COVER PAGE

OBJECT ORIENTED PROGRAMMING

CCS20704

PRACTICAL LAB REPORT

OBJECT ORIENTED PROGRAMMING

CCS20704

Received Date : **04 March 2025**

Submission Due Date: 12 May 2025

Lecturer : **Dr. Jamal Abdullahi Nuh**

Weightage : **30%**

Semester : **FEBRUARY 2025**

Instruction to students:

- This is a GROUP assignment.
- Complete this cover sheet and attach it to your assignment (first page).

Student declaration:		
I declare that: • ***This assignment is my/our own work.*** • ***I/we understand what is meant by plagiarism.*** • ***My lecturer has the right to deduct my marks in case of:*** - ***Late submission*** - ***Any plagiarism found in my assignment.***		
NAME: MUHAMMAD HARISH HAQIM BIN ADNAN **MATRICS NO:** 012025020434 	**NAME:** MUHAMMAD ISA BIN ABDUL RAHIM **MATRIC NO:** 012025021044 	**NAME:** JAIKARAN A/L THANGAVELU **MATRIC NO:** 012025021305 

Contents

1.0 Introduction

In Object Oriented Programming Assignment 3, we have created a complete Java GUI application by using Apache Netbeans IDE 25 and Javax Swing libraries about computer store name “AlphaTechsz” that is used for adding products such as computer, software system, and IO device to the inventory and to the order, and the user can also check out the order. Furthermore, the user can also modify the data of the products, delete the products, and add new products. The GUI application also has login and logout functionality and the profile menu where user can edit its details. In this assignment, we will show the program codes and the program outputs of the practical implementation of Java GUI application about computer store “AlphaTechsz”. Various GUI elements have been implemented inside this Java GUI application project such as ImageIcon, text fields, buttons, message boxes, dropdown menu, tables, labels, and more.

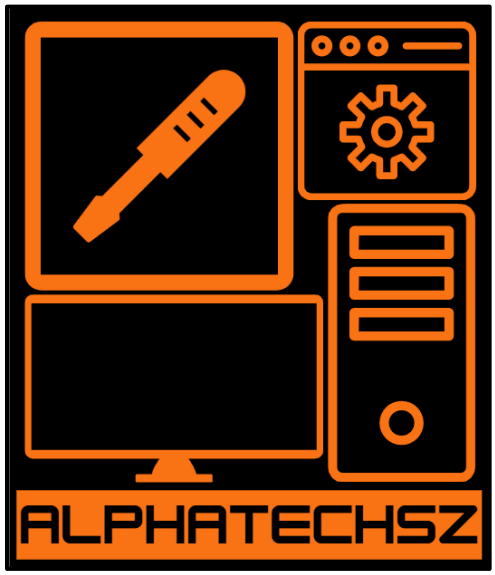


Figure 1-1: Logo of “AlphaTechsz” computer store that will be used for ImageIcon of GUI application

2.0 Google Drive Link to Download the Source Codes

Link:

https://drive.google.com/drive/folders/1Mr5KA2YEyCNC0T69iqeQMWsA5T7B1msj?usp=sharing

Download the “ComputerStore” folder and run it using Apache Netbeans IDE 25.

3.0 Program Codes

3.1 Main.java

```java
// Package name (Group ID and the project name)
package com.alphatechsz.computerstore;
/* Group ID: com.alphatechsz
*  Store Name: Alpha Techsz
*  Project Name: ComputerStore
*  Purpose: To create a Java GUI application about computer store
*/
// Main class (To run the program codes of this project)
public class Main
{
    // Main method
    public static void main(String[] args)
    {
        // Set user details
        User.setID(newID: 10);
        User.setName(newName:"Harish Haqim");
        User.setEmail(newEmail: "harishhaqim@outlook.com");
        User.setPassword(newPassword:"12345");
        // Set pre-data inventory products
        Inventory.products.add(new Computer(ID: 1, brand: "HP", model: "Victus 16", type: "Laptop", size: "16
        Inventory.products.add(new Computer(ID: 2, brand: "HP", model: "Victus 15", type: "Laptop", size: "15
        Inventory.products.add(new SoftwareSystem(ID: 3, brand: "Microsoft", model: "Windows 11 Home", type
        Inventory.products.add(new SoftwareSystem(ID: 4, brand: "Microsoft", model: "Windows 11 Profession
        Inventory.products.add(new IODevice(ID: 5, brand: "Razer", model: "Hyper-X", type: "Mouse", size: "2x6
        Inventory.products.add(new IODevice(ID: 6, brand: "Aula", model: "120 RGB Board", type: "Keyboard",
        // GUI begins with login menu
        LoginMenu loginMenu = new LoginMenu();
        // Set login menu window to visible
        loginMenu.setVisible(b: true);
    }
}
```

Figure 3-1: Program codes of Main.java

```java
// Set user details
User.setID(newID: 10);
User.setName(newName:"Harish Haqim");
User.setEmail(newEmail: "harishhaqim@outlook.com");
User.setPassword(newPassword:"12345");
// Set pre-data inventory products
Inventory.products.add(new Computer(ID: 1, brand: "HP", model: "Victus 16", type: "Laptop", size: "16 i
Inventory.products.add(new Computer(ID: 2, brand: "HP", model: "Victus 15", type: "Laptop", size: "15 i
Inventory.products.add(new SoftwareSystem(ID: 3, brand: "Microsoft", model: "Windows 11 Home", type:
Inventory.products.add(new SoftwareSystem(ID: 4, brand: "Microsoft", model: "Windows 11 Professiona
Inventory.products.add(new IODevice(ID: 5, brand: "Razer", model: "Hyper-X", type: "Mouse", size: "2x6x
Inventory.products.add(new IODevice(ID: 6, brand: "Aula", model: "120 RGB Board", type: "Keyboard", s
```

Figure 3-2: Setting user's data and adding products to the inventory

Firstly, we have the set the user details such ID, name, email and password for authentication functionalities. Then, we also have set pre data for products inside the inventory list to show in the tables later.

```java
// GUI begins with login menu
LoginMenu loginMenu = new LoginMenu();
// Set login menu window to visible
loginMenu.setVisible(b: true);
```

Figure 3-3: Creating instance of LoginMenu.java to begin the application

After that, we have declared the object of LoginMenu.java class to start the GUI application starting from the LoginMenu.java.

Total lines of codes = 31

3.2 LoginMenu.java

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
 */
package com.alphatechsz.computerstore;

import java.awt.Color;
import javax.swing.ImageIcon;
import javax.swing.JOptionPane;

/**
 *
 * @author Haqim
 */
public class LoginMenu extends javax.swing.JFrame {
    /**
     * Creates new form LoginMenu
     */
    public LoginMenu() {
        initComponents();
        // Set background color
        getContentPane().setBackground(new Color(r: 164, g: 219, b: 232));
    }
    /**
     * 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

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

    private void exitButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // Close current window
        this.dispose();
        JOptionPane.showMessageDialog(parentComponent:null, message:"Exit successfully",
                                      title: "Exit Message", messageType:JOptionPane.INFORMATION_MESSAGE);
    }

    private void loginButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // Get data from input fields
        String emailOrId = emailOrIdInput.getText();
        char[] passwordChars = passwordInput.getPassword();
        String password = new String(value: passwordChars);
        System.out.println(emailOrId);
```

Figure 3-4: Program codes of LoginMenu.java

```java
private void loginButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // Get data from input fields
    String emailOrId = emailOrIdInput.getText();
    char[] passwordChars = passwordInput.getPassword();
    String password = new String(value: passwordChars);
    System.out.println(x: emailOrId);
    System.out.println(x: password);
    // Check if the email / id and password are equal to the user data
    if((emailOrId.equalsIgnoreCase(anotherString: User.getEmail())
            || Integer.parseInt(s: emailOrId) == User.getID())
            && password.equalsIgnoreCase(anotherString: User.getPassword()))
    {
        this.dispose();
        JOptionPane.showMessageDialog(parentComponent:null, message:"Login successfully",
                                      title: "Login Message", messageType:JOptionPane.INFORMATION_MESSAGE);
        MainMenu mainMenu = new MainMenu();
        mainMenu.setVisible(b: true);
    }
    else
    {
        JOptionPane.showMessageDialog(parentComponent:null, message:"Incorrect ID / Email or Password!",
                                      title: "Incorrect input login Message", messageType:JOptionPane.INFORMATION_MESSAGE);
    }
}
```

Figure 3-5: Program codes of login button functionality

Next, when the user presses the login button, the GUI will check whether the authentication details entered by the user is correct or not by using if and else condition.

Total lines of code: 203

3.3 MainMenu.java

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
 */
package com.alphatechsz.computerstore;

import java.awt.Color;
import javax.swing.JOptionPane;
import javax.swing.ImageIcon;
/**
 *
 * @author Haqim
 */
public class MainMenu extends javax.swing.JFrame {

    /**
     * Creates new form MainMenu
     */
    public MainMenu() {
        initComponents();
        getContentPane().setBackground(new Color(r: 164, g: 219, b: 232));
    }

    /**
     * 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
```

Figure 3-6: Program codes of MainMenu.java

```java
    private void toOrderMenuButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        this.dispose();
        OrderMenu orderMenu = new OrderMenu();
        orderMenu.setVisible(b: true);
    }

    private void logoutButtonActionPerformed(java.awt.event.ActionEvent evt) {
        this.dispose();
        JOptionPane.showMessageDialog(parentComponent:null, message:"Logout successfully",
            title: "Logout Message", messageType:JOptionPane.INFORMATION_MESSAGE);
        LoginMenu loginMenu = new LoginMenu();
        loginMenu.setVisible(b: true);
    }

    private void toSoftwareSystemMenuButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        this.dispose();
        SoftwareSystemMenu softwareSystemMenu = new SoftwareSystemMenu();
        softwareSystemMenu.setVisible(b: true);
    }

    private void toComputerMenuButtonActionPerformed(java.awt.event.ActionEvent evt) {
        this.dispose();
        ComputerMenu computerMenu = new ComputerMenu();
        computerMenu.setVisible(b: true);
    }

    private void toIoDeviceMenuActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        this.dispose();
        IODeviceMenu ioDeviceMenu = new IODeviceMenu();
        ioDeviceMenu.setVisible(b: true);
    }

    private void toProfileButtonActionPerformed(java.awt.event.ActionEvent evt) {
        this.dispose();
        ProfileMenu profileMenu = new ProfileMenu();
        profileMenu.setVisible(b: true);
    }
```

Figure 3-7: Program codes of all buttons inside the MainMenu.java

After that, the user will be transferred to the main menu where they can choose to go to the profile menu, computer menu, software system menu, io device menu, order menu, or just simply logout from the system.

Total lines of code: 224

3.4 Profile Menu

The profile menu is used for users to view its information such as ID, name, email, and password. And the user can also edit and save its new information inside menu

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
 */
package com.alphatechsz.computerstore;

import java.awt.Color;
import javax.swing.JOptionPane;
import javax.swing.ImageIcon;

/**
 *
 * @author Haqim
 */
public class ProfileMenu extends javax.swing.JFrame {

    /**
     * Creates new form ProfileMenu
     */
    public ProfileMenu() {
        initComponents();
        // Set the background color
        getContentPane().setBackground(new Color(r: 164, g: 219, b: 232));
        // Show the user's data in the window
        this.userIdField.setText(t: Integer.toString(i: User.getID()));
        this.userNameField.setText(t: User.getName());
        this.userEmailField.setText(t: User.getEmail());
        this.userPasswordField.setText(t: User.getPassword());
    }

    /**
     * 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
```

Figure 3-8: Program codes of ProfileMenu.java

```java
    private void backButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // Close the current window and back to the main menu
        this.dispose();
        MainMenu mainMenu = new MainMenu();
        mainMenu.setVisible(b: true);
    }

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

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

    }

    private void saveButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // Save and set the new data for the current user
        User.setID(newID: Integer.parseInt(s: userIdField.getText()));
        User.setEmail(newEmail: userEmailField.getText());
        User.setName(newName: userNameField.getText());
        User.setPassword(newPassword: userPasswordField.getText());
        userIdField.setText(t: Integer.toString(i: User.getID()));
        userEmailField.setText(t: User.getEmail());
        userNameField.setText(t: User.getName());
        userPasswordField.setText(t: User.getPassword());
        JOptionPane.showMessageDialog(parentComponent: null, message: "Saved successfully",
                                      title: "Saved Message", messageType: JOptionPane.INFORMATION_MESSAGE);
    }
```

Figure 3-9: Program codes of save button and back button inside ProfileMenu.java

If the user decided to go to the profile menu, they could edit their details such as ID, name, email, and password and have the function to save the new details that they input.

Total lines of code: 249

3.5 ComputerMenu.java

In ComputerMenu.java, the user has the functions to add computers, update computers, delete computers and add the computer to the order. It also displays tables to display all computers inside the inventory list and the back button to return to the main menu.

Total lines of code: 671

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
 */
package com.alphatechsz.computerstore;

/**
 *
 * @author Haqim
 */

import java.awt.Color;
import javax.swing.*;
import javax.swing.table.*;
public class ComputerMenu extends javax.swing.JFrame {

    /**
     * Creates new form ComputerMenu
     */
    public ComputerMenu() {
        initComponents();
        // Set background color
        getContentPane().setBackground(new Color(r: 164, g: 219, b: 232));
    }

    /**
     * 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

    private void computerTableMouseClicked(java.awt.event.MouseEvent evt) {
        // Declare model of computerTable
        // Set the input fields to the current selected product data
        DefaultTableModel computerTableModel = (DefaultTableModel) computerTable.getModel();
        idField.setText(t: computerTableModel.getValueAt(row:computerTable.getSelectedRow(), column: 1).toString());
        brandField.setText(t: computerTableModel.getValueAt(row:computerTable.getSelectedRow(), column: 2).toString());
        modelField.setText(t: computerTableModel.getValueAt(row:computerTable.getSelectedRow(), column: 3).toString());
        typeField.setText(t: computerTableModel.getValueAt(row:computerTable.getSelectedRow(), column: 4).toString());
        sizeField.setText(t: computerTableModel.getValueAt(row:computerTable.getSelectedRow(), column: 5).toString());
        colorField.setText(t: computerTableModel.getValueAt(row:computerTable.getSelectedRow(), column: 6).toString());
        yearField.setText(t: computerTableModel.getValueAt(row:computerTable.getSelectedRow(), column: 7).toString());
        cpuField.setText(t: computerTableModel.getValueAt(row:computerTable.getSelectedRow(), column: 8).toString());
        gpuField.setText(t: computerTableModel.getValueAt(row:computerTable.getSelectedRow(), column: 9).toString());
        motherboardField.setText(t: computerTableModel.getValueAt(row:computerTable.getSelectedRow(), column: 10).toString());
```

Figure 3-10: Program codes of ComputerMenu.java containing all codes of GUI elements and functionalities inside the menu

3.6 SoftwareSystemMenu.java

Similar like ComputerMenu.java, in SoftwareSystemMenu.java, the user has the functions to add software systems, update software systems, delete software systems and add the software systems to the order. It also displays tables to display all software systems inside the inventory list and the back button to return to the main menu.

Total lines of code: 627

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
 */
package com.alphatechsz.computerstore;

import java.awt.Color;
import javax.swing.*;
import javax.swing.table.*;

/**
 *
 * @author Haqim
 */
public class SoftwareSystemMenu extends javax.swing.JFrame {

    /**
     * Creates new form SoftwareSystemMenu
     */
    public SoftwareSystemMenu() {
        initComponents();
        // Set background color
        getContentPane().setBackground(new Color(r: 164, g: 219, b: 232));
    }

    /**
     * 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

    private void idFieldActionPerformed(java.awt.event.ActionEvent evt) {...3 lines }

    private void macFieldActionPerformed(java.awt.event.ActionEvent evt) {...3 lines }

    private void addButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        // Check if one of the fields is empty
        if(!(idField.getText().isEmpty() == true || brandField.getText().isEmpty() == true || modelField.getText().isEmpty()
                || sizeField.getText().isEmpty() == true || colorField.getText().isEmpty() == true || yearField.getText().isE
                || windowsField.getText().isEmpty() == true || macField.getText().isEmpty() == true || linuxField.getText().
            int ID = Integer.parseInt(s: idField.getText());
            String brand = brandField.getText();
            String model = modelField.getText();
            String type = typeField.getText();
            String size = sizeField.getText();
```

Figure 3-11: Program codes of SoftwareSystemMenu.java containing all of GUI elements and functionalities inside the menu

3.7 IODeviceMenu.java

Similar like ComputerMenu.java and SoftwareSystemMenu.java, in IODeviceMenu.java, the user has the functions to add IO devices, update IO devices, delete IO devices and add the IO devices to the order. It also displays tables to display all IO devices inside the inventory list and the back button to return to the main menu.

Total lines of code: 654

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
 */
package com.alphatechsz.computerstore;

/**
 *
 * @author Haqim
 */

import java.awt.Color;
import javax.swing.*;
import javax.swing.table.*;

public class IODeviceMenu extends javax.swing.JFrame {

    /**
     * Creates new form IODeviceMenu
     */
    public IODeviceMenu() {
        initComponents();
        // Set background color
        getContentPane().setBackground(new Color(r: 164, g: 219, b: 232));
    }

    /**
     * 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

    private void idFieldActionPerformed(java.awt.event.ActionEvent evt) {...3 lines }

    private void wiredFieldActionPerformed(java.awt.event.ActionEvent evt) {...3 lines }

    private void addButtonActionPerformed(java.awt.event.ActionEvent evt) {
        // Add product to the inventory and the table
        // Check if one of the fields is empty
        if(!(idField.getText().isEmpty() == true || brandField.getText().isEmpty() == true || modelField.getText().isEmpty() == true || typeField.getTex
                || sizeField.getText().isEmpty() == true || colorField.getText().isEmpty() == true || yearField.getText().isEmpty() == true || maleFiel
                || femaleField.getText().isEmpty() == true || wiredField.getText().isEmpty() == true || wirelessUsbField.getText().isEmpty() || bluetoo
                || priceField.getText().isEmpty() == true)){
            int ID = Integer.parseInt(s: idField.getText());
            String brand = brandField.getText();
```

Figure 3-12: Program codes of IODeviceMenu.java containing all GUI elements and functionalities inside the menu

3.8 OrderMenu.java

In OrderMenu.java, the user can see the list of all products that the user has added to the order. They also can see the total items inside the order, and they can apply the 10% discount on each price of the products and display the total price of all items. Besides that, in this GUI also have the dropdown menu to choose which payment method the user preferred.

Total lines of code: 377

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
 */
package com.alphatechsz.computerstore;

import java.awt.Color;
import javax.swing.*;
import javax.swing.table.*;

/**
 *
 * @author Haqim
 */
public class OrderMenu extends javax.swing.JFrame {

    /**
     * Creates new form OrderMenu
     */
    public OrderMenu() {
        initComponents();
        // Set background color
        getContentPane().setBackground(new Color(r: 164, g: 219, b: 232));
        // Set total price of all items
        double totalPrice = 0;
        for(Product product : Order.products){
            totalPrice += product.getPrice();
        }
        totalPriceField.setText(t: String.format(format: "%.2f", args: totalPrice));
        totalItemField.setText(t: String.valueOf(i: Order.products.size()));
    }

    /**
     * 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

    private void backButtonActionPerformed(java.awt.event.ActionEvent evt) {
        this.dispose();
        MainMenu mainMenu = new MainMenu();
        mainMenu.setVisible(b: true);
    }

    private void applyDiscountCheckboxActionPerformed(java.awt.event.ActionEvent evt) {
        // Apply 10% discount on each items
```

Figure 3-13: Program codes of OrderMenu.java containing all GUI elements and functionalities inside the menu

4.0 Program Outputs

4.1 LoginMenu.java

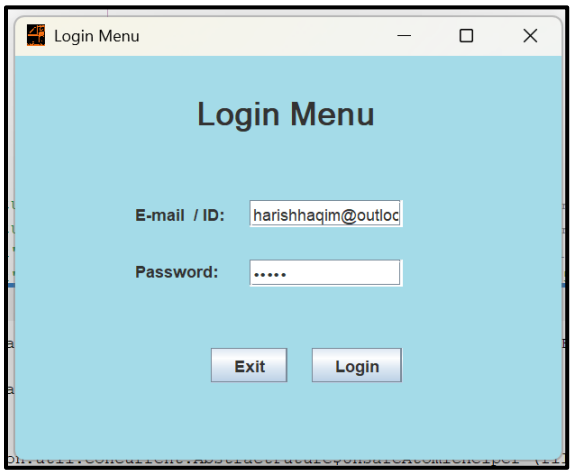


Figure 4-1: GUI interface of login menu

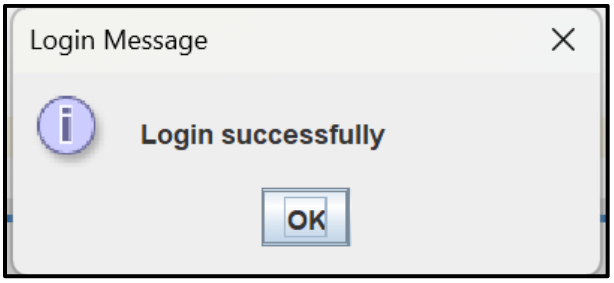


Figure 4-2: A message box stated that the user has successfully logged in

4.2 MainMenu.java

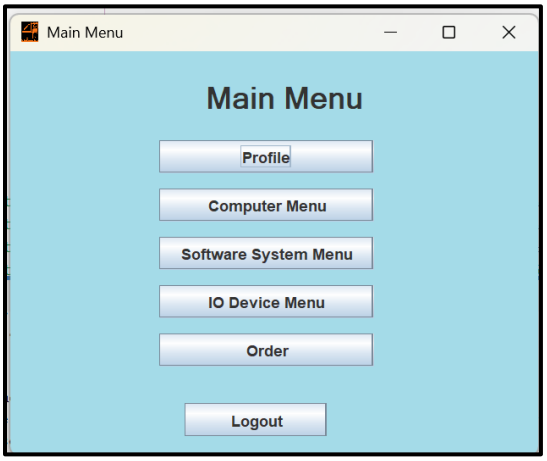


Figure 4-3: GUI interface of main menu

4.3 ProfileMenu.java

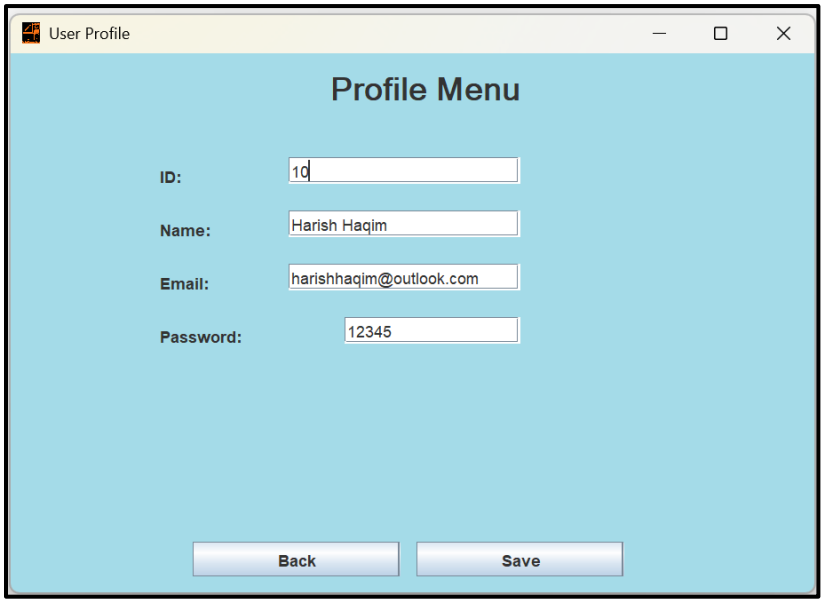


Figure 4-4: GUI interface of profile menu

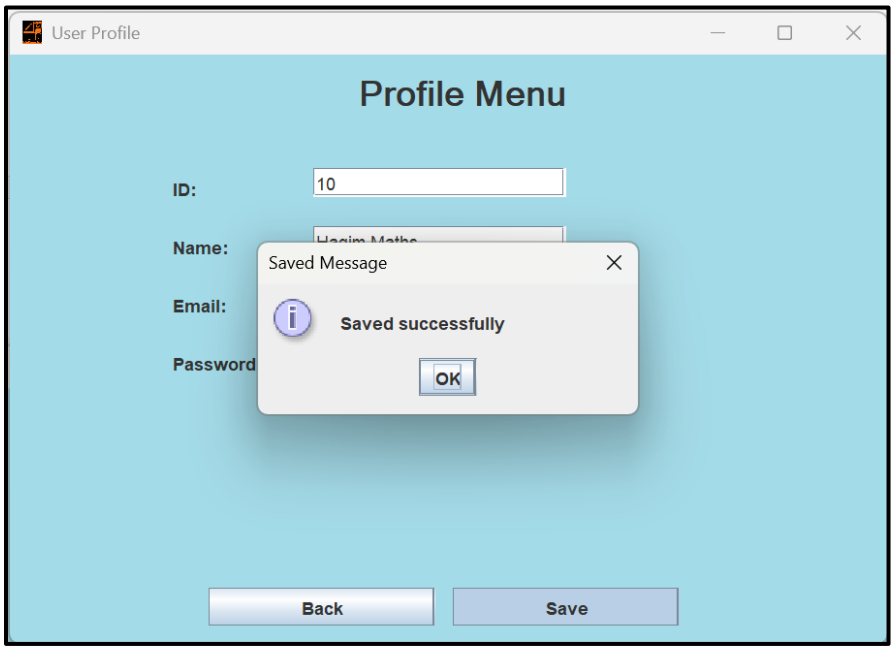


Figure 4-5: A message box stated that the user has successfully saved the new information about its details

4.4 ComputerMenu.java

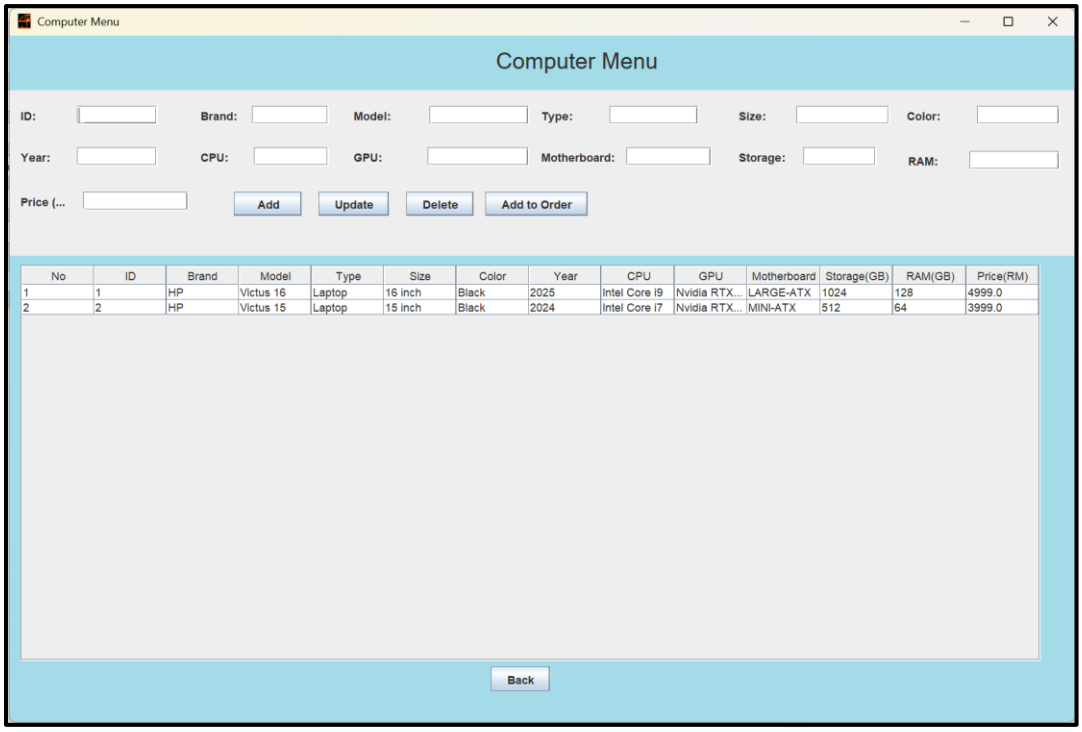


Figure 4-6: GUI interface of computer menu

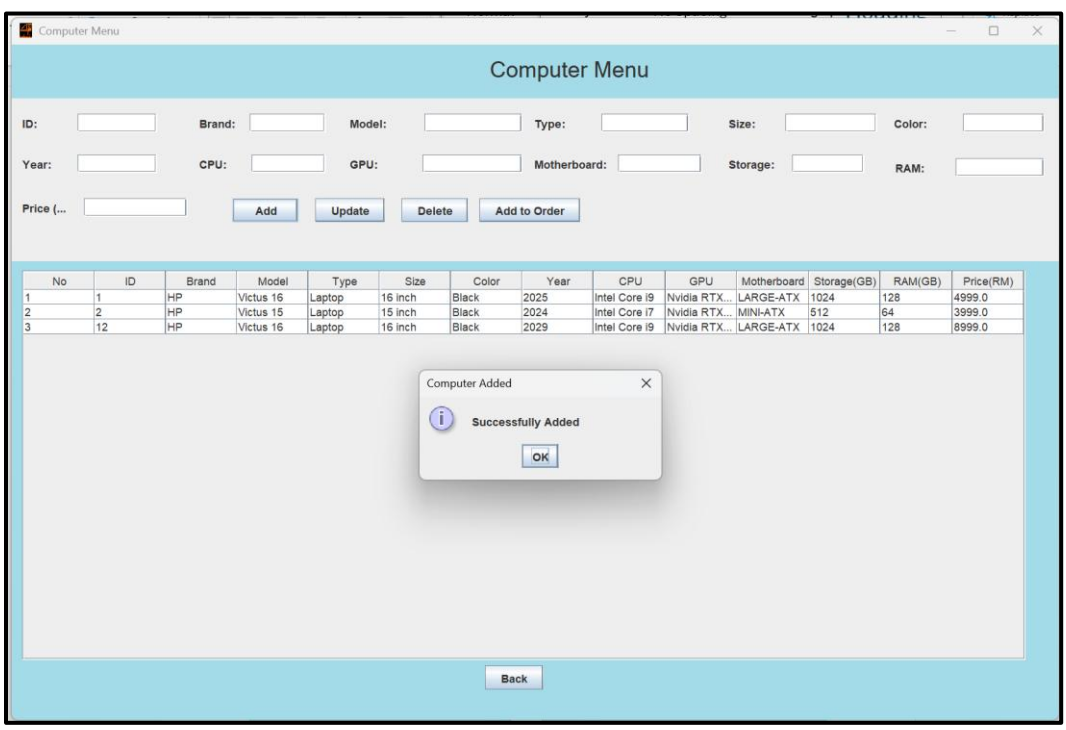


Figure 4-7: A message box pops up after the user adds a new computer

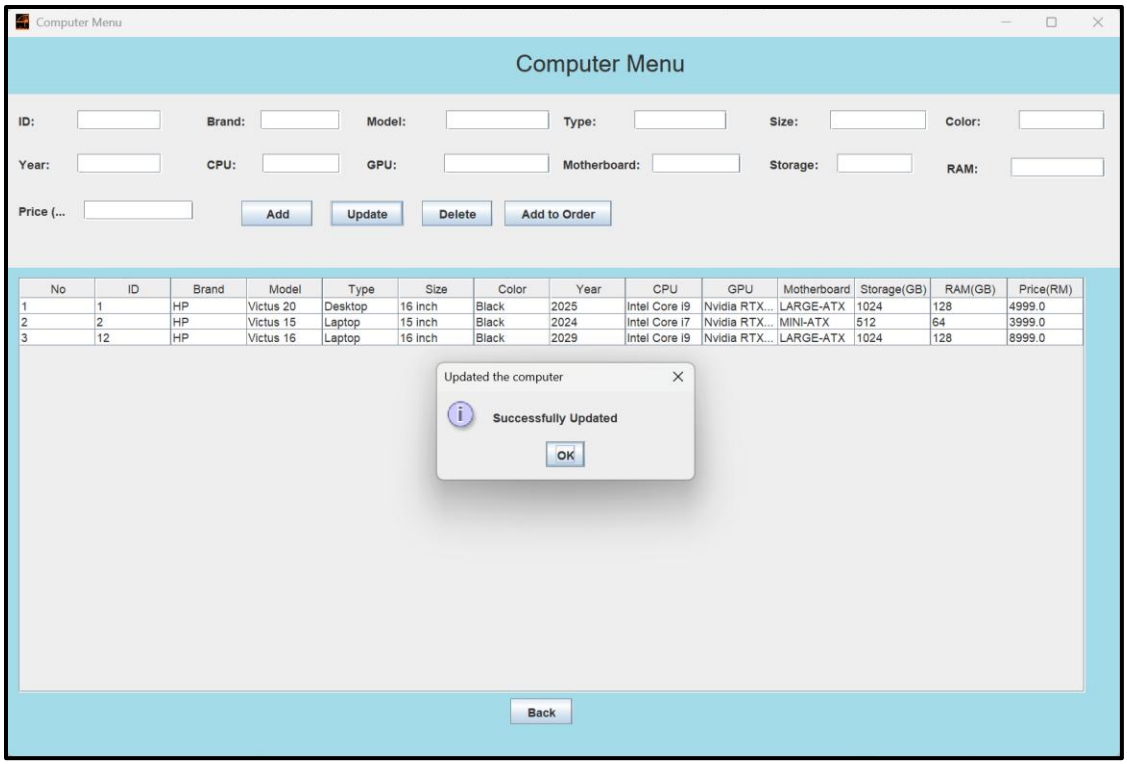


Figure 4-8: A message box pops up after the user updates the selected existing information of a computer information

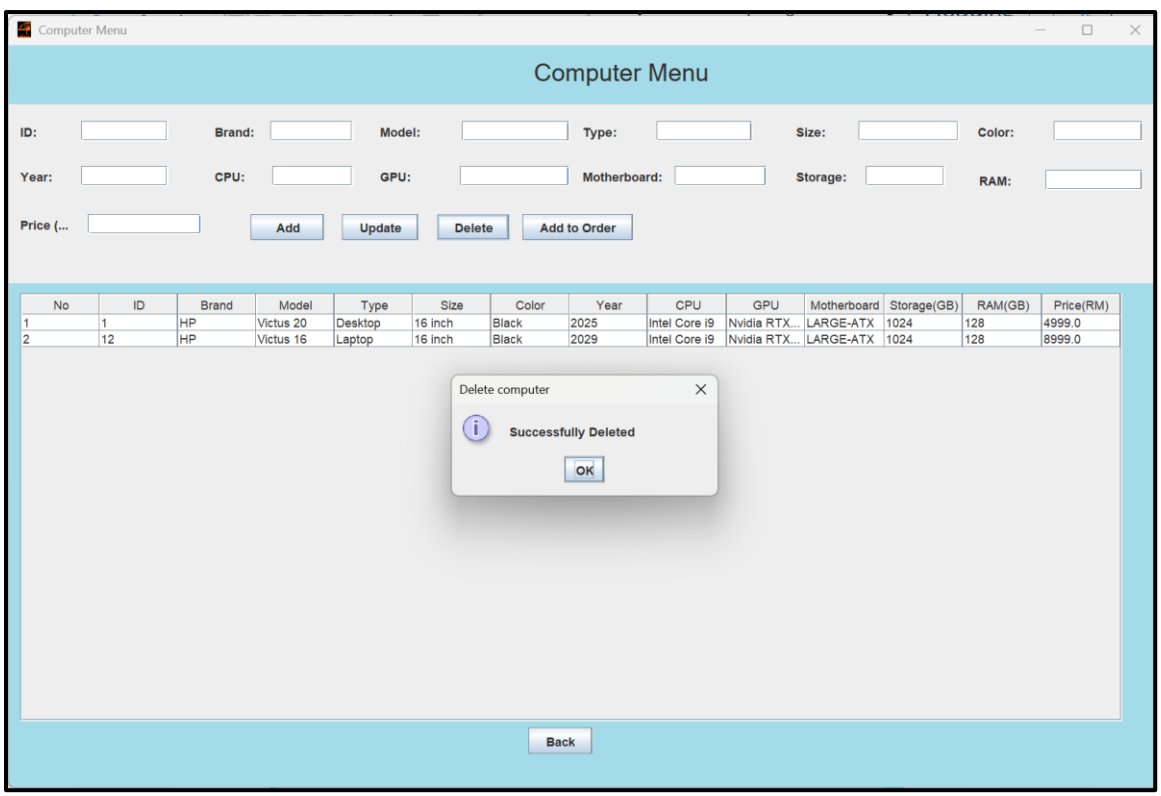


Figure 4-9: A message box pops up after the user deletes the selected computer

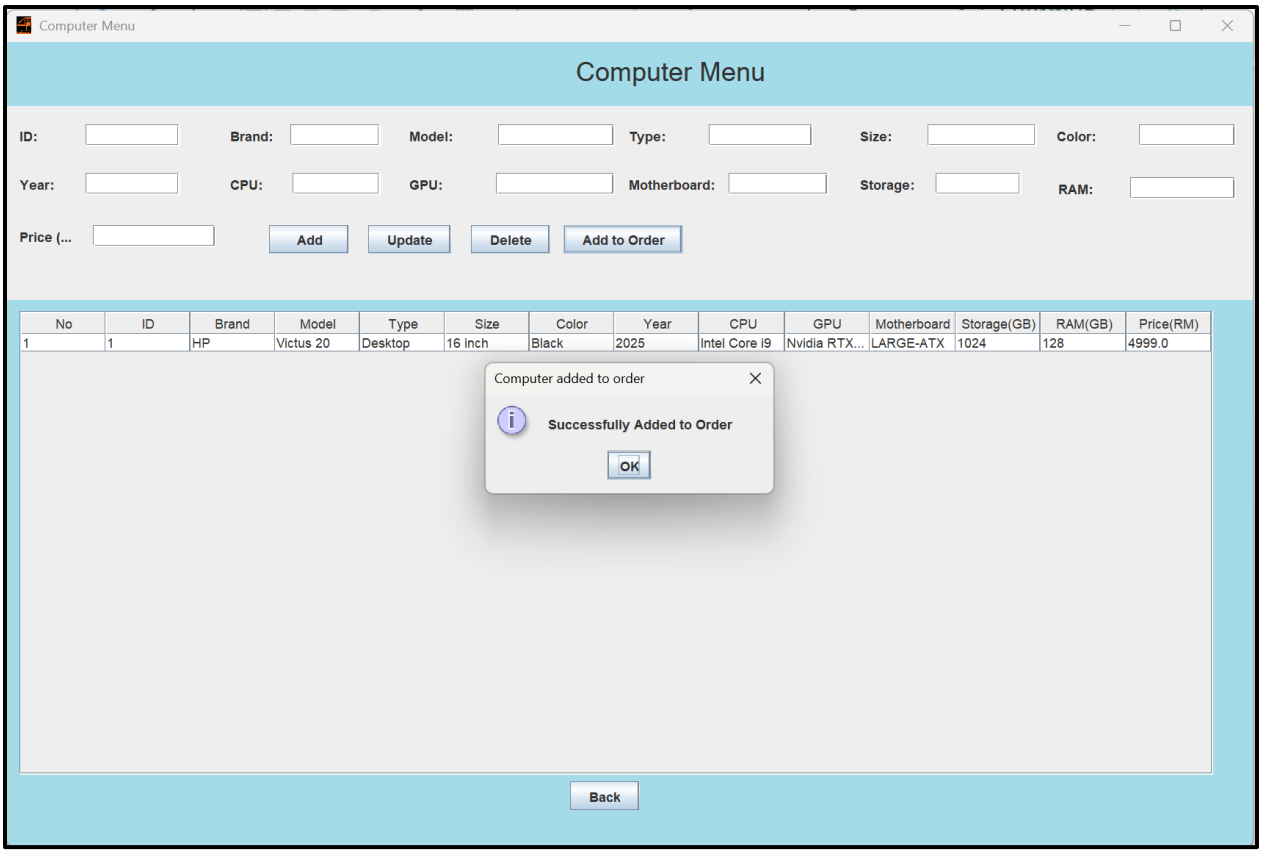


Figure 4-10: A message box pops up after the user adds the selected computer to the order

4.5 SoftwareSystemMenu.java

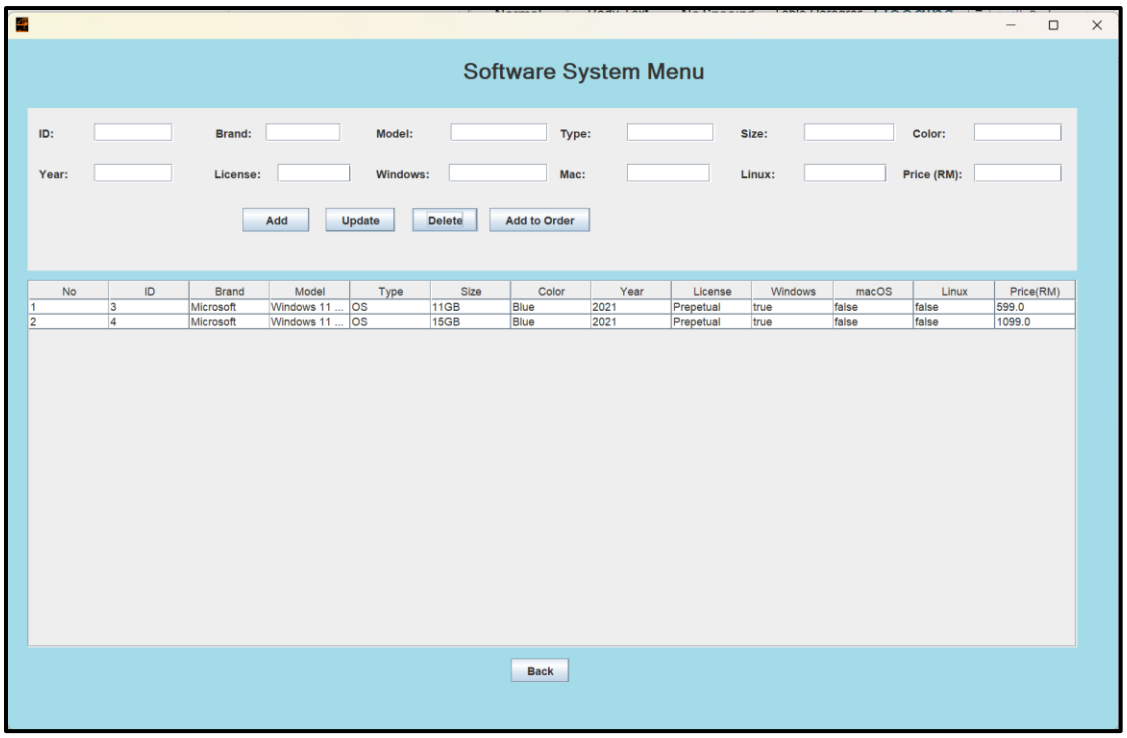


Figure 4-11: GUI interface of software system menu

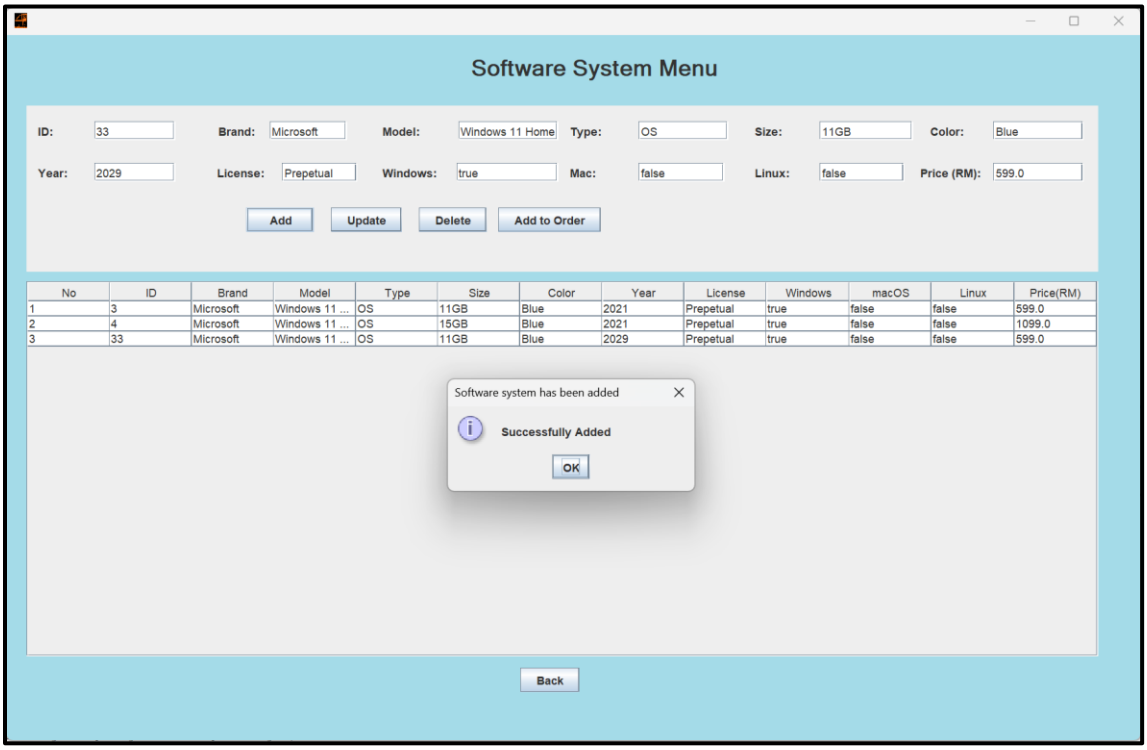


Figure 4-12: A message box pops up after the user adds a new software system

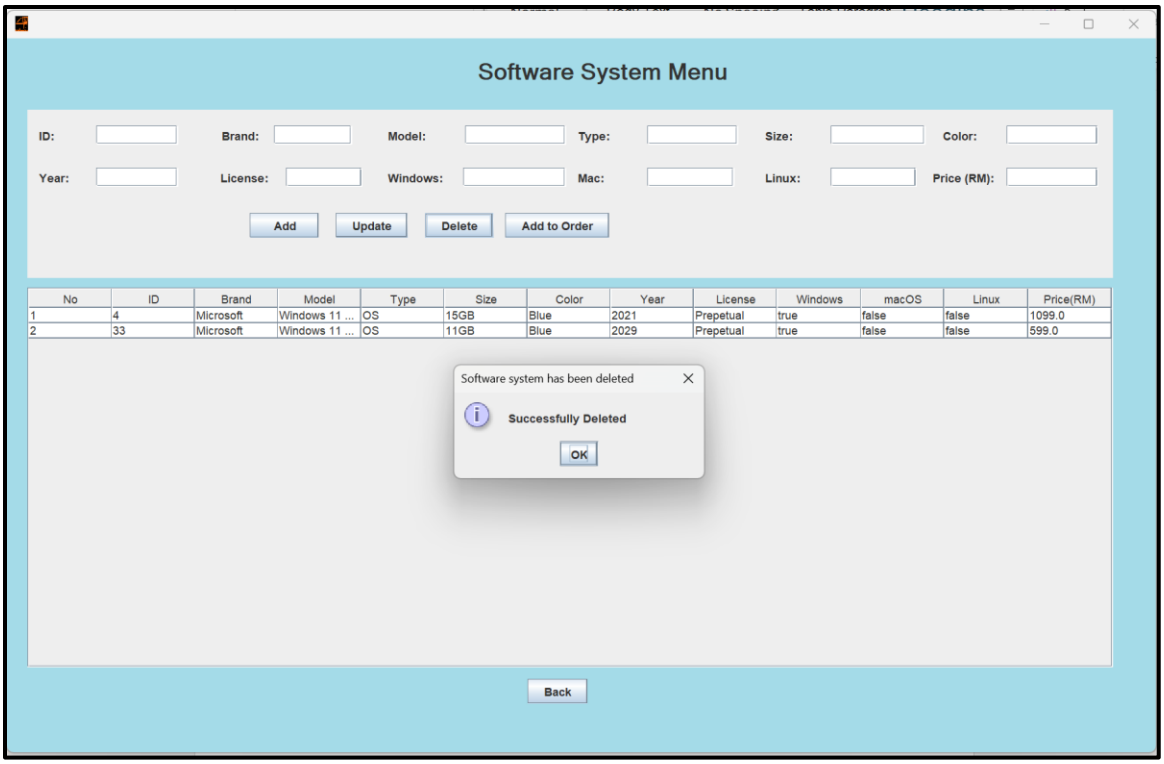


Figure 4-13: A message box pops up after the user deletes a selected software system

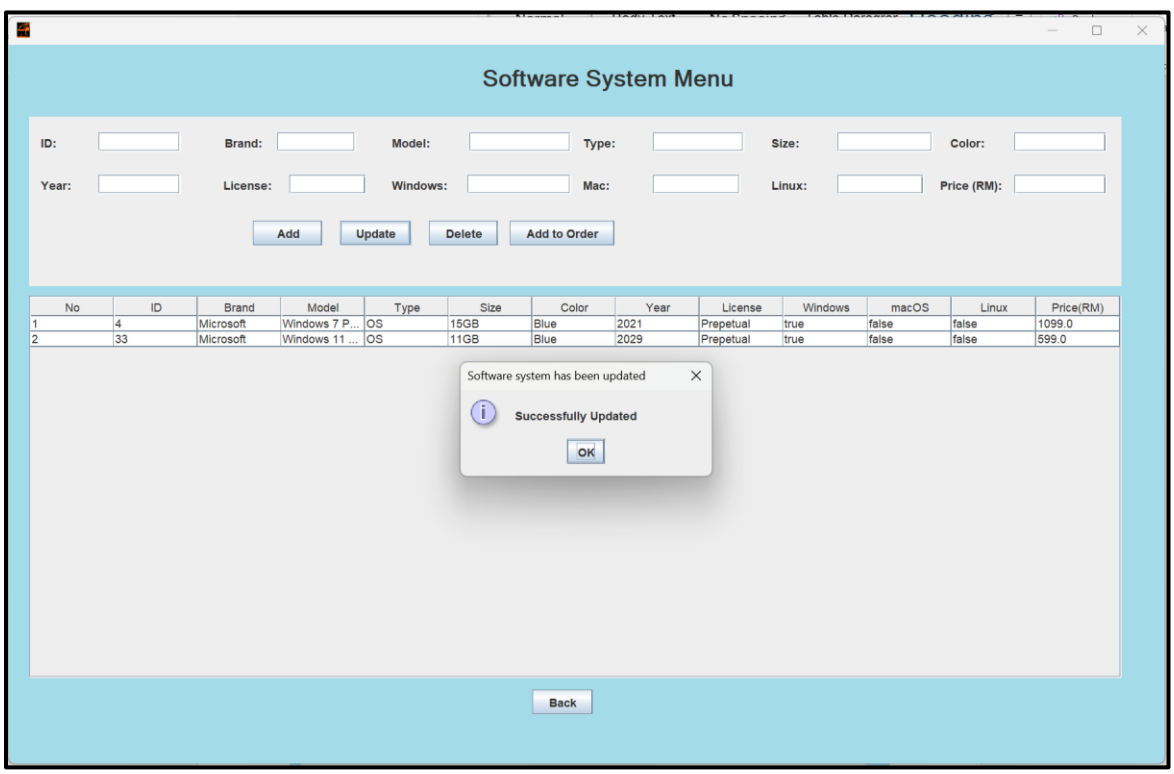


Figure 4-14: A message box pops up after the user has successfully updated the selected existing software system information

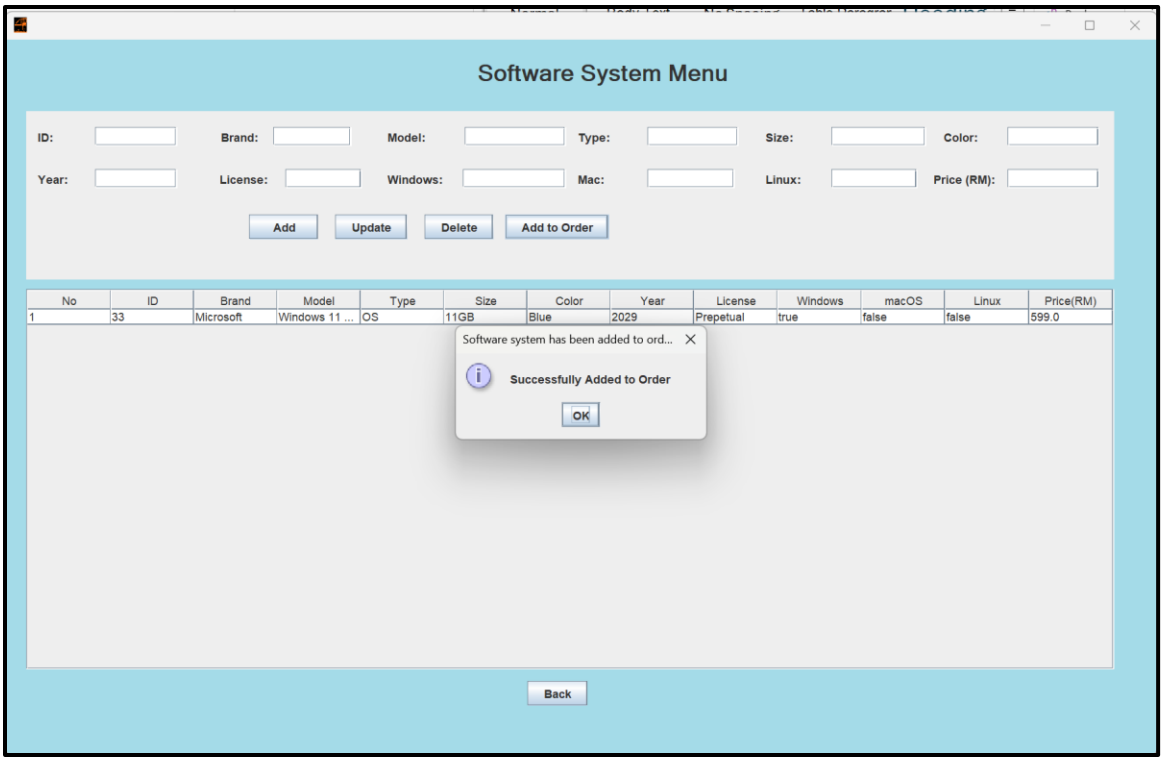


Figure 4-15: A message box pops up after the user adds selected software system to the order

4.6 IODeviceMenu.java

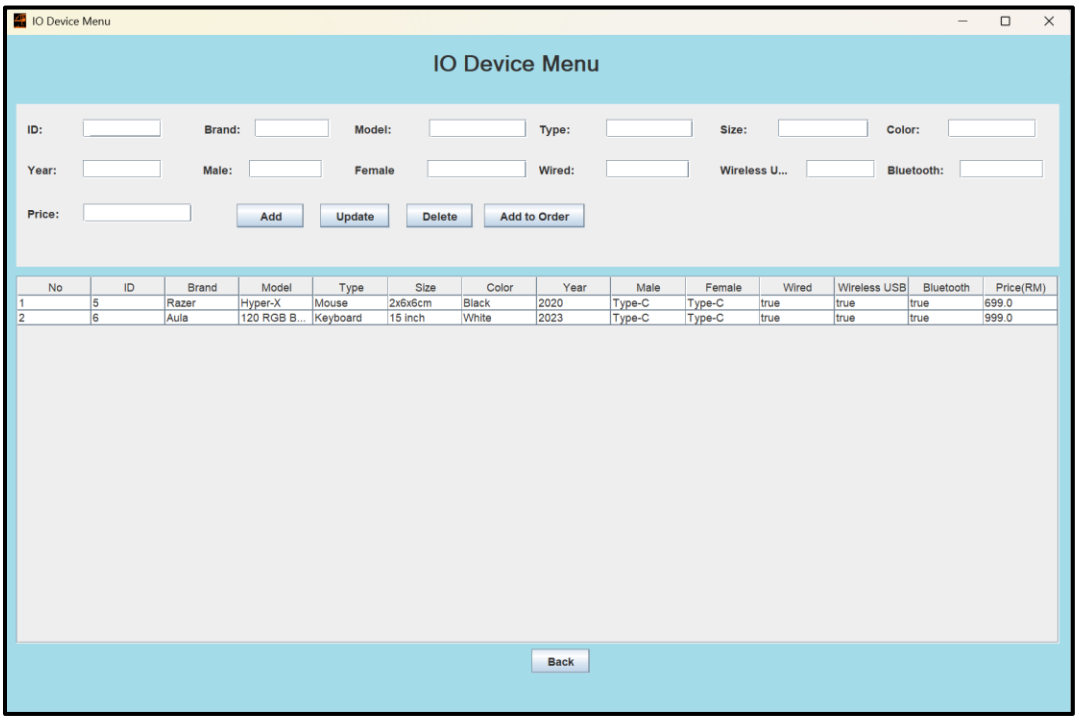


Figure 4-16: GUI interface of IO device menu

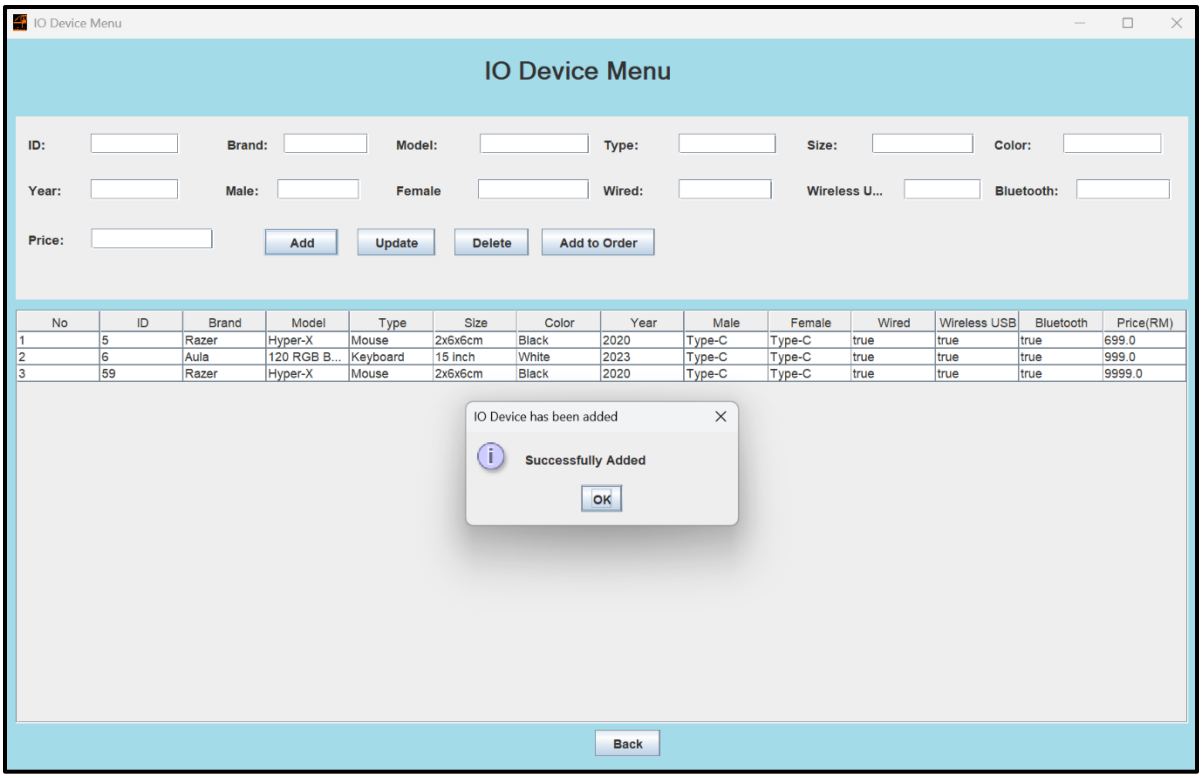


Figure 4-17: A message box pops up after the user adds a new IO device

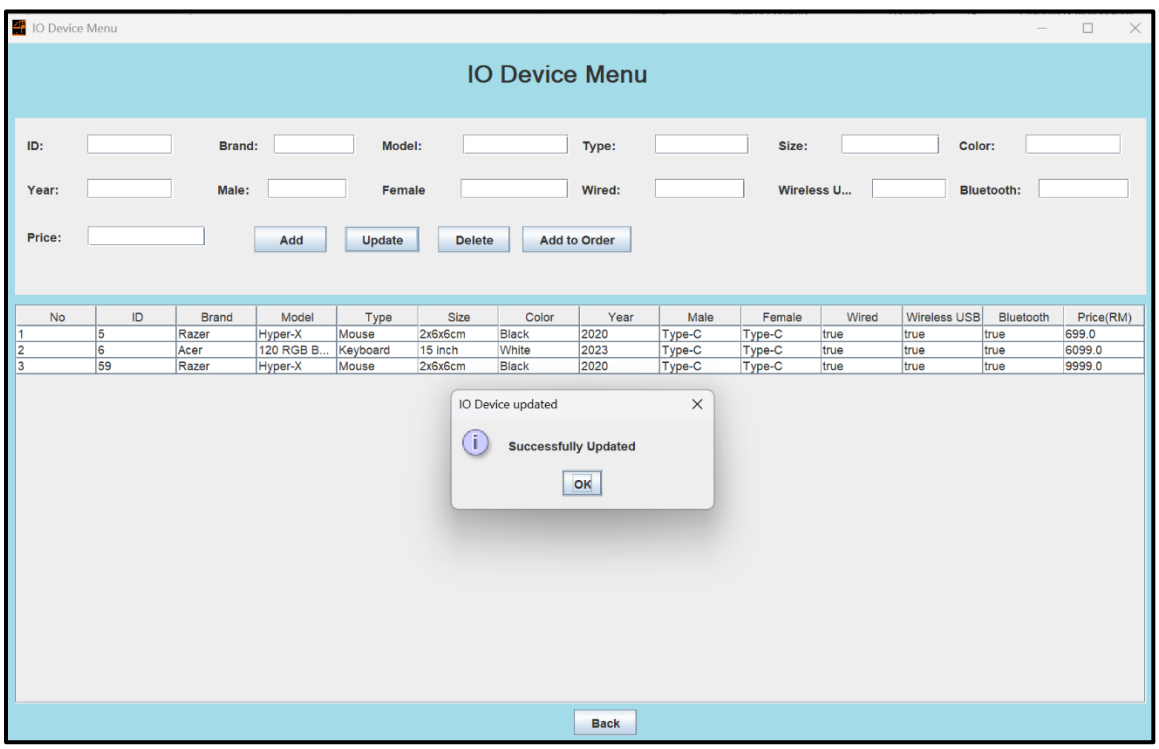


Figure 4-18: A message box pops up after the user updated the selected existing IO device information

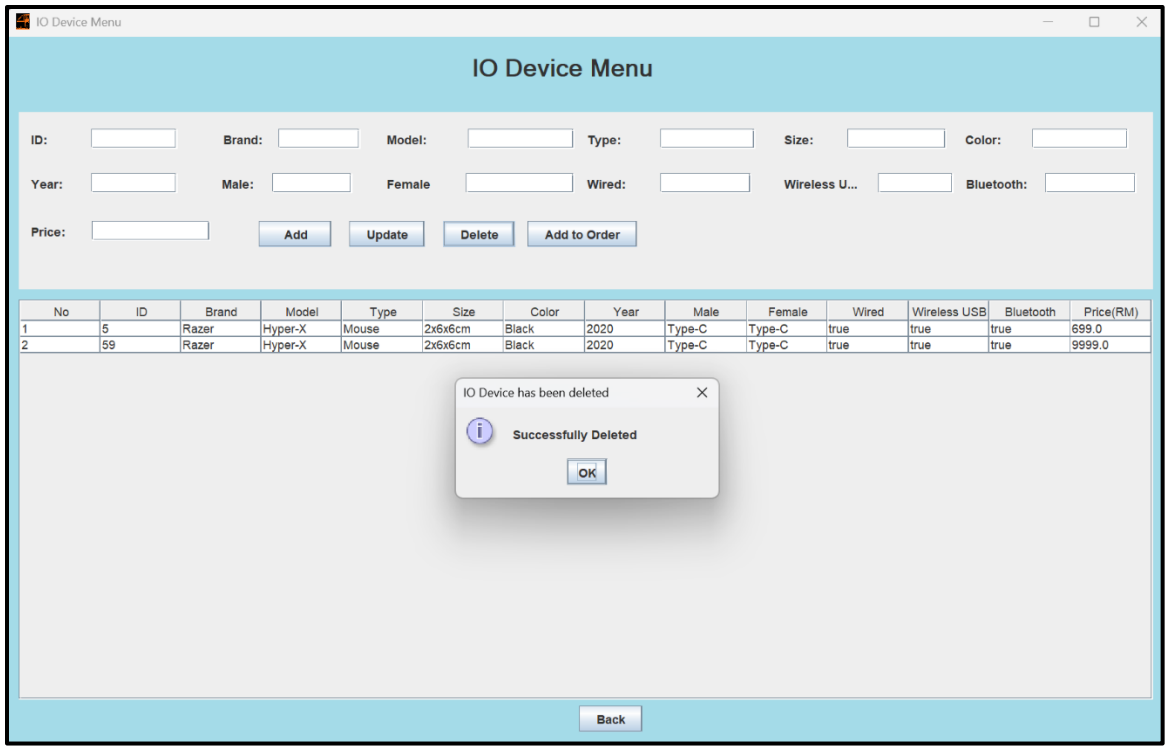
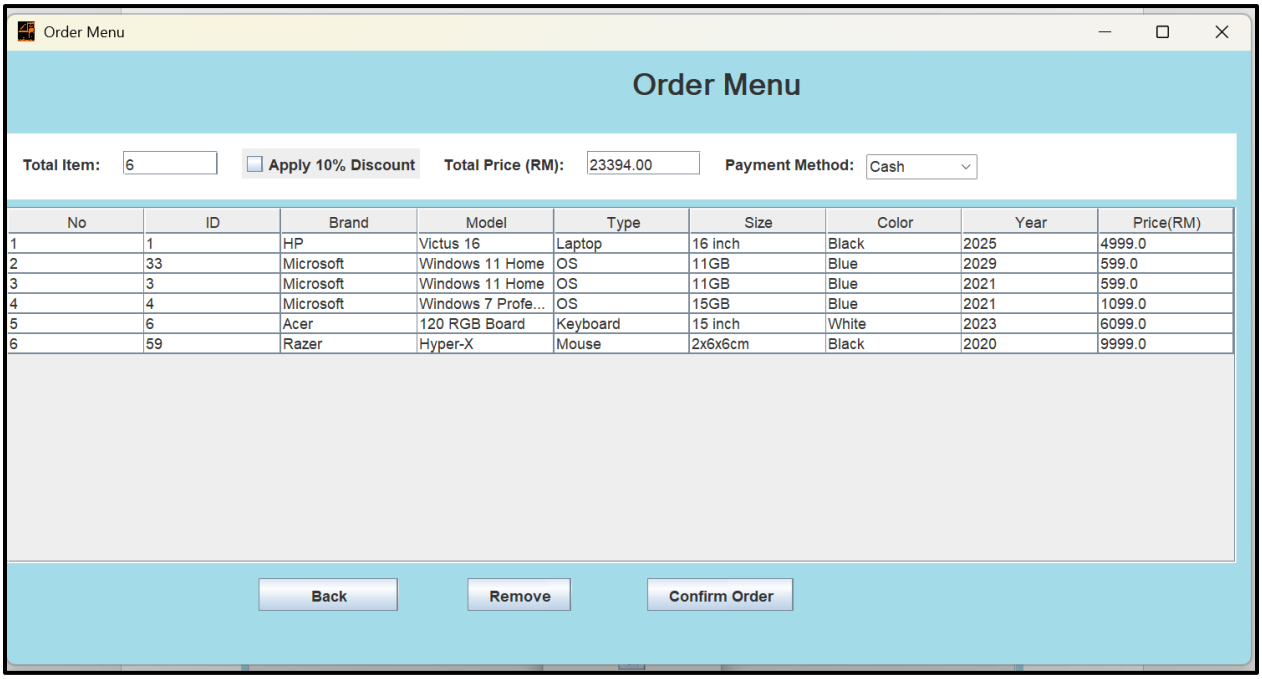


Figure 4-19: A message box pops up after the user deletes a selected IO device

Figure 4-20: A message box pops up after the user adds the IO device to the order

4.7 OrderMenu.java



No	ID	Brand	Model	Type	Size	Color	Year	Price(RM)
1	1	HP	Victus 16	Laptop	16 inch	Black	2025	4999.0
2	33	Microsoft	Windows 11 Home	OS	11GB	Blue	2029	599.0
3	3	Microsoft	Windows 11 Home	OS	11GB	Blue	2021	599.0
4	4	Microsoft	Windows 7 Profe...	OS	15GB	Blue	2021	1099.0
5	6	Acer	120 RGB Board	Keyboard	15 inch	White	2023	6099.0
6	59	Razer	Hyper-X	Mouse	2x6x6cm	Black	2020	9999.0

Figure 4-21: GUI interface of order menu

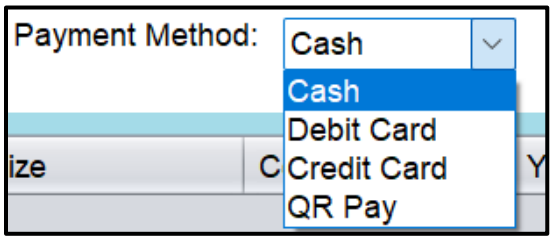


Figure 4-22: Dropdown menu of payment methods

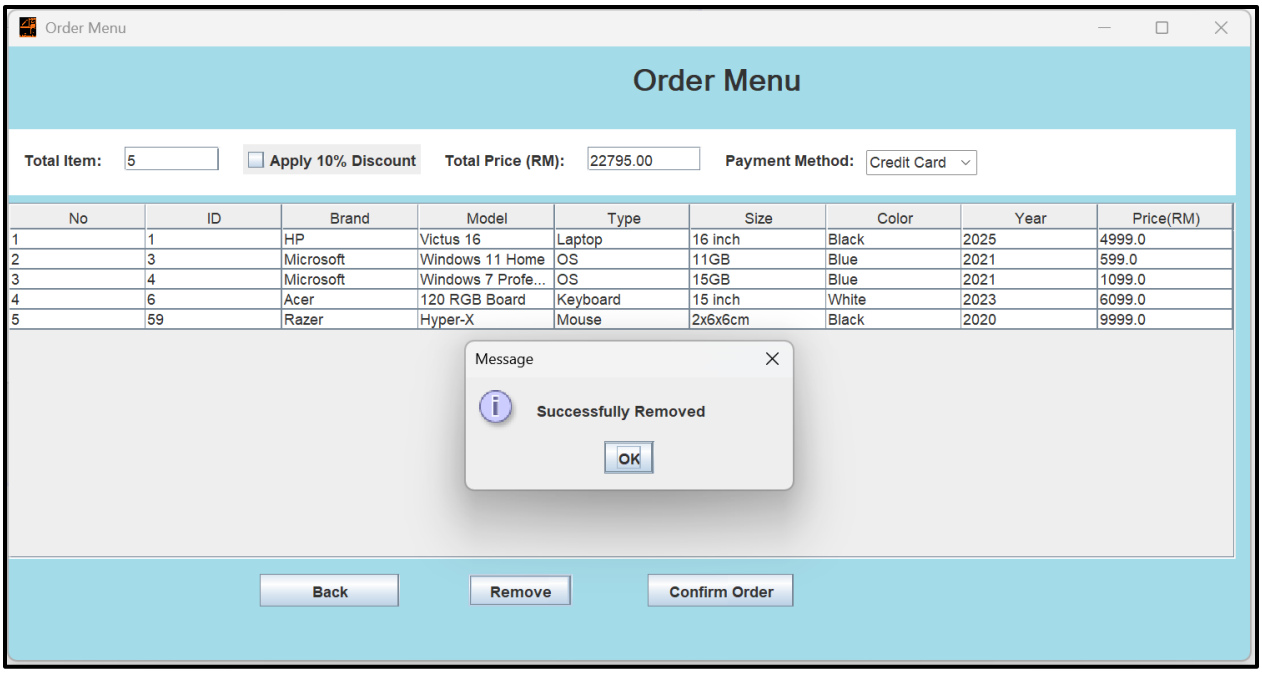


Figure 4-23: A message box pops up after the user removes a product from the order

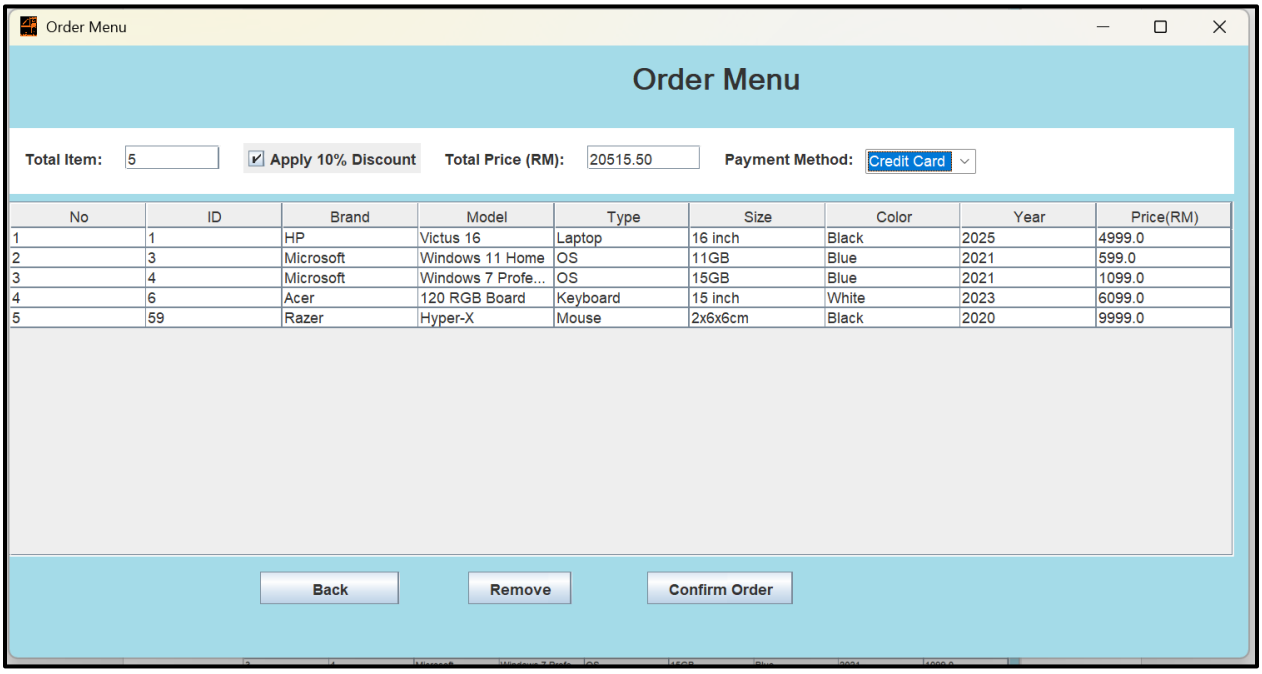


Figure 4-24: Field of total price has changed after the user applies for the 10% discount on each of the products price

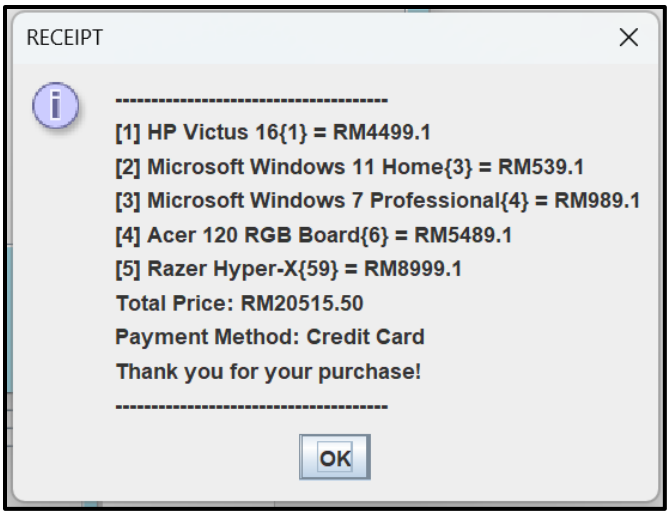


Figure 4-25: A message box pops up after the user presses the confirm order button to display receipt consisting of list of products and the payment method chosen

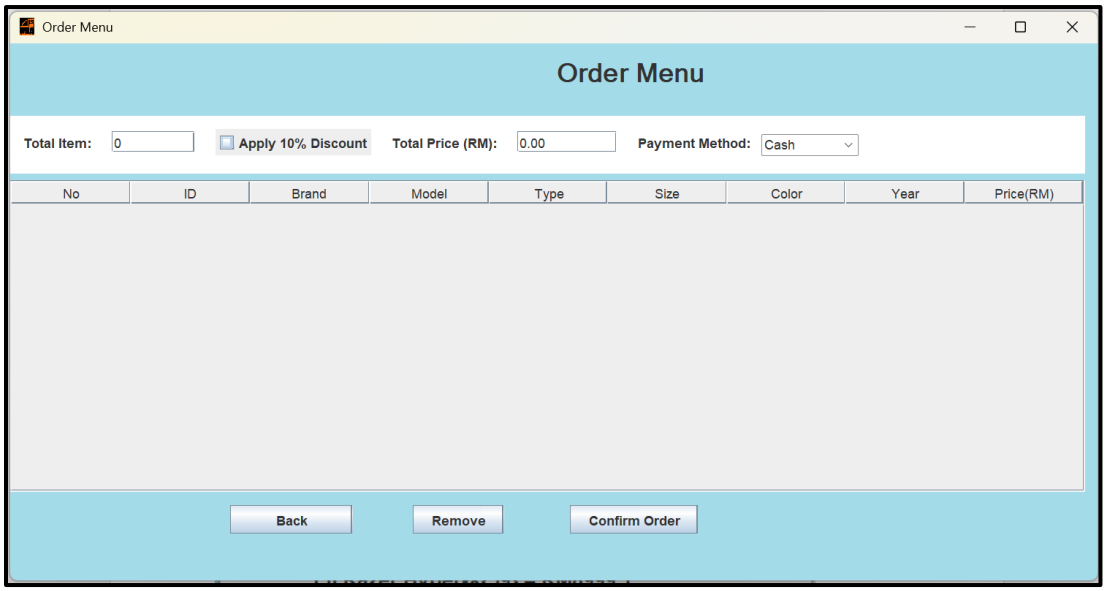


Figure 4-26: Order menu becomes empty after the user has confirmed the order

4.8 Logout Message

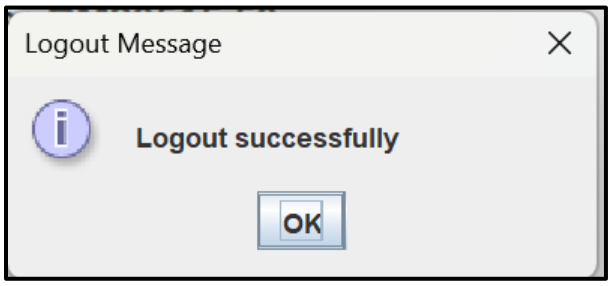


Figure 4-27: A logout message box pops up after the user logs out from the main menu

4.9 Exit Program Message

Figure 4-28: An exit message box pops up after the user exits from the system

5.0 Conclusion

In conclusion, the “AlphaTechsz” computer store GUI application is a robust and efficient system due to its various functionalities such as adding GUI elements, functions of authentication adding, modifying, removing, and products information manipulations and more. By using Apache Netbeans IDE 25 and the Javax Swing library, we can develop comprehensive and efficient GUI applications easily. The GUI application that we have created has also eased the flow of the users to do data manipulation, ordering, and authentication which can be also used in other systems in the future.

RUBRIC FOR ASSIGNMNET 3

Criteria	Excellent (4)	Good (3)	Satisfactory (2)	Needs Improvement (1)	Marks
Code Modification and Problem-Solving (20%)	Demonstrates exceptional proficiency in modifying code, executing changes accurately, efficiently, and creatively. Displays advanced problem-solving skills.	Displays proficient code modification skills with accurate implementation and good problem-solving abilities.	Demonstrates basic code modification skills with some accuracy. May encounter challenges in creative solutions and problem-solving.	Exhibits limited code modification skills with frequent errors. Struggles with problem-solving aspects.	
Practical Application (20%)	Clearly and effectively demonstrates the practical application of the modified code in the presentation. Actions align seamlessly with intended outcomes.	Effectively showcases practical application, though some areas may lack clarity or could be more fluid.	Demonstrates practical application but with occasional inconsistencies. Some areas may require improvement in execution.	Struggles to effectively demonstrate the practical application of the modified code. Actions may not align well with intended outcomes.	
Component Diversity and Creativity (10%)	Includes at least 5 diverse components in the application, showcasing creativity and a wide skill set.	Includes 5 components as required, with a good variety. May lack creativity in the selection or execution of some components.	Includes at least 5 components, but the variety or execution may be limited.	Includes fewer than 5 components or lacks variety, impacting the overall scope of the modification.	
Insightful Tactile Aspects Discussion (10%)	Provides a thorough discussion of the tactile aspects of modifications, detailing precision and coordination involved.	Offers clear insights into the tactile aspects but may lack depth in some areas.	Provides basic insights into the tactile aspects, with limited depth or detail.	Offers minimal information on the tactile aspects, lacking clarity and depth.	
Creativity, Code Clarity, and Documentation (10%)	High creativity, excellent code clarity, and comprehensive documentation explaining the thought process, purpose of attributes/methods, and control structure usage, reflecting strong creativity and effective code explanations.	Some creativity and code clarity. Documentation is present but incomplete, showing moderate creativity and an attempt at code explanation.	Limited creativity and code clarity. Minimal or unclear documentation is present, demonstrating minimal effort in adding creativity and explaining the code.	Lack of creativity and code clarity. No documentation is provided, indicating poor creativity and an absence of code explanations.	

Individual Member Evaluation (30%) 1) Task Completion 2) Quality of Work 3) Able to explain clearly 4) Problem-Solving &Technical skills 5) Explain things using OOP terms 6) Q&A	(S1)	(S1)	(S3)
TOTAL MARKS			

REMARKS