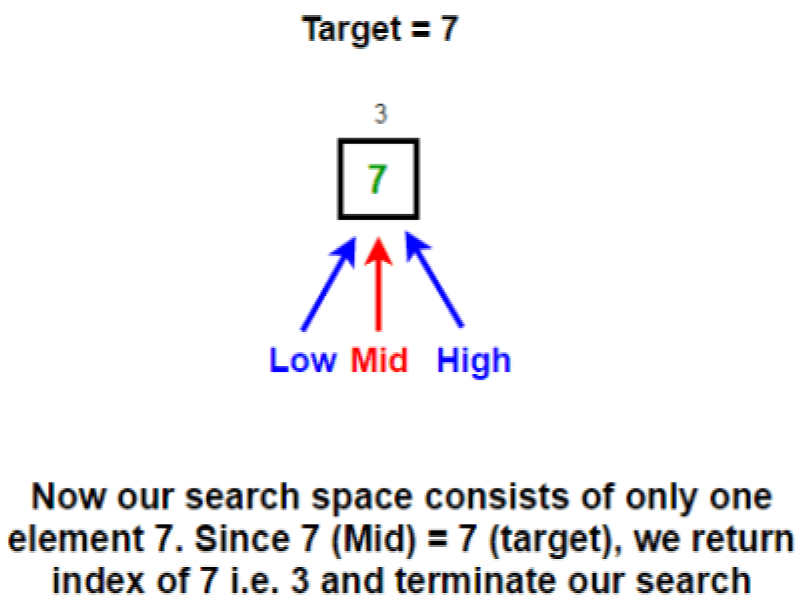

Target = 7
3
7
Low
Mid
High
Now our search space consists of only one element 7. Since 7 (Mid) = 7 (target), we return index of 7 i.e. 3 and terminate our search

Figure 3: Binary Search 3

Finally, one value is left which is the value which after comparing with target is returned and the binary search is terminated.

Read more about binary search here…

4.1) Implementation of Binary Search

In our system the search function is based on the price which is input as integer. To use the principle of binary search in the program to search any dish item according to their price, two sorted array lists were created in which the items were sorted according to their price. In the first array list, price sorted items were stored while the other array list contained only the prices of the items in it. These two array lists have the same items stored within them so that when we take the index from the price array list to the items array list the product is mapped.

Then to search the item according to their price, the price array list is taken with only sorted data in form of an integer and is put through the binary search algorithm along with the search term. The first and last index info is also passed as a parameter so that at the program will know the range in which it will have to search for the term. If the item with that price doesn’t exist, the program will notify the user about the error.

4.2) Flowchart of Binary Search

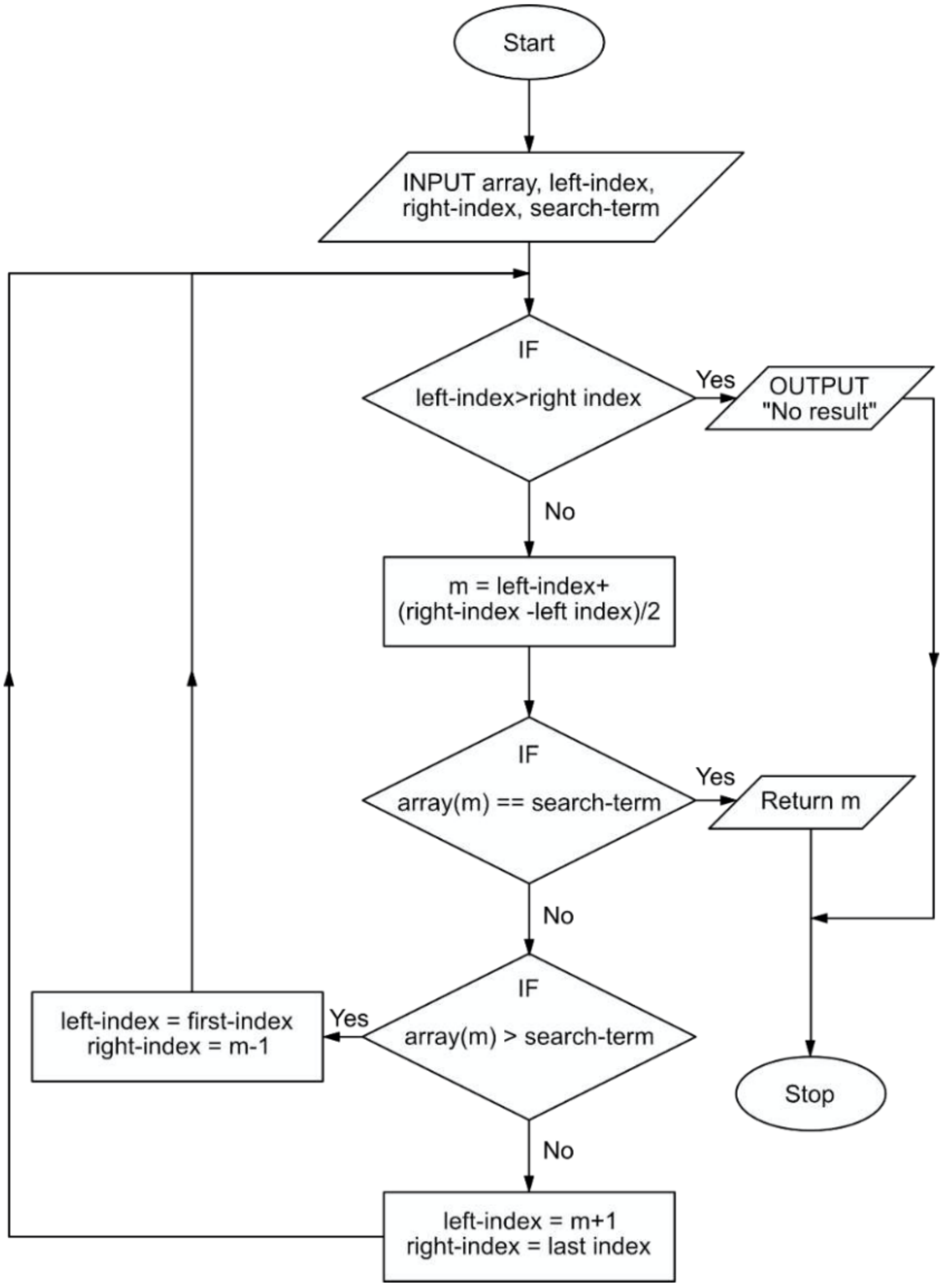
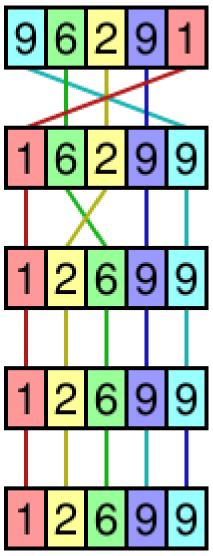

Start
INPUT array, left-index, right-index, search-term
IF left-index>right index
Yes
OUTPUT "No result"
No
m = left-index+ (right-index -left index)/2
IF array(m) == search-term
Yes
Return m
No
IF array(m) > search-term
Yes
left-index = first-index right-index = m-1
No
left-index = m+1 right-index = last index
Stop

Figure 4: Flowchart of Binary Search

4.3) Sorting

Sorting is very necessary for the proper functioning of the program especially for using the binary search algorithm. Binary search functions properly only if the data are sorted. So for sorting the data in the program i.e. price, the selection sort was used. Selection Sort could be a type of sorting calculation. A sorting calculation may be a strategy for reorganizing an expansive number of things into a particular order, such as in sequential order, highest-to-lowest esteem or shortest-to-longest separate. Sorting algorithms take records of things as input information, perform particular operations on those records and convey ordered arrays as output. Applications of sorting algorithms include organizing items by cost on a retail website and determining arrange of sites on a search engine comes about page.



Selection sort is a straightforward sorting calculation. This sorting algorithm is an in-place comparison-based calculation in which the list is separated into two parts, the sorted portion at the left end and the unsorted portion at the right end. At first, the sorted part is empty and the unsorted portion is the whole list. The littlest component is chosen from the unsorted array and swapped with the furthest left component, which component gets to be a portion of the sorted array. This process continues moving unsorted array boundary by one element to the right.

Sorting Algorithm

Step 1: Iterating over unsorted array until the minimum value is found.

Step 2: Minimum value is found after reaching to the end.

Step 3: Swap the minimum element and first element in the unsorted array.

Step 4: First element is considered as sorted element.

Step 5: The same steps are repeated until the whole array is sorted.

(Sehgal, 2018)

5) Method Description

Below are the method descriptions of each of the used in the main method MenuInfo():

MenuInfo() This method calls other methods named initComponents() loadData().	
initComponents() This method manages all the components of GUI and is auto generated.	void
openData() This method adds 12 food itrems to the system.	void
loadData() This method retrieves all the data stored in csv file and display the data in the table.	void
btnSearchActionPerformed(java.awt.event.ActionEvent evt) This method calls another method named sort() and also search for the dish based on the price entered by the customer and displays the information as a pop up message.	void
sort() This method sorts the data of the table and also calls SelectionSorter method .	void
searchList(int low, int high, int key) This method determines the dish which the customers was searching for.	int
btnInsertActionPerformed(java.awt.event.ActionEvent evt) Method that take user input of the dish, stores it and display it in the table when the insert button is clicked.	void
btnClearActionPerformed(java.awt.event.ActionEvent evt) This method clears all the text fields.	void

btnExitActionPerformed(java.awt.event.ActionEvent evt) This method exits the running program.	void
btnItemTypeAvailableActionPerformed(java.awt.event.ActionEvent evt) This method takes user input from combo box and search for the item and display a pop up message stating the availability of dish and a method availableItem() is also called.	void
availableItem() This method display the food item which the customer was searching for.	void
txtSearchPriceKeyTyped(java.awt.event.KeyEvent evt) This method is the key pressed event method for the price search which controls the key typed by the user when pressing the keys on their keyboard. Here the user can only press the digit buttons, backspace and delete keys.	void
menuItemOpenActionPerformed(java.awt.event.ActionEvent evt) This method calls a method named openData().	void
txtPriceKeyTyped(java.awt.event.KeyEvent evt) This method is the key pressed event method of the price input which controls the key typed by the user when pressing the keys on their keyboard. Here the user can only press the digit buttons, backspace and delete keys.	void
txtDishNameKeyTyped(java.awt.event.KeyEvent evt) This method is the key pressed event method of the dish name input which controls the key typed by the user when pressing the keys on their keyboard. Here the user can only press the alphabet buttons, backspace and delete keys.	void
txtDishNumKeyTyped(java.awt.event.KeyEvent evt)	void

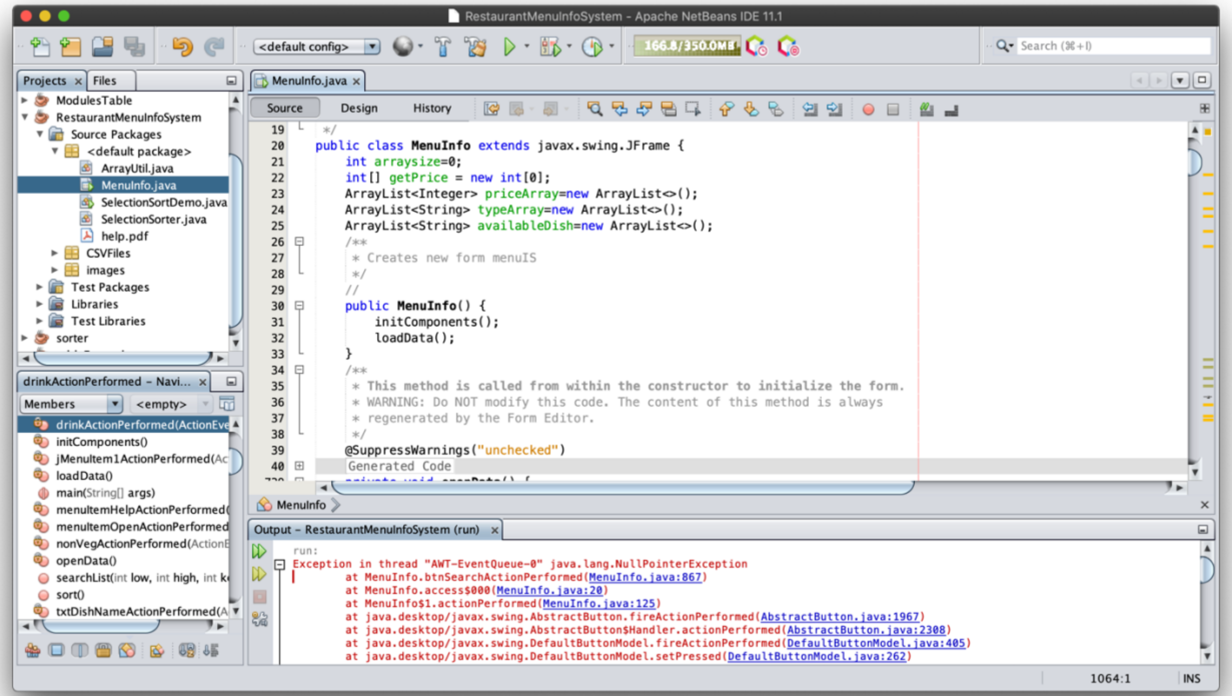
This method is the key pressed event method of the dish number input which controls the key typed by the user when pressing the keys on their keyboard. Here the user can only press the digit buttons, backspace and delete keys.	
menuItemHelpActionPerformed(java.awt.event.ActionEvent evt) This method opens a file which is a usermanual.	void
jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) This method exits the running program.	void
main(String args[]) This is the main method for MenuInfo class.It also calls another function run().	void

6) Testing

6.1) Testing in NetBeans:

Objective	Checking whether the program can be imported and be run from NetBeans or not.
Action	Opening NetBeans and the RestaurantMenuInfoSystem project. Running the MenuInfo class.
Expected Result	The file gets compiled and GUI opens.
Actual Result	File compiled and GUI is opened after giving the path where the file is located.
Conclusion	Test Successful.

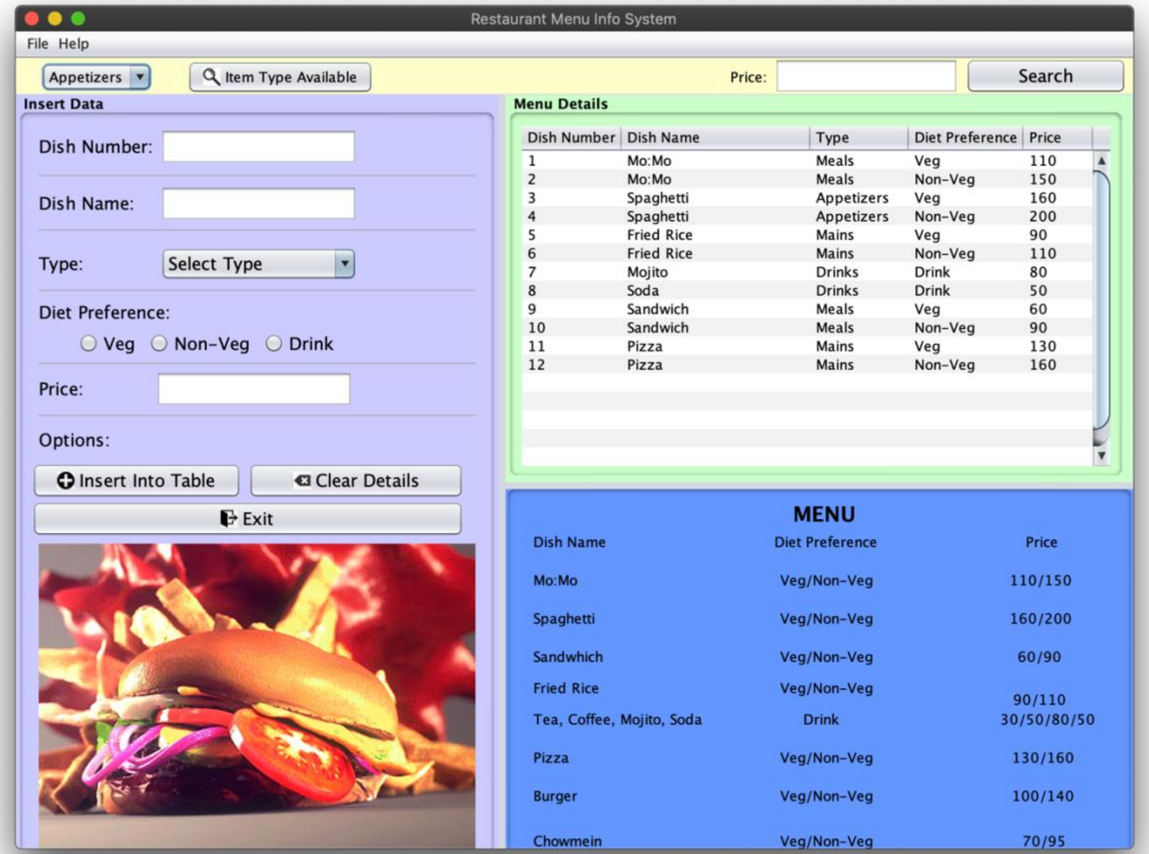
(Note: This project will have different testing themes on different OS as different testing were done on Windows and Mac OS X)



```
 */
public class MenuInfo extends javax.swing.JFrame {
    int arraysize=0;
    int[] getPrice = new int[0];
    ArrayList<Integer> priceArray=new ArrayList<>();
    ArrayList<String> typeArray=new ArrayList<>();
    ArrayList<String> availableDish=new ArrayList<>();
    /**
     * Creates new form menuIS
     */
    //
    public MenuInfo() {
        initComponents();
        loadData();
    }
    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    Generated Code
```

```
run:
Exception in thread "AWT-EventQueue-0" java.lang.NullPointerException
	at MenuInfo.btnSearchActionPerformed(MenuInfo.java:867)
	at MenuInfo.access$000(MenuInfo.java:20)
	at MenuInfo$1.actionPerformed(MenuInfo.java:125)
	at java.desktop/javax.swing.AbstractButton.fireActionPerformed(AbstractButton.java:1967)
	at java.desktop/javax.swing.AbstractButton$Handler.actionPerformed(AbstractButton.java:2308)
	at java.desktop/javax.swing.DefaultButtonModel.fireActionPerformed(DefaultButtonModel.java:405)
	at java.desktop/javax.swing.DefaultButtonModel.setPressed(DefaultButtonModel.java:262)
```

Figure 5: Testing 1



Dish Number	Dish Name	Type	Diet Preference	Price
1	Mo:Mo	Meals	Veg	110
2	Mo:Mo	Meals	Non-Veg	150
3	Spaghetti	Appetizers	Veg	160
4	Spaghetti	Appetizers	Non-Veg	200
5	Fried Rice	Mains	Veg	90
6	Fried Rice	Mains	Non-Veg	110
7	Mojito	Drinks	Drink	80
8	Soda	Drinks	Drink	50
9	Sandwich	Meals	Veg	60
10	Sandwich	Meals	Non-Veg	90
11	Pizza	Mains	Veg	130
12	Pizza	Mains	Non-Veg	160

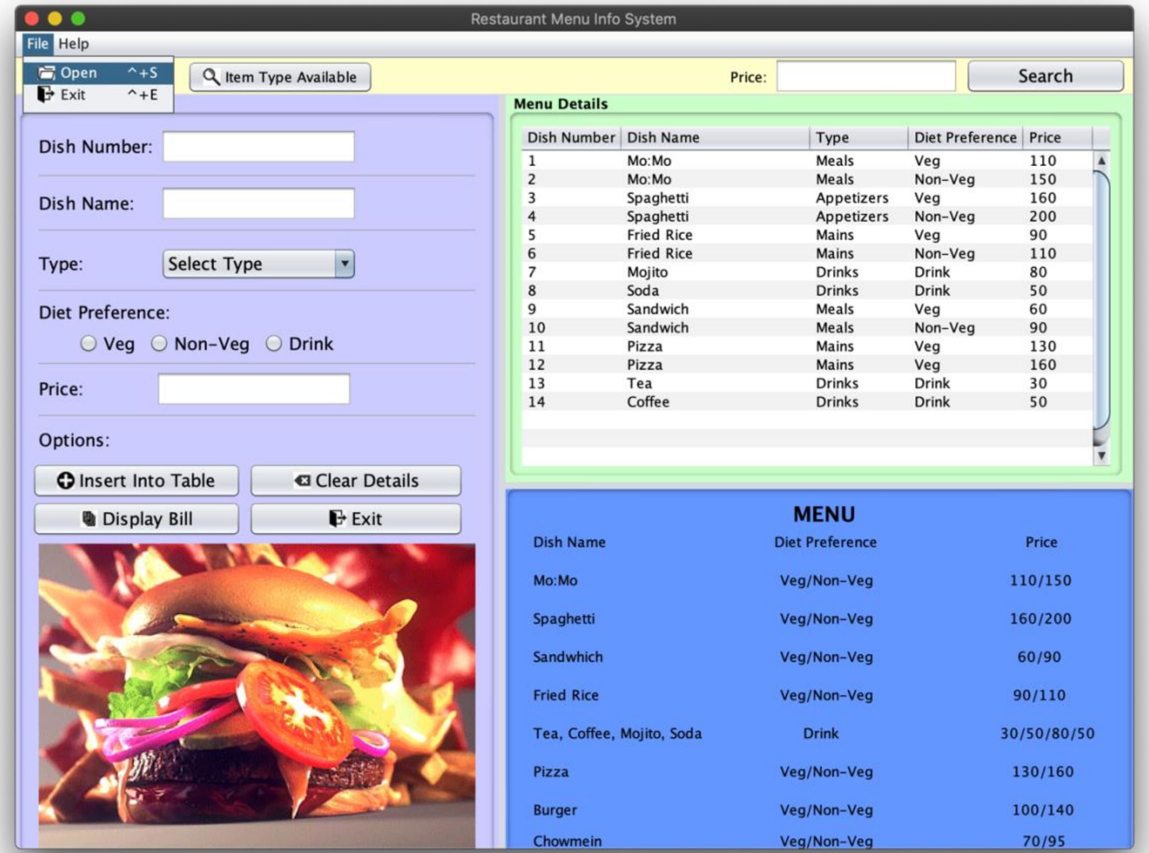
MENU

Dish Name	Diet Preference	Price
Mo:Mo	Veg/Non-Veg	110/150
Spaghetti	Veg/Non-Veg	160/200
Sandwhich	Veg/Non-Veg	60/90
Fried Rice	Veg/Non-Veg	90/110
Tea, Coffee, Mojito, Soda	Drink	30/50/80/50
Pizza	Veg/Non-Veg	130/160
Burger	Veg/Non-Veg	100/140
Chowmein	Veg/Non-Veg	70/95

Figure 6: Testing 2

6.2) Opening a CSV file to add data to the table:

Objective	Checking whether the program adds new items or not after opening the CSV file.
Action	Opening the CSV file from open menu in the menu bar.
Expected Result	The file gets opened and the items will get added. The product is displayed on the table.
Actual Result	The items was added and was displayed at the table.
Conclusion	Test Successful.



Menu Details

Dish Number	Dish Name	Type	Diet Preference	Price
1	Mo:Mo	Meals	Veg	110
2	Mo:Mo	Meals	Non-Veg	150
3	Spaghetti	Appetizers	Veg	160
4	Spaghetti	Appetizers	Non-Veg	200
5	Fried Rice	Mains	Veg	90
6	Fried Rice	Mains	Non-Veg	110
7	Mojito	Drinks	Drink	80
8	Soda	Drinks	Drink	50
9	Sandwich	Meals	Veg	60
10	Sandwich	Meals	Non-Veg	90
11	Pizza	Mains	Veg	130
12	Pizza	Mains	Veg	160
13	Tea	Drinks	Drink	30
14	Coffee	Drinks	Drink	50

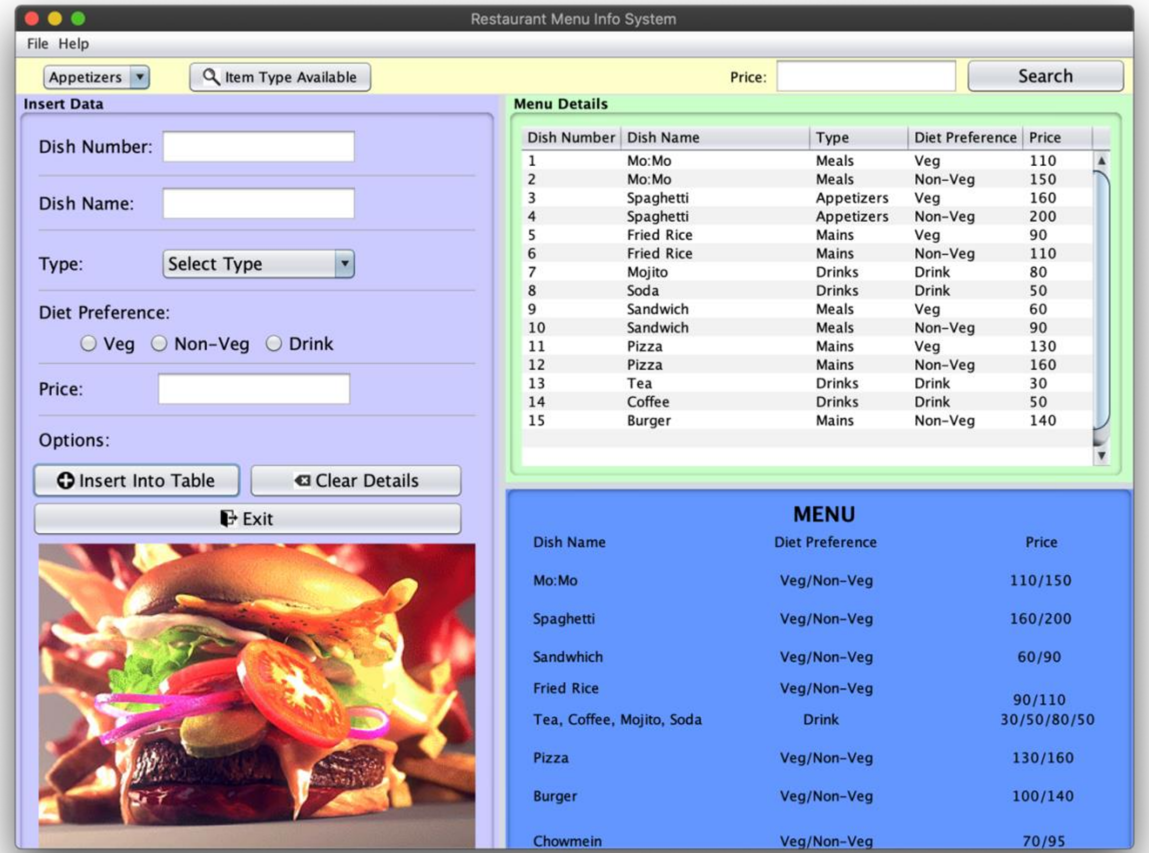
MENU

Dish Name	Diet Preference	Price
Mo:Mo	Veg/Non-Veg	110/150
Spaghetti	Veg/Non-Veg	160/200
Sandwhich	Veg/Non-Veg	60/90
Fried Rice	Veg/Non-Veg	90/110
Tea, Coffee, Mojito, Soda	Drink	30/50/80/50
Pizza	Veg/Non-Veg	130/160
Burger	Veg/Non-Veg	100/140
Chowmein	Veg/Non-Veg	70/95

Figure 7: Testing 3

6.3) Adding an item to the table:

Objective	Checking whether the program adds new items or not. Also if all the information gets added or not.
Action	Adding new products to a unique Model number Checking the added data in the table
Expected Result	The product will get added, Serial Number will be generated, The product is displayed on the table.
Actual Result	The product was added, New Serial Number for the product was generated, The product was displayed at the table.
Conclusion	Test Successful.



Dish Number	Dish Name	Type	Diet Preference	Price
1	Mo:Mo	Meals	Veg	110
2	Mo:Mo	Meals	Non-Veg	150
3	Spaghetti	Appetizers	Veg	160
4	Spaghetti	Appetizers	Non-Veg	200
5	Fried Rice	Mains	Veg	90
6	Fried Rice	Mains	Non-Veg	110
7	Mojito	Drinks	Drink	80
8	Soda	Drinks	Drink	50
9	Sandwich	Meals	Veg	60
10	Sandwich	Meals	Non-Veg	90
11	Pizza	Mains	Veg	130
12	Pizza	Mains	Non-Veg	160
13	Tea	Drinks	Drink	30
14	Coffee	Drinks	Drink	50
15	Burger	Mains	Non-Veg	140

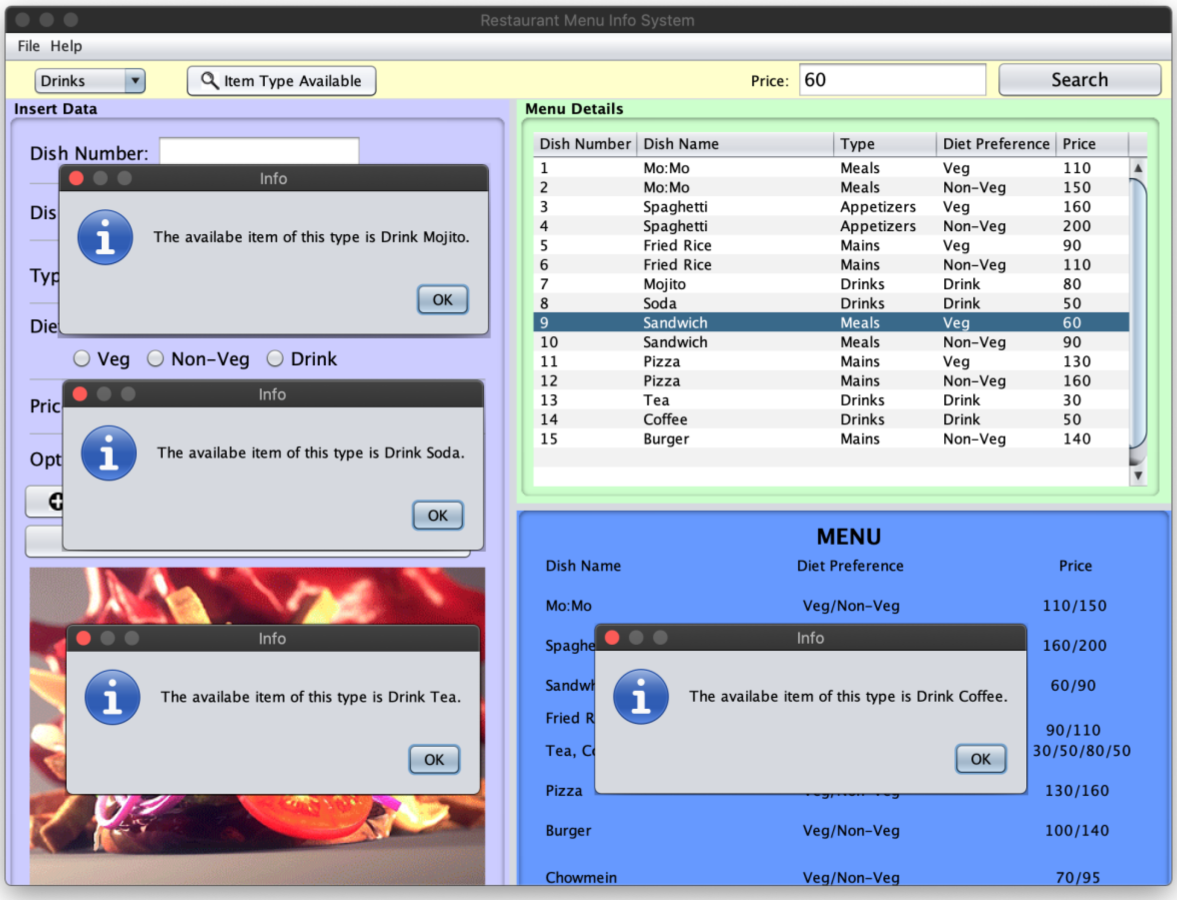
MENU

Dish Name	Diet Preference	Price
Mo:Mo	Veg/Non-Veg	110/150
Spaghetti	Veg/Non-Veg	160/200
Sandwhich	Veg/Non-Veg	60/90
Fried Rice	Veg/Non-Veg	90/110
Tea, Coffee, Mojito, Soda	Drink	30/50/80/50
Pizza	Veg/Non-Veg	130/160
Burger	Veg/Non-Veg	100/140
Chowmein	Veg/Non-Veg	70/95

Figure 8: Testing 4

6.4) Searching items by Dish type:

Objective	Checking whether the program can find items according to the type of the dish.
Action	Searching for items of drinks type.
Expected Result	The program will show as a popup of all the items having the type named drinks when the search button will be pressed.
Actual Result	The program shows popup of all the items having the type named drinks when the search button will be pressed.
Conclusion	Test Successful.



Price: 60

Dish Number:

The availabe item of this type is Drink Mojito.

The availabe item of this type is Drink Soda.

The availabe item of this type is Drink Tea.

The availabe item of this type is Drink Coffee.

Dish Number	Dish Name	Type	Diet Preference	Price
1	Mo:Mo	Meals	Veg	110
2	Mo:Mo	Meals	Non-Veg	150
3	Spaghetti	Appetizers	Veg	160
4	Spaghetti	Appetizers	Non-Veg	200
5	Fried Rice	Mains	Veg	90
6	Fried Rice	Mains	Non-Veg	110
7	Mojito	Drinks	Drink	80
8	Soda	Drinks	Drink	50
9	Sandwich	Meals	Veg	60
10	Sandwich	Meals	Non-Veg	90
11	Pizza	Mains	Veg	130
12	Pizza	Mains	Non-Veg	160
13	Tea	Drinks	Drink	30
14	Coffee	Drinks	Drink	50
15	Burger	Mains	Non-Veg	140

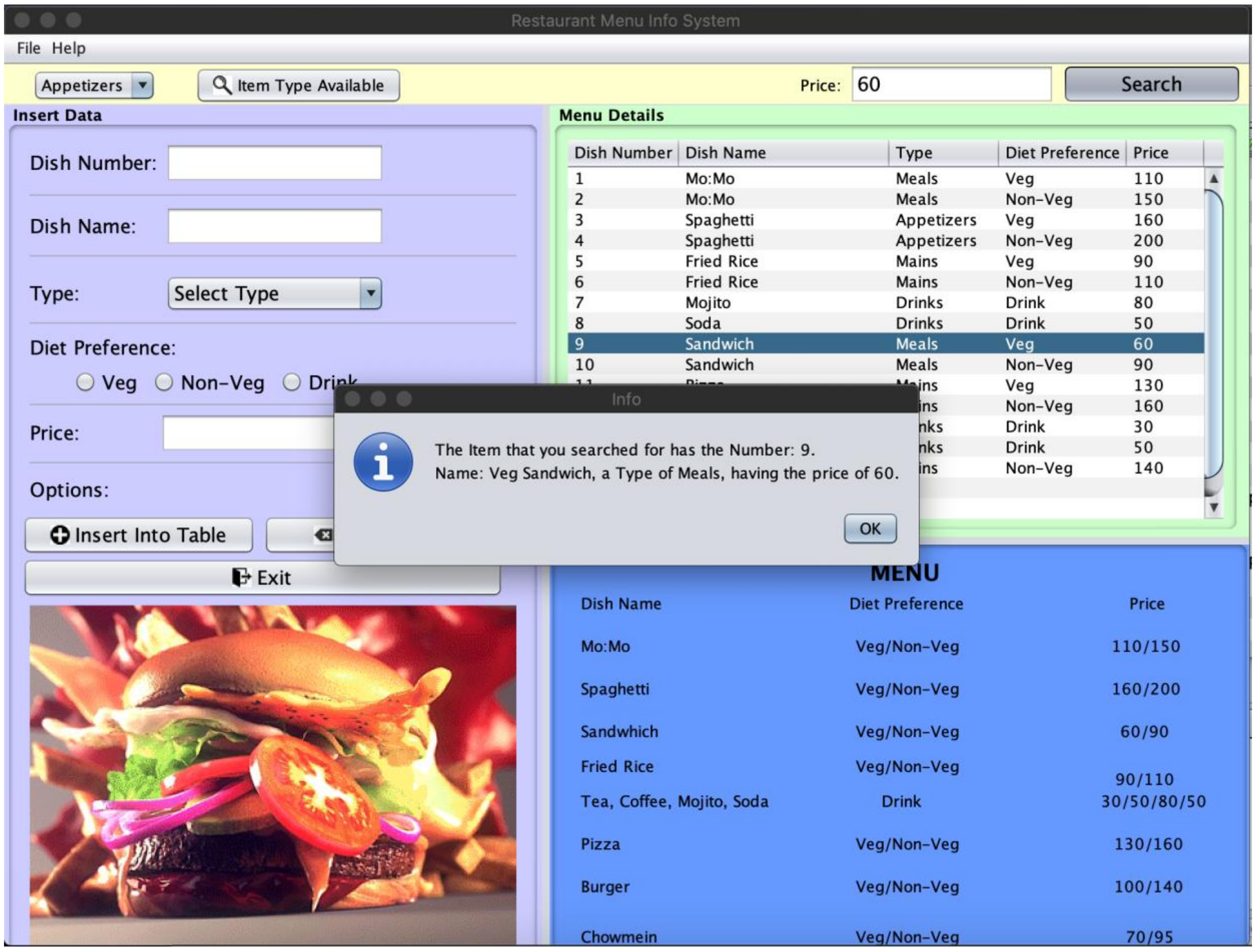
MENU

Dish Name	Diet Preference	Price
Mo:Mo	Veg/Non-Veg	110/150
Spaghe		160/200
Sandwi		60/90
Fried R		90/110
Tea, C		30/50/80/50
Pizza		130/160
Burger	Veg/Non-Veg	100/140
Chowmein	Veg/Non-Veg	70/95

Figure 9: Testing 5

6.5) Search items by Dish Price:

Objective	Checking whether the program can find items according to the price of the dish.
Action	Searching for items having the price of 60.
Expected Result	The program will show as a popup of all the items having the price 60 when the search button will be pressed.
Actual Result	The program shows popup of all the items having the price 60 when the search button will be pressed.
Conclusion	Test Successful.



Price: 60

Dish Number:

Dish Name:

Type: Select Type

Diet Preference: Veg Non-Veg Drink

Price:

Options:

Dish Number	Dish Name	Type	Diet Preference	Price
1	Mo:Mo	Meals	Veg	110
2	Mo:Mo	Meals	Non-Veg	150
3	Spaghetti	Appetizers	Veg	160
4	Spaghetti	Appetizers	Non-Veg	200
5	Fried Rice	Mains	Veg	90
6	Fried Rice	Mains	Non-Veg	110
7	Mojito	Drinks	Drink	80
8	Soda	Drinks	Drink	50
9	Sandwich	Meals	Veg	60
10	Sandwich	Meals	Non-Veg	90
11	Pizza	Mains	Veg	130
		ins	Non-Veg	160
		nks	Drink	30
		nks	Drink	50
		ins	Non-Veg	140

The Item that you searched for has the Number: 9.
Name: Veg Sandwich, a Type of Meals, having the price of 60.

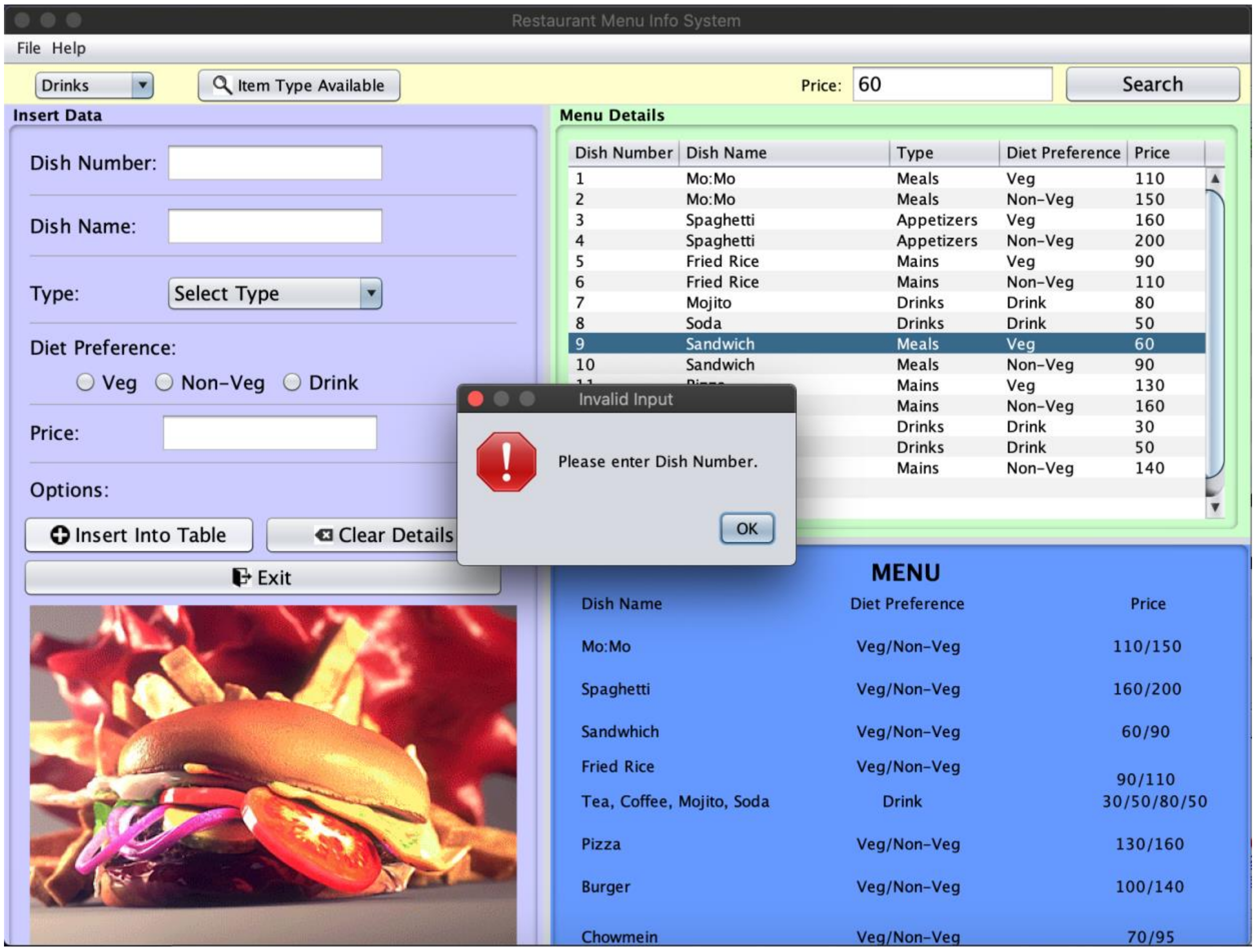
MENU

Dish Name	Diet Preference	Price
Mo:Mo	Veg/Non-Veg	110/150
Spaghetti	Veg/Non-Veg	160/200
Sandwhich	Veg/Non-Veg	60/90
Fried Rice	Veg/Non-Veg	90/110
Tea, Coffee, Mojito, Soda	Drink	30/50/80/50
Pizza	Veg/Non-Veg	130/160
Burger	Veg/Non-Veg	100/140
Chowmein	Veg/Non-Veg	70/95

Figure 10: Testing 6

6.6) System Validation and Function of the Program:

Objective	Checking whether the program can handle various exceptions or not.
Action	Inputting nothing in the dish name and when we don't select Diet Preference whilst inputting the data in the table. Also when we don't put any price whilst searching the item by its price.
Expected Result	The program will display error in a pop-up box to notify about the errors that have occurred due to improper use of the system.
Actual Result	The program specified exact reason for the error for the user suggesting they alter their actions for the right result.
Conclusion	Test Successful.



Price: 60

Dish Number:

Dish Name:

Type: Select Type

Diet Preference: Veg Non-Veg Drink

Price:

Options:

Please enter Dish Number.

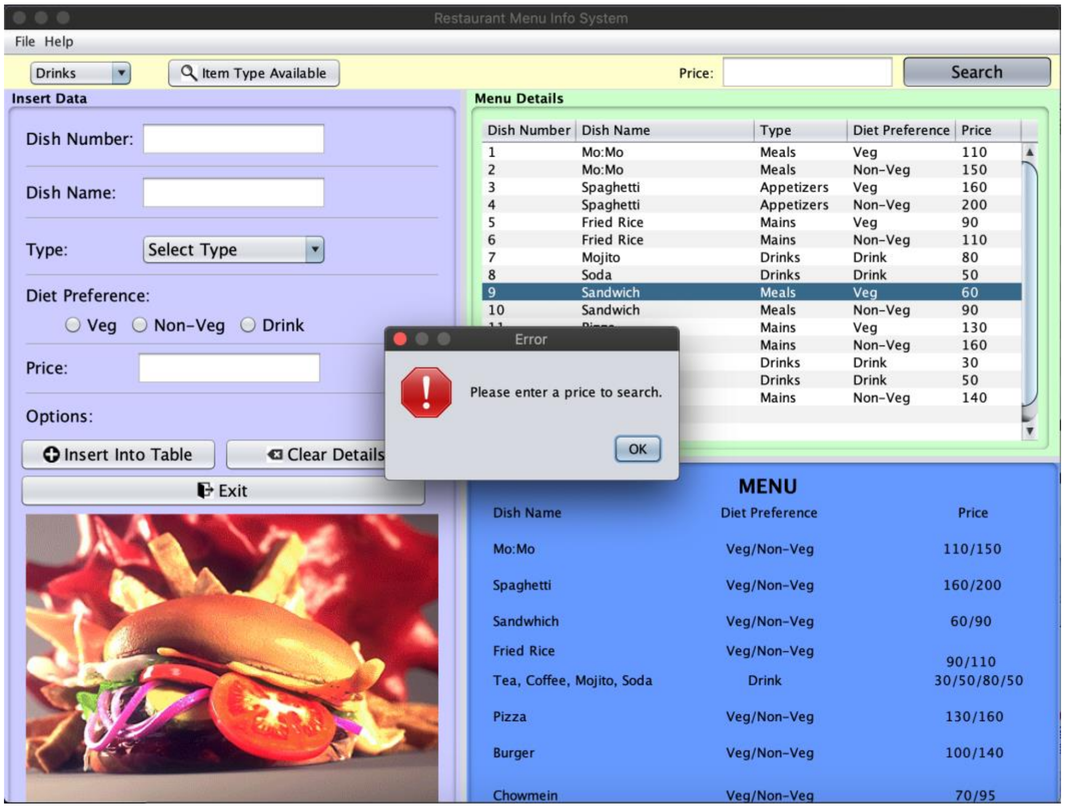
Menu Details

Dish Number	Dish Name	Type	Diet Preference	Price
1	Mo:Mo	Meals	Veg	110
2	Mo:Mo	Meals	Non-Veg	150
3	Spaghetti	Appetizers	Veg	160
4	Spaghetti	Appetizers	Non-Veg	200
5	Fried Rice	Mains	Veg	90
6	Fried Rice	Mains	Non-Veg	110
7	Mojito	Drinks	Drink	80
8	Soda	Drinks	Drink	50
9	Sandwich	Meals	Veg	60
10	Sandwich	Meals	Non-Veg	90
11	Pizza	Mains	Veg	130
		Mains	Non-Veg	160
		Drinks	Drink	30
		Drinks	Drink	50
		Mains	Non-Veg	140

MENU

Dish Name	Diet Preference	Price
Mo:Mo	Veg/Non-Veg	110/150
Spaghetti	Veg/Non-Veg	160/200
Sandwhich	Veg/Non-Veg	60/90
Fried Rice	Veg/Non-Veg	90/110
Tea, Coffee, Mojito, Soda	Drink	30/50/80/50
Pizza	Veg/Non-Veg	130/160
Burger	Veg/Non-Veg	100/140
Chowmein	Veg/Non-Veg	70/95

Figure 11: Testing 7



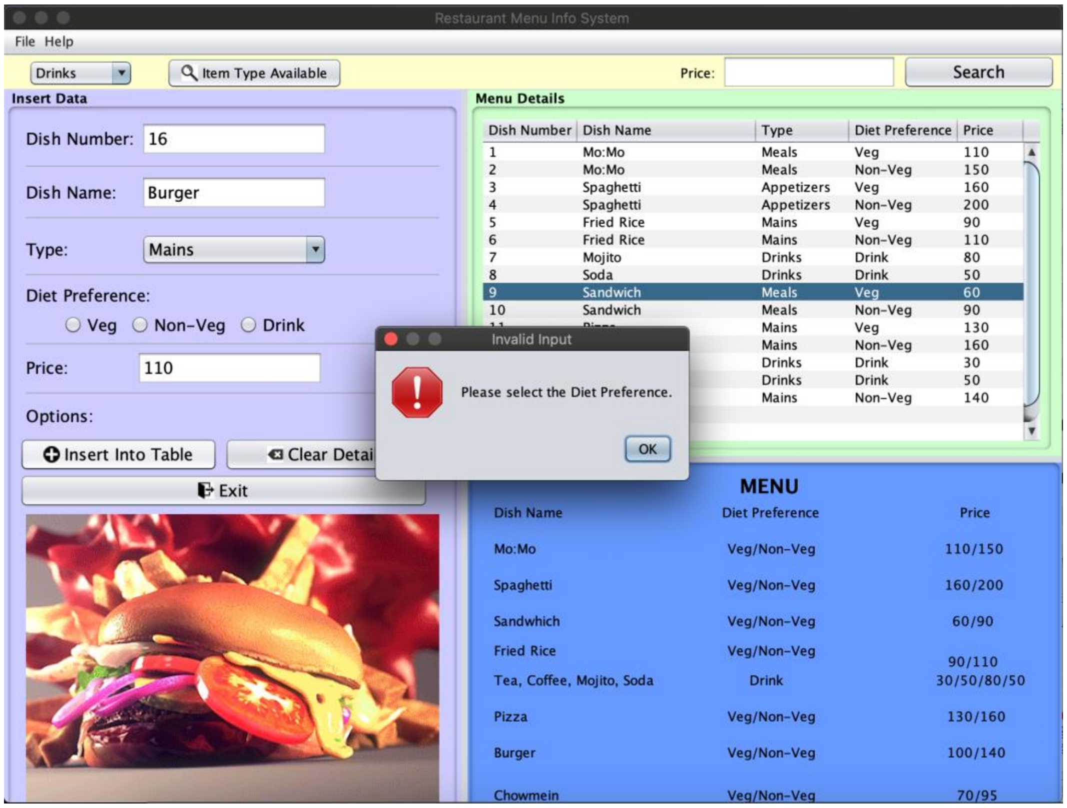
Dish Number	Dish Name	Type	Diet Preference	Price
1	Mo:Mo	Meals	Veg	110
2	Mo:Mo	Meals	Non-Veg	150
3	Spaghetti	Appetizers	Veg	160
4	Spaghetti	Appetizers	Non-Veg	200
5	Fried Rice	Mains	Veg	90
6	Fried Rice	Mains	Non-Veg	110
7	Mojito	Drinks	Drink	80
8	Soda	Drinks	Drink	50
9	Sandwich	Meals	Veg	60
10	Sandwich	Meals	Non-Veg	90
11	Pizza	Mains	Veg	130
		Mains	Non-Veg	160
		Drinks	Drink	30
		Drinks	Drink	50
		Mains	Non-Veg	140

Please enter a price to search.

MENU

Dish Name	Diet Preference	Price
Mo:Mo	Veg/Non-Veg	110/150
Spaghetti	Veg/Non-Veg	160/200
Sandwhich	Veg/Non-Veg	60/90
Fried Rice	Veg/Non-Veg	90/110
Tea, Coffee, Mojito, Soda	Drink	30/50/80/50
Pizza	Veg/Non-Veg	130/160
Burger	Veg/Non-Veg	100/140
Chowmein	Veg/Non-Veg	70/95

Figure 12: Testing 9



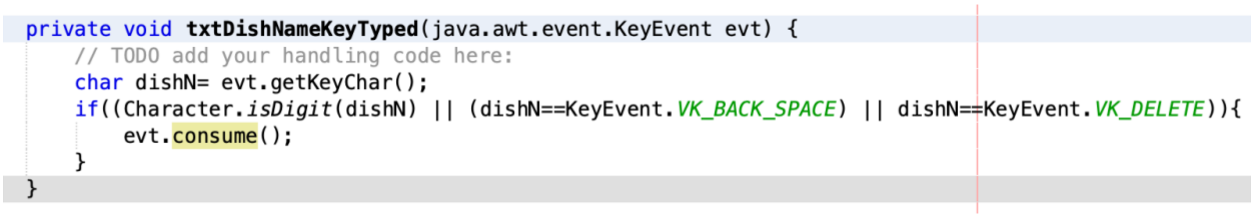
Dish Number	Dish Name	Type	Diet Preference	Price
1	Mo:Mo	Meals	Veg	110
2	Mo:Mo	Meals	Non-Veg	150
3	Spaghetti	Appetizers	Veg	160
4	Spaghetti	Appetizers	Non-Veg	200
5	Fried Rice	Mains	Veg	90
6	Fried Rice	Mains	Non-Veg	110
7	Mojito	Drinks	Drink	80
8	Soda	Drinks	Drink	50
9	Sandwich	Meals	Veg	60
10	Sandwich	Meals	Non-Veg	90
11	Pizza	Mains	Veg	130
		Mains	Non-Veg	160
		Drinks	Drink	30
		Drinks	Drink	50
		Mains	Non-Veg	140

Please select the Diet Preference.

MENU

Dish Name	Diet Preference	Price
Mo:Mo	Veg/Non-Veg	110/150
Spaghetti	Veg/Non-Veg	160/200
Sandwhich	Veg/Non-Veg	60/90
Fried Rice	Veg/Non-Veg	90/110
Tea, Coffee, Mojito, Soda	Drink	30/50/80/50
Pizza	Veg/Non-Veg	130/160
Burger	Veg/Non-Veg	100/140
Chowmein	Veg/Non-Veg	70/95

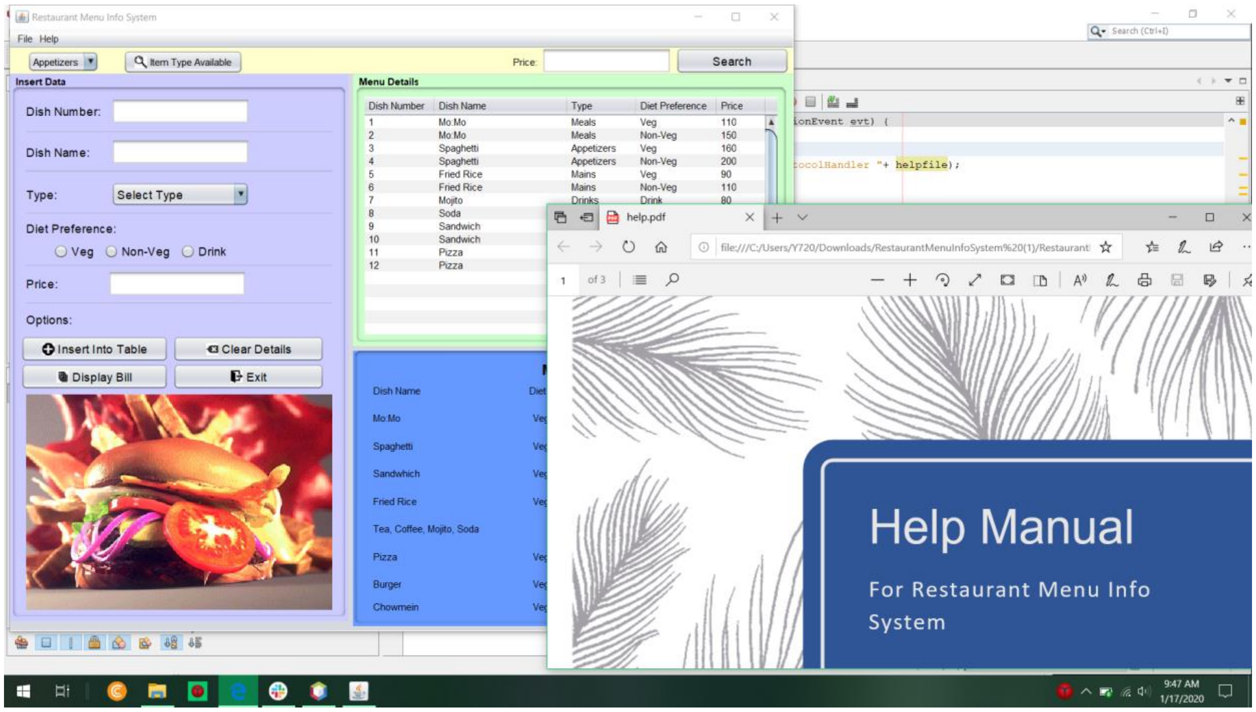
Figure 13: Testing 10



```
private void txtDishNameKeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    char dishN= evt.getKeyChar();
    if((Character.isDigit(dishN) || (dishN==KeyEvent.VK_BACK_SPACE) || dishN==KeyEvent.VK_DELETE)){
        evt.consume();
    }
}
```

Figure 14: Testing 11

This testing is done as when a user inputs wrong key in the DishName text field, they cannot input press the keys other that alphabets, backspace and delete keys. So, the user cannot input unsuitable values in the text fields.



Dish Number	Dish Name	Type	Diet Preference	Price
1	Mo:Mo	Meals	Veg	110
2	Mo:Mo	Meals	Non-Veg	150
3	Spaghetti	Appetizers	Veg	160
4	Spaghetti	Appetizers	Non-Veg	200
5	Fried Rice	Mains	Veg	90
6	Fried Rice	Mains	Non-Veg	110
7	Mojito	Drinks	Drink	80
8	Soda			
9	Sandwich			
10	Sandwich			
11	Pizza			
12	Pizza			

Dish Name	Diet
Mo:Mo	Veg
Spaghetti	Veg
Sandwhich	Veg
Fried Rice	Veg
Tea, Coffee, Mojito, Soda	
Pizza	Veg
Burger	Veg
Chowmein	Veg

Figure 15: Testing 12

Opening the Help Manual when pressing the help option in the Help menu from the menu bar.

7) Conclusion:

While building up any sort of data frameworks, information must be made, put away, added/altered or erased. So, to make this errand simpler information structures must be executed. Furthermore, to process any snippet of data, to sort or to look through different calculations are required to be executed in the program. The arranging calculation utilized in this program is a variation of Selection sort, in this altered choice sort will consequently sort the client contribution to the Array List as indicated by their cost. Despite the fact that direct sort is very wasteful, this rendition of choice sort was utilized in light of the fact that it is more obvious and functions admirably on a modest quantity of information. The arranging was required to sort the information as per their costs in light of the fact that the looking through calculation utilized in this program is a parallel inquiry which can work just on arranged information. Parallel hunt is a proficient looking through calculation as it very well may be communicated in logarithmic capacities. This implies this looking through calculation will work truly well when there is a gigantic measure of information from which a thing should be looked. A different java class was additionally made, which dealt with the entirety of the procedures and strategies required to think about and recover data about any PC object in the primary java class which gave a stage to the GUI.

The UI is likewise a key factor that should be broke down and work upon as it enormously impacts the activities that a client can do in a data framework. So, to improve the client experience the GUI is made as basic and alluring as could reasonably be expected with the goal that it will influence the client emphatically while utilizing this program. While testing, the product was required to play out a specific undertaking which it has done easily. The GUI can be effectively opened by bringing in the task and including new information is moderately simple. The program will sort the information as it is entered by its cost so when the client look through any item the item can straightforwardly utilize a double quest calculation to scan for that thing. The program additionally includes numerous little different highlights like clearing all the information fields, leaving the program and review the client direct for the program. The assistance button likewise contains a capacity that will drill down every one of the things put away in the ArrayList which can be seen in the reassure.

There were a lot of complications in the making of this information system, but we had our tutors and module leader to help us out of the difficult situation and we were were well-guided by them. Eventually, we got to know about most of the things precisely and we are now more familiar with searching and sorting. Since, during this project, we took help from various sites, books along with our module leaders and seniors. The main highlights of this system is the implementation of binary search which were done in group task, as well as individual task.

To conclude, even if the project was difficult in the beginning, we cleared our queries by asking our tutors and module leader, which made us understand more about the project and the ways to do different works. We are thankful to our tutors and module leader for guiding us throughout the completion of this project. Similarly, we got opportunity to understand, practice, learn and improve our skills by working on this Restaurant Menu Info System and to co-operate with members of our group. Finally, we got to enhance our intelligence, creativity and skills, and henceforth can address to the problems if we face related to it in the near future.

8) Bibliography

JavaTpoint. (2011-2018) [Online]. Available from: https://www.javatpoint.com/java-oops-concepts [Accessed 17 Jan 2020].

Sehgal, K. (2018) *An Intriduction to Selection Sort*.

Techopedia. (2020) [Online]. Available from: https://www.techopedia.com/definition/5435/graphical-user-interface-gui [Accessed January 2020].

tutorialspoint. (2020) [Online]. Available from: https://www.tutorialspoint.com/data_structures_algorithms/binary_search_algorithm.htm [Accessed 16 Jan 2020].

9) Appendix:

More about Binary Search:

First of all, the data is sorted according to the price and is kept in the array list. After the completion of data sorting, index number zero of the array list is stored in the variable named as low and the last index number of the array list is stored in the variable named as high. The value entered by the user is stored in the variable named as key and the index to be stored in the variable named mid is found by getting the average of index number stored in the variable low and high.

After getting the index number to be stored in the variable named mid it is checked with the key value whether the value at that index is equal with the key value or not. If it is equal, then the binary search is finished but if it isn't equal with the key value then it is checked that whether the key value is greater or smaller than value stored at that index number stored in the variable named mid in the array list as the data are sorted. If the key value is greater than the data then variable named low stores the index number one step greater than that of previous index number stored in the variable named mid and if the key value is smaller than the data at that index then variable named high stores the index number one step smaller than that of previous index number stored in the variable named mid.

The above process is repeated until the value at the index number stored in variable mid becomes equal to the value stored in the variable key. The loop will run until the index number stored in the variable low will be smaller than the index number stored in the variable high. If the condition mid equals to key is not met even after the completion of the loop, then it means there is not such value in the array list.

Go back to Implementation of Binary Search…..

Help Manual of the Program:

Welcome to the Help Manual for the Restaurant Menu Info System This is an information system application made for a restaurant called as the Restaurant Menu Info System. Here, you can find out the items available in the restaurant menu and you can search the item according to the price that you want, as well as through the type of the item.

❖ How to use this application for searching the dish from price?

  First of all, put the price in the text field denoted by price and then press the search button. If the food is available of the price you entered, then a dialogue box will appear showing the details about that food. If not u will be displayed a message saying search data not available. You must enter valid data i.e. integer data, otherwise it will not accept your data. Also, you cannot input empty data, else it will display another message saying to input price.

❖ How to find the dish from its type?

  Click on the combo box on the top left corner, below the menu bar and then click the type as your wish and then click the item type available button. Then a dialogue box will be appeared showing the details about that food. If not, u will be displayed a message saying search data not available.

❖ The data can be also added into the table by entering the valid values in the text fields and by clicking in the insert button.

❖ The button Clear can be used to clear the text fields, if you inserted wrong data in the text fields.

❖ The button Exit can be used to exit the application. Exit can also be done from the file menu in the menu bar.

❖ You can open a selected csv file from the open icon in the menu bar.

❖ You can access this help manual anytime from the help menu in the menu bar.

❖ Following are the shortcut keys that can be pressed for ease of use:
  - Ctrl + S = Open
  - Ctrl + E= Exit
  - Ctrl + H= User Help

Whole Code of the Program:

```
import java.awt.event.KeyEvent;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.net.URL;

import java.util.ArrayList;

import javax.swing.JOptionPane;

import javax.swing.table.DefaultTableModel;


/*

 * To change this license header, choose License Headers in Project
Properties.

 * To change this template file, choose Tools | Templates

 * and open the template in the editor.

 */
```

```
/**
 *
 * @author abhi
 */
public class MenuInfo extends javax.swing.JFrame {
    int arraysize=0;
    int[] getPrice = new int[0];
    ArrayList<Integer> priceArray=new ArrayList<>();
    ArrayList<String> typeArray=new ArrayList<>();
    ArrayList<String> availableDish=new ArrayList<>();
    /**
     * Creates new form menuIS
     */
    //
    public MenuInfo() {
        initComponents();
        loadData();
    }
    /**
     * This method is called from within the constructor to initialize the
form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        buttonGroup1 = new javax.swing.ButtonGroup();
        jPanel2 = new javax.swing.JPanel();
        btnSearch = new javax.swing.JButton();
        txtSearchPrice = new javax.swing.JTextField();
        cbAvailableItem = new javax.swing.JComboBox<>();
        btnItemTypeAvailable = new javax.swing.JButton();
        lblSearchPrice = new javax.swing.JLabel();
        jPanel4 = new javax.swing.JPanel();
        lblDishNum = new javax.swing.JLabel();
        txtDishNum = new javax.swing.JTextField();
        jSeparator2 = new javax.swing.JSeparator();
        txtDishName = new javax.swing.JTextField();
        lblDishName = new javax.swing.JLabel();
        jSeparator4 = new javax.swing.JSeparator();
        lblType = new javax.swing.JLabel();
        cbType = new javax.swing.JComboBox<>();
        jSeparator3 = new javax.swing.JSeparator();
        lblDietPreference = new javax.swing.JLabel();
        veg = new javax.swing.JRadioButton();
        drink = new javax.swing.JRadioButton();
        nonVeg = new javax.swing.JRadioButton();
        jSeparator1 = new javax.swing.JSeparator();
        lblPrice = new javax.swing.JLabel();
        txtPrice = new javax.swing.JTextField();
        jSeparator7 = new javax.swing.JSeparator();
        jLabel1 = new javax.swing.JLabel();
        btnInsert = new javax.swing.JButton();
        btnClear = new javax.swing.JButton();
        btnExit = new javax.swing.JButton();
        jLabel2 = new javax.swing.JLabel();
        jPanel1 = new javax.swing.JPanel();
        jScrollPane1 = new javax.swing.JScrollPane();
        menuDetailsTable = new javax.swing.JTable();
        jPanel3 = new javax.swing.JPanel();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel6 = new javax.swing.JLabel();
        jLabel7 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        jLabel9 = new javax.swing.JLabel();
        jLabel10 = new javax.swing.JLabel();
        jLabel11 = new javax.swing.JLabel();
        jLabel12 = new javax.swing.JLabel();
        jLabel13 = new javax.swing.JLabel();
        jLabel14 = new javax.swing.JLabel();
        jLabel15 = new javax.swing.JLabel();
        jLabel16 = new javax.swing.JLabel();
        jLabel17 = new javax.swing.JLabel();
        jLabel18 = new javax.swing.JLabel();
        jLabel19 = new javax.swing.JLabel();
        jLabel20 = new javax.swing.JLabel();
        jLabel21 = new javax.swing.JLabel();
        jLabel22 = new javax.swing.JLabel();
        jLabel23 = new javax.swing.JLabel();
        jLabel24 = new javax.swing.JLabel();
        jLabel25 = new javax.swing.JLabel();
        jLabel26 = new javax.swing.JLabel();
        jLabel27 = new javax.swing.JLabel();
        jLabel28 = new javax.swing.JLabel();
        jLabel29 = new javax.swing.JLabel();
        jLabel30 = new javax.swing.JLabel();
        jMenuBar1 = new javax.swing.JMenuBar();
        menuFile = new javax.swing.JMenu();
        menuItemOpen = new javax.swing.JMenuItem();
        jMenuItem1 = new javax.swing.JMenuItem();
        menuHelp = new javax.swing.JMenu();
        menuItemHelp = new javax.swing.JMenuItem();
        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("Restaurant Menu Info System");
        setBackground(new java.awt.Color(204, 204, 204));

        jPanel2.setBackground(new java.awt.Color(255, 255, 204));

        btnSearch.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        btnSearch.setText("Search");
        btnSearch.setFocusable(false);
btnSearch.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
        btnSearch.setVerticalTextPosition(javax.swing.SwingConstants.BOTTOM);
        btnSearch.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnSearchActionPerformed(evt);
            }
        });

        txtSearchPrice.setFont(new java.awt.Font("SansSerif", 0, 15)); //
NOI18N
        txtSearchPrice.setToolTipText("");
        txtSearchPrice.setName(""); // NOI18N
        txtSearchPrice.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtSearchPriceActionPerformed(evt);
            }
        });
        txtSearchPrice.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                txtSearchPriceKeyTyped(evt);
            }
        });


        cbAvailableItem.setModel(new javax.swing.DefaultComboBoxModel<>(new
String[] { "Appetizers", "Meals", "Mains", "Drinks" }));

        cbAvailableItem.addActionListener(new java.awt.event.ActionListener()
{

            public void actionPerformed(java.awt.event.ActionEvent evt) {

                cbAvailableItemActionPerformed(evt);

            }

        });


        btnItemTypeAvailable.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/search icon.jpg"))); //
NOI18N

        btnItemTypeAvailable.setText("Item Type Available");

        btnItemTypeAvailable.setActionCommand("");

        btnItemTypeAvailable.addActionListener(new
java.awt.event.ActionListener() {

            public void actionPerformed(java.awt.event.ActionEvent evt) {

                btnItemTypeAvailableActionPerformed(evt);

            }

        });


        lblSearchPrice.setText("Price:");


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

        jPanel2.setLayout(jPanel2Layout);

        jPanel2Layout.setHorizontalGroup(


jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

            .addGroup(jPanel2Layout.createSequentialGroup()

                .addGap(22, 22, 22)

                .addComponent(cbAvailableItem,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                .addGap(30, 30, 30)
                .addComponent(btnItemTypeAvailable)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 308,
Short.MAX_VALUE)
                .addComponent(lblSearchPrice)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(txtSearchPrice,
javax.swing.GroupLayout.PREFERRED_SIZE, 159,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(btnSearch,
javax.swing.GroupLayout.PREFERRED_SIZE, 139,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                .addComponent(txtSearchPrice)
                .addComponent(lblSearchPrice))
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addGap(0, 0, Short.MAX_VALUE)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.TRAILING)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                        .addComponent(btnItemTypeAvailable,
javax.swing.GroupLayout.PREFERRED_SIZE, 29,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                        .addComponent(cbAvailableItem,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(btnSearch)))
        );

        txtSearchPrice.getAccessibleContext().setAccessibleName("");

        jPanel4.setBackground(new java.awt.Color(204, 204, 255));

jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder("Insert
Data"));

        lblDishNum.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        lblDishNum.setText("Dish Number:");

        txtDishNum.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        txtDishNum.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtDishNumActionPerformed(evt);
            }
        });
        txtDishNum.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                txtDishNumKeyTyped(evt);
            }
        });

        txtDishName.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        txtDishName.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtDishNameActionPerformed(evt);
            }
        });
        txtDishName.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                txtDishNameKeyTyped(evt);
            }
        });

        lblDishName.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        lblDishName.setText("Dish Name:");

        lblType.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        lblType.setText("Type:");

        cbType.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        cbType.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] {
"Select Type", "Appetizers", "Meals", "Mains", "Drinks" }));
        cbType.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                cbTypeActionPerformed(evt);
            }
        });

        lblDietPreference.setFont(new java.awt.Font("SansSerif", 0, 15)); //
NOI18N
        lblDietPreference.setText("Diet Preference:");

        buttonGroup1.add(veg);
        veg.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        veg.setText("Veg");
        veg.setToolTipText("");
        veg.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                vegActionPerformed(evt);
            }
        });

        buttonGroup1.add(drink);
        drink.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        drink.setText("Drink");
        drink.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                drinkActionPerformed(evt);
            }
        });

        buttonGroup1.add(nonVeg);
        nonVeg.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        nonVeg.setText("Non-Veg");
        nonVeg.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                nonVegActionPerformed(evt);
            }
        });

        lblPrice.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        lblPrice.setText("Price:");

        txtPrice.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        txtPrice.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                txtPriceActionPerformed(evt);
            }
        });
        txtPrice.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                txtPriceKeyTyped(evt);
            }
        });

        jLabel1.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        jLabel1.setText("Options:");

        btnInsert.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        btnInsert.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/insert icon.jpg"))); //
NOI18N
        btnInsert.setText("Insert Into Table");
        btnInsert.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnInsertActionPerformed(evt);
            }
        });

        btnClear.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        btnClear.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/clear icon.jpg"))); //
NOI18N
        btnClear.setText("Clear Details");
        btnClear.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnClearActionPerformed(evt);
            }
        });

        btnExit.setFont(new java.awt.Font("SansSerif", 0, 15)); // NOI18N
        btnExit.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/exit icon.jpg"))); //
NOI18N
        btnExit.setText("Exit");
        btnExit.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnExitActionPerformed(evt);
            }
        });

        jLabel2.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/img1.gif"))); // NOI18N

        javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4);
        jPanel4.setLayout(jPanel4Layout);
        jPanel4Layout.setHorizontalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel4Layout.createSequentialGroup()

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
                    .addGroup(jPanel4Layout.createSequentialGroup()
                        .addComponent(btnInsert,
javax.swing.GroupLayout.PREFERRED_SIZE, 181,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(btnClear,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
                    .addComponent(btnExit,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
                .addGap(16, 16, 16))
            .addGroup(jPanel4Layout.createSequentialGroup()
                .addContainerGap()

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
```

```
                    .addComponent(jLabel2,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                    .addComponent(jSeparator1,
javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(jSeparator3,
javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(jSeparator4,
javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(jSeparator2,
javax.swing.GroupLayout.Alignment.TRAILING)
                    .addComponent(jSeparator7)
                    .addGroup(jPanel4Layout.createSequentialGroup()

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
                            .addGroup(jPanel4Layout.createSequentialGroup()
                                .addComponent(lblDishNum,
javax.swing.GroupLayout.PREFERRED_SIZE, 99,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(txtDishNum,
javax.swing.GroupLayout.PREFERRED_SIZE, 170,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(jPanel4Layout.createSequentialGroup()
                                .addComponent(lblType,
javax.swing.GroupLayout.PREFERRED_SIZE, 99,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(cbType,
javax.swing.GroupLayout.PREFERRED_SIZE, 170,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(jPanel4Layout.createSequentialGroup()
                                .addComponent(lblDishName,
javax.swing.GroupLayout.PREFERRED_SIZE, 99,
javax.swing.GroupLayout.PREFERRED_SIZE)

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

```
                                .addComponent(txtDishName,
javax.swing.GroupLayout.PREFERRED_SIZE, 170,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(jPanel4Layout.createSequentialGroup()
                                .addComponent(lblPrice,
javax.swing.GroupLayout.PREFERRED_SIZE, 99,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(2, 2, 2)
                                .addComponent(txtPrice,
javax.swing.GroupLayout.PREFERRED_SIZE, 170,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addComponent(lblDietPreference)
                            .addGroup(jPanel4Layout.createSequentialGroup()
                                .addGap(34, 34, 34)
                                .addComponent(veg)

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

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                                .addComponent(drink))
                            .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 99,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(0, 0, Short.MAX_VALUE)))
                .addContainerGap())
        );
        jPanel4Layout.setVerticalGroup(

jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel4Layout.createSequentialGroup()

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                    .addComponent(lblDishNum,
javax.swing.GroupLayout.PREFERRED_SIZE, 35,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                    .addComponent(txtDishNum,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jSeparator2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(8, 8, 8)

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                    .addComponent(lblDishName,
javax.swing.GroupLayout.PREFERRED_SIZE, 23,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtDishName,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jSeparator4,
javax.swing.GroupLayout.PREFERRED_SIZE, 8,
javax.swing.GroupLayout.PREFERRED_SIZE)

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

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                    .addComponent(lblType,
javax.swing.GroupLayout.PREFERRED_SIZE, 32,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(cbType,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jSeparator3,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

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

```
                .addComponent(lblDietPreference,
javax.swing.GroupLayout.PREFERRED_SIZE, 24,
javax.swing.GroupLayout.PREFERRED_SIZE)

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

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                    .addComponent(veg)
                    .addComponent(nonVeg)
                    .addComponent(drink))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jSeparator1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

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

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                    .addComponent(lblPrice,
javax.swing.GroupLayout.PREFERRED_SIZE, 28,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(txtPrice))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jSeparator7,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 28,
javax.swing.GroupLayout.PREFERRED_SIZE)

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

.addGroup(jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                    .addComponent(btnInsert)
                    .addComponent(btnClear))
                .addGap(2, 2, 2)
                .addComponent(btnExit)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel2, 
javax.swing.GroupLayout.PREFERRED_SIZE, 265, 
javax.swing.GroupLayout.PREFERRED_SIZE))
        );


        jPanel1.setBackground(new java.awt.Color(204, 255, 204));


        jScrollPane1.setBackground(new java.awt.Color(204, 255, 204));

jScrollPane1.setBorder(javax.swing.BorderFactory.createTitledBorder(javax.swi
ng.BorderFactory.createTitledBorder(javax.swing.BorderFactory.createTitledBor
der(javax.swing.BorderFactory.createTitledBorder(javax.swing.BorderFactory.cr
eateTitledBorder("Menu Details"))))));


        menuDetailsTable.setModel(new javax.swing.table.DefaultTableModel(
            new Object [][] {
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},
                {null, null, null, null, null},

                {null, null, null, null, null},

                {null, null, null, null, null},

                {null, null, null, null, null}

            },

            new String [] {

                "Dish Number", "Dish Name", "Type", "Diet Preference",
"Price"

            }

        ));

        jScrollPane1.setViewportView(menuDetailsTable);

        if (menuDetailsTable.getColumnModel().getColumnCount() > 0) {

            menuDetailsTable.getColumnModel().getColumn(0).setMinWidth(86);

            menuDetailsTable.getColumnModel().getColumn(0).setMaxWidth(86);

            menuDetailsTable.getColumnModel().getColumn(2).setMinWidth(85);

            menuDetailsTable.getColumnModel().getColumn(2).setMaxWidth(85);

            menuDetailsTable.getColumnModel().getColumn(3).setMinWidth(99);

            menuDetailsTable.getColumnModel().getColumn(3).setMaxWidth(99);

            menuDetailsTable.getColumnModel().getColumn(4).setMinWidth(60);

            menuDetailsTable.getColumnModel().getColumn(4).setMaxWidth(60);

        }


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

        jPanel1.setLayout(jPanel1Layout);

        jPanel1Layout.setHorizontalGroup(


jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

            .addGap(0, 0, Short.MAX_VALUE)


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

                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 536,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)))
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 334, Short.MAX_VALUE)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 334,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(0, 0, Short.MAX_VALUE)))
        );

        jPanel3.setBackground(new java.awt.Color(102, 153, 255));
        jPanel3.setBorder(javax.swing.BorderFactory.createTitledBorder(""));

        jLabel4.setFont(new java.awt.Font("sansserif", 1, 18)); // NOI18N
        jLabel4.setText("MENU");

        jLabel5.setText("Dish Name");

        jLabel3.setText("Diet Preference");

        jLabel6.setText("Price");

        jLabel7.setText("Mo:Mo");

        jLabel8.setText("Spaghetti");
        jLabel9.setText("Sandwhich");

        jLabel10.setText("Fried Rice");

        jLabel11.setText("Tea, Coffee, Mojito, Soda");

        jLabel12.setText("Pizza");

        jLabel13.setText("Burger");

        jLabel14.setText("Chowmein");

        jLabel15.setText("Veg/Non-Veg");

        jLabel16.setText("Veg/Non-Veg");

        jLabel17.setText("Veg/Non-Veg");

        jLabel18.setText("Veg/Non-Veg");

        jLabel19.setText("Drink");

        jLabel20.setText("Veg/Non-Veg");

        jLabel21.setText("Veg/Non-Veg");

        jLabel22.setText("Veg/Non-Veg");

        jLabel23.setText("110/150");

        jLabel24.setText("160/200");
        jLabel25.setText("60/90");

        jLabel26.setText("90/110");

        jLabel27.setText("30/50/80/50");

        jLabel28.setText("130/160");

        jLabel29.setText("100/140");

        jLabel30.setText("70/95");

        javax.swing.GroupLayout jPanel3Layout = new
javax.swing.GroupLayout(jPanel3);
        jPanel3.setLayout(jPanel3Layout);
        jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addGap(14, 14, 14)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
                    .addGroup(jPanel3Layout.createSequentialGroup()
                        .addComponent(jLabel10)
                        .addGap(0, 0, Short.MAX_VALUE))
                    .addGroup(jPanel3Layout.createSequentialGroup()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.TRAILING)
                                .addComponent(jLabel5)
                                .addComponent(jLabel7,
javax.swing.GroupLayout.Alignment.LEADING)
                                .addComponent(jLabel8,
javax.swing.GroupLayout.Alignment.LEADING))
                            .addComponent(jLabel9)
                            .addComponent(jLabel14)
                            .addComponent(jLabel13)
                            .addComponent(jLabel12)
                            .addComponent(jLabel11))

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.TRAILING)

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
jPanel3Layout.createSequentialGroup()
                                .addGap(78, 78, 78)
                                .addComponent(jLabel4)
                                .addContainerGap())
                            .addGroup(jPanel3Layout.createSequentialGroup()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)

.addGroup(jPanel3Layout.createSequentialGroup()
                                        .addGap(67, 67, 67)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
                                            .addComponent(jLabel17)
                                            .addComponent(jLabel16)
                                            .addComponent(jLabel20)
                                            .addComponent(jLabel21)
                                            .addComponent(jLabel22)
                                            .addComponent(jLabel15)

.addGroup(jPanel3Layout.createSequentialGroup()
                                                .addGap(20, 20, 20)
```

```
                                                        .addComponent(jLabel19))
                                                    .addComponent(jLabel18)))

.addGroup(jPanel3Layout.createSequentialGroup()
                                                .addGap(62, 62, 62)
                                                .addComponent(jLabel3)))

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 106,
Short.MAX_VALUE)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
                                                            .addComponent(jLabel28,
javax.swing.GroupLayout.Alignment.TRAILING)
                                                            .addComponent(jLabel29,
javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()

.addComponent(jLabel30)
                                                                .addGap(7, 7, 7)))
                                                        .addGap(41, 41, 41))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()
                                                        .addComponent(jLabel27)
                                                        .addGap(23, 23, 23))
```

```
.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()
                                                .addComponent(jLabel26)
                                                .addGap(48, 48, 48))))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()
                                        .addGap(0, 0, Short.MAX_VALUE)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()
                                                .addComponent(jLabel6)
                                                .addGap(55, 55, 55))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()
                                                .addComponent(jLabel25)
                                                .addGap(52, 52, 52))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
                                                    .addComponent(jLabel23)
                                                    .addComponent(jLabel24))
                                                .addGap(43, 43, 43))))))))))
        );
        jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addComponent(jLabel4)
```

```
.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
                    .addGroup(jPanel3Layout.createSequentialGroup()

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

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                            .addComponent(jLabel5)
                            .addComponent(jLabel3)
                            .addComponent(jLabel6))
                        .addGap(18, 18, 18)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                            .addComponent(jLabel7)
                            .addComponent(jLabel23)
                            .addComponent(jLabel15))
                        .addGap(18, 18, 18)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                            .addComponent(jLabel8)
                            .addComponent(jLabel16)
                            .addComponent(jLabel24))
                        .addGap(18, 18, 18)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                            .addComponent(jLabel9)
                            .addComponent(jLabel17)
                            .addComponent(jLabel25))
                        .addGap(12, 12, 12)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
```

```
                            .addComponent(jLabel10)
                            .addComponent(jLabel18))
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                            .addComponent(jLabel11)
                            .addComponent(jLabel19)
                            .addComponent(jLabel27))
                        .addGap(18, 18, 18)
.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                            .addComponent(jLabel12)
                            .addComponent(jLabel20)
                            .addComponent(jLabel28))
                        .addGap(18, 18, 18)
.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                            .addComponent(jLabel13)
                            .addComponent(jLabel21)
                            .addComponent(jLabel29))
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
                            .addComponent(jLabel14)
                            .addComponent(jLabel22)
                            .addComponent(jLabel30))
                        .addContainerGap())
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jLabel26)
                        .addGap(128, 128, 128))))
        );

        menuFile.setText("File");


menuItemOpen.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event
.KeyEvent.VK_S, java.awt.event.InputEvent.CTRL_MASK));
        menuItemOpen.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/open icon.jpg"))); //
NOI18N
        menuItemOpen.setText("Open");
        menuItemOpen.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                menuItemOpenActionPerformed(evt);
            }
        });
        menuFile.add(menuItemOpen);


jMenuItem1.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.K
eyEvent.VK_E, java.awt.event.InputEvent.CTRL_MASK));
        jMenuItem1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/exit icon.jpg"))); //
NOI18N
        jMenuItem1.setText("Exit");
        jMenuItem1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem1ActionPerformed(evt);
            }
        });
        menuFile.add(jMenuItem1);
        jMenuBar1.add(menuFile);


        menuHelp.setText("Help");



menuItemHelp.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event
.KeyEvent.VK_H, java.awt.event.InputEvent.CTRL_MASK));
        menuItemHelp.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/help icon.jpg"))); //
NOI18N
        menuItemHelp.setText("User Help");
        menuItemHelp.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                menuItemHelpActionPerformed(evt);
            }
        });
        menuHelp.add(menuItemHelp);


        jMenuBar1.add(menuHelp);


        setJMenuBar(jMenuBar1);


        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jPanel4,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G, false)
                    .addComponent(jPanel1,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                    .addComponent(jPanel3,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(1, 1, 1)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G, false)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jPanel4,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(7, 7, 7))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jPanel1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jPanel3,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                        .addContainerGap())))
        );
        pack();
    }// </editor-fold>
    private void openData() {
        URL filePath = MenuInfo.class.getResource("CSVFiles/open.csv");
//getting url of csv file to load from package CourseWork
        try {
            File file = new File(filePath.toURI());
            BufferedReader br = new BufferedReader(new FileReader(file));
            DefaultTableModel model = (DefaultTableModel)
menuDetailsTable.getModel();
            Object[] tableLines = br.lines().toArray();
            for(int i = 0; i < tableLines.length; i++){
                String[] row = tableLines[i].toString().split(",");
                //model.insertRow(i, row);
                int rowCount = menuDetailsTable.getRowCount();
                int colCount = menuDetailsTable.getColumnCount();
                boolean insert = false;
                int counterRow = 0;
                while (insert == false)
                {
                    int counterCol = 0;
                    boolean emptyRow = true;
                    while ( emptyRow == true && counterCol < colCount )
                    {
                        //System.out.println("CHECKING ROW: " + counterRow +
", CHECKING COLUMN " + counterCol);
                        Object cellValue =
menuDetailsTable.getValueAt(counterRow, counterCol);
                        //System.out.println("VALUE: "+cellValue);
                        if ( cellValue == null )
                        {
                            counterCol++;
                        }else
                        {
                            //System.out.println("value is found, skip row");
                            break; // the row is not empty, jump to nextRow
                        }
                    }
                    if (emptyRow= true && counterCol == colCount)
                    {
                       //row is empty
                       //System.out.println("Row is empty ");
                       insert = true;
                    }
                    else
                    {
                        counterRow++;
                    }
                }
                if (insert==true)
                {
                    int count = 0;
                    for (Object datum : row)
                    {
                        menuDetailsTable.setValueAt(datum, counterRow,
count);
                        count++;
                    }
                }
            }
        }
        catch (Exception e) {
            JOptionPane.showMessageDialog(this,"File not
found.","Error",JOptionPane.ERROR_MESSAGE);
        }
    }
    private void loadData() {
        URL filePath = MenuInfo.class.getResource("CSVFiles/load.csv");
//getting url of csv file to load from the package
        try {
            File file = new File(filePath.toURI());
            BufferedReader br = new BufferedReader(new FileReader(file));
            DefaultTableModel model = (DefaultTableModel)
menuDetailsTable.getModel();
            Object[] tableLines = br.lines().toArray();
            for(int i = 0; i < tableLines.length; i++){
                String[] row = tableLines[i].toString().split(",");
                //model.insertRow(i, row);
                int rowCount = menuDetailsTable.getRowCount();
                int colCount = menuDetailsTable.getColumnCount();
                boolean insert = false;
                int counterRow = 0;
                while (insert == false) {
                    int counterCol = 0;
                    boolean emptyRow = true;
                    while ( emptyRow == true && counterCol < colCount ) {
                        //System.out.println("CHECKING ROW: " + counterRow +
", CHECKING COLUMN " + counterCol);
                        Object cellValue =
menuDetailsTable.getValueAt(counterRow, counterCol);
                        //System.out.println("VALUE: "+cellValue);
                        if ( cellValue == null ) {
                            counterCol++;
                        }
                        else {
                            //System.out.println("value is found, skip row");
                            break; // the row is not empty, jump to nextRow
                        }
                    }
                    if (emptyRow= true && counterCol == colCount) {
                       //row is empty
                       //System.out.println("Row is empty ");
                       insert = true;
                    }
                    else {
                        counterRow++;
                    }
                }
                if (insert==true) {
                    int count = 0;
                    for (Object datum : row) {
                        menuDetailsTable.setValueAt(datum, counterRow,
count);
                        count++;
                    }
                }
                else {
                    model.addRow(row);
                }
            }
        }
        catch (Exception e) {
            JOptionPane.showMessageDialog(this,"File not
found.","Error",JOptionPane.ERROR_MESSAGE);
        }
    }
    private void btnSearchActionPerformed(java.awt.event.ActionEvent evt) {
        sort();
        if(txtSearchPrice.getText().equals("")) {
            JOptionPane.showMessageDialog(this,"Please enter a price to
search.","Error",JOptionPane.ERROR_MESSAGE);
        }
        else {
            int key = Integer.parseInt(txtSearchPrice.getText());

            int low =0;

            int high = arraysize-1;

            int found = searchList(low,high,key);

            if (found == -1) {

                JOptionPane.showMessageDialog(this, "The searched dish is not 
found!","Alert",JOptionPane.WARNING_MESSAGE);

            }

            else {

                String tosearch=txtSearchPrice.getText();

                for (int i=0; i<menuDetailsTable.getRowCount();i++) {

                    String 
foundval=menuDetailsTable.getValueAt(i,4).toString();

                    if(foundval.equals(tosearch)) {

                        JOptionPane.showMessageDialog(this,

                                "The Item that you searched for has the 
Number: "+menuDetailsTable.getValueAt(i,0).toString()+".\n"+

                                "Name: 
"+menuDetailsTable.getValueAt(i,3).toString()+" 
"+menuDetailsTable.getValueAt(i,1).toString()+

                                ", a Type of 
"+menuDetailsTable.getValueAt(i,2).toString()+

                                ", having the price of 
"+menuDetailsTable.getValueAt(i,4).toString()+".",

                                "Info",JOptionPane.INFORMATION_MESSAGE);

                    }

                }

            }

        }

    }

    public void sort() {

        int a =0;

        int temp =0;

        for (int i=0; i<menuDetailsTable.getRowCount(); i++) {

            String s = (String)menuDetailsTable.getValueAt(a,4);
            a++;
            if(s!=null && s.length()!=0 ) {
                arraysize++;
            }
        }
        getPrice = new int[arraysize];
        int i=0;
        a=0;
        do {
            String s1 = (String)menuDetailsTable.getValueAt(a,4);
            if(s1!=null && s1.length()!=0 ) {
                getPrice[i]=Integer.parseInt(s1);
                i++;
            }
            a++;
        }
        while(a<menuDetailsTable.getRowCount());
        SelectionSorter.sort(getPrice);
    }
    public int searchList(int low, int high, int key) {
        if (low<=high) {
            int mid =(low+high)/2;
            if (getPrice[mid]==key) {
                return getPrice[mid];
                }
            else if(getPrice[mid]>key) {
                return searchList(low,mid-1,key) ;
                }
            else if(getPrice[mid]<key) {
                return searchList(mid+1,high,key);
                }
        }
        return -1;

    }


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

        // TODO add your handling code here:

    }


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

        // TODO add your handling code here:

        String DishNumber=txtDishNum.getText();

        String DishName=txtDishName.getText();

        String Price=txtPrice.getText();

        String DishType=cbType.getSelectedItem().toString();

        int value= cbType.getSelectedIndex();

        if("[a-z]".equals(DishNumber) ||
"".equals(DishNumber))JOptionPane.showMessageDialog(this,"Please enter Dish
Number.","Invalid Input",JOptionPane.ERROR_MESSAGE);

        else
if("".equals(DishName))JOptionPane.showMessageDialog(this,"Please enter Dish
Name.","Invalid Input",JOptionPane.ERROR_MESSAGE);

        else if(value==0) JOptionPane.showMessageDialog(this,"Please select
Dish Type.","Invalid Input",JOptionPane.ERROR_MESSAGE);

        else if(buttonGroup1.getSelection()==null)
JOptionPane.showMessageDialog(this,"Please select the Diet
Preference.","Invalid Input",JOptionPane.ERROR_MESSAGE);

        else if("".equals(Price)) JOptionPane.showMessageDialog(this,"Please
Enter Dish Price.","Invalid Input",JOptionPane.ERROR_MESSAGE);

        else{

            String Diet="";

            if(veg.isSelected()){

                Diet=veg.getText();

            }

            else if(nonVeg.isSelected()){

                Diet=nonVeg.getText();

            }
            else if(drink.isSelected()){
                Diet=drink.getText();
            }
            int insertPrice = Integer.parseInt(Price);
            priceArray.add(insertPrice);
            typeArray.add(DishType);
            try{
                String[]
dishDetails={DishNumber,DishName,DishType,Diet,Price};
                int rowCount=menuDetailsTable.getRowCount();
                int columnCount=menuDetailsTable.getColumnCount();
                int nextRow=0;
                boolean emptyRowFlag = false;
                String s;
                do{
                    s=(String)menuDetailsTable.getValueAt(nextRow,0);
                    if(s !=null && s.length() !=0)nextRow++;
                    else emptyRowFlag = true;
                    txtDishNum.setText("");
                    txtDishName.setText("");
                    txtPrice.setText("");
                    buttonGroup1.clearSelection();
                    cbType.setSelectedIndex(0);
                }
                while (nextRow < rowCount && !emptyRowFlag);
                    for (int i=0; i<columnCount; i++){

menuDetailsTable.setValueAt(dishDetails[i],nextRow,i);
                    }
            }
            catch(Exception e){
                JOptionPane.showMessageDialog(this,"Your Table has limited
rows to save data.","Alert",JOptionPane.WARNING_MESSAGE);
```

```
            }
        }
    }
    private void btnClearActionPerformed(java.awt.event.ActionEvent evt) {
        txtDishNum.setText("");
        txtDishName.setText("");
        txtPrice.setText("");
        buttonGroup1.clearSelection();
        cbType.setSelectedIndex(0);
    }
    private void btnExitActionPerformed(java.awt.event.ActionEvent evt) {
        System.exit(0);
    }
    private void 
btnItemTypeAvailableActionPerformed(java.awt.event.ActionEvent evt) {
        availableItem();
    }
    private void availableItem(){
        String type=cbAvailableItem.getSelectedItem().toString();
        int rowCount = menuDetailsTable.getRowCount();
        for (int i=0; i<rowCount;i++){
            String s= menuDetailsTable.getValueAt(i,2).toString();
            if (type.equals(s)){
                String a= menuDetailsTable.getValueAt(i,1).toString();
                String b= menuDetailsTable.getValueAt(i,3).toString();
                JOptionPane.showMessageDialog(this,
                        "The availabe item of this type is "+b+" "+a+".",
                        "Info",
                        JOptionPane.INFORMATION_MESSAGE);
            }
        }
    }
    private void cbAvailableItemActionPerformed(java.awt.event.ActionEvent
evt) {

        // TODO add your handling code here:

    }


    private void txtSearchPriceKeyTyped(java.awt.event.KeyEvent evt) {

        // TODO add your handling code here:

                  char sf= evt.getKeyChar();

        if(!(Character.isDigit(sf) || (sf==KeyEvent.VK_BACK_SPACE) ||
sf==KeyEvent.VK_DELETE)){

            evt.consume();

        }

    }


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

        openData();

    }


    private void txtPriceKeyTyped(java.awt.event.KeyEvent evt) {

        // TODO add your handling code here:

        char dishP= evt.getKeyChar();

        if(!(Character.isDigit(dishP) || (dishP==KeyEvent.VK_BACK_SPACE) ||
dishP==KeyEvent.VK_DELETE)){

            evt.consume();

        }

    }


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

        // TODO add your handling code here:

    }


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

        // TODO add your handling code here:
    }


    private void txtDishNameKeyTyped(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        char dishN= evt.getKeyChar();
        if((Character.isDigit(dishN) || (dishN==KeyEvent.VK_BACK_SPACE) ||
dishN==KeyEvent.VK_DELETE)){
            evt.consume();
        }
    }


    private void txtDishNameActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }


    private void txtDishNumKeyTyped(java.awt.event.KeyEvent evt) {
        // TODO add your handling code here:
        char dn= evt.getKeyChar();
        if(!(Character.isDigit(dn) || (dn==KeyEvent.VK_BACK_SPACE) ||
dn==KeyEvent.VK_DELETE)){
            evt.consume();
        }
    }


    private void txtDishNumActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }


    private void nonVegActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }


    private void drinkActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void vegActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void menuItemHelpActionPerformed(java.awt.event.ActionEvent evt)
{
        try{
            File helpfile = new File ("src\\help.pdf");
            Runtime.getRuntime().exec("rundll32 url.dll,FileProtocolHandler
"+ helpfile);
        }
        catch (Exception e){
            JOptionPane.showMessageDialog(this,"File not
found."+e,"Error",JOptionPane.ERROR_MESSAGE);
        }
    }

    private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
        System.exit(0);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting
code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with
the default look and feel.
```

```
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

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

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

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

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

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new MenuInfo().setVisible(true);
            }
        });
    }
    // Variables declaration - do not modify
    private javax.swing.JButton btnClear;
    private javax.swing.JButton btnExit;
    private javax.swing.JButton btnInsert;
    private javax.swing.JButton btnItemTypeAvailable;
    private javax.swing.JButton btnSearch;
    private javax.swing.ButtonGroup buttonGroup1;
    private javax.swing.JComboBox<String> cbAvailableItem;
    private javax.swing.JComboBox<String> cbType;
    private javax.swing.JRadioButton drink;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel13;
    private javax.swing.JLabel jLabel14;
    private javax.swing.JLabel jLabel15;
    private javax.swing.JLabel jLabel16;
    private javax.swing.JLabel jLabel17;
    private javax.swing.JLabel jLabel18;
    private javax.swing.JLabel jLabel19;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel20;
    private javax.swing.JLabel jLabel21;
    private javax.swing.JLabel jLabel22;
    private javax.swing.JLabel jLabel23;
    private javax.swing.JLabel jLabel24;
    private javax.swing.JLabel jLabel25;
    private javax.swing.JLabel jLabel26;
    private javax.swing.JLabel jLabel27;
    private javax.swing.JLabel jLabel28;
    private javax.swing.JLabel jLabel29;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel30;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JMenuItem jMenuItem1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JPanel jPanel4;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JSeparator jSeparator2;
    private javax.swing.JSeparator jSeparator3;
    private javax.swing.JSeparator jSeparator4;
    private javax.swing.JSeparator jSeparator7;
    private javax.swing.JLabel lblDietPreference;
    private javax.swing.JLabel lblDishName;
    private javax.swing.JLabel lblDishNum;
    private javax.swing.JLabel lblPrice;
    private javax.swing.JLabel lblSearchPrice;
    private javax.swing.JLabel lblType;
    private javax.swing.JTable menuDetailsTable;
    private javax.swing.JMenu menuFile;
    private javax.swing.JMenu menuHelp;

    private javax.swing.JMenuItem menuItemHelp;

    private javax.swing.JMenuItem menuItemOpen;

    private javax.swing.JRadioButton nonVeg;

    private javax.swing.JTextField txtDishName;

    private javax.swing.JTextField txtDishNum;

    private javax.swing.JTextField txtPrice;

    private javax.swing.JTextField txtSearchPrice;

    private javax.swing.JRadioButton veg;

    // End of variables declaration



}
```