**ASSIGNMENT 1 FRONT SHEET**

|  |  |  |  |
| --- | --- | --- | --- |
| **Qualification** | **BTEC Level 5 HND Diploma in Computing** | | |
| **Unit number and title** |  | | |
| **Submission date** | 26/08/2023 | **Date Received 1st submission** |  |
| **Re-submission Date** |  | **Date Received 2nd submission** |  |
| **Student Name** | Dinh Dinh Khoi | **Student ID** | GCC210345 |
| **Class** | GCC1003 | **Assessor name** | Nguyen Kim Khanh |
| **Student declaration**  I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice. | | | |
|  |  | **Student’s signature** |  |

**Grading grid**

|  |
| --- |
| Grade (0-10) |
|  |

|  |  |  |
| --- | --- | --- |
| **❒ Summative Feedback: ❒ Resubmission Feedback:** | | |
| **Grade:** | **Assessor Signature:** | **Date:** |
| **IV Signature:** | | |

Assessment Brief

|  |  |
| --- | --- |
| Student Name/ID Number |  |
| **Unit Number and Title** | **Object Oriented Programming with Java** |
| Academic Year | 2020 - 2021 |
| Unit Tutor |  |
| **Assignment Number & Title** | **Design, Implement and Test a GUI application** |
| **Issue Date** |  |
| Submission Date |  |
| IV Name & Date |  |

|  |
| --- |
| **Submission Format** |
| The submission is in the form of a written report. This should be written in a concise, formal business style using single spacing and font size 12. You are required to make use of headings, paragraphs and subsections as appropriate, and all work must be supported with research and referenced using the Harvard referencing system. Please also provide a bibliography using the Harvard referencing system. |

|  |
| --- |
| **Unit Learning Outcomes** |
| **LO1** Understand basic programming skills and OOP paradigm  **LO2** Understand how to detect errors and handle errors  **LO3** Understand how to working with files in applications  **LO4** Understand how to build GUI application |
| **Assignment Brief** |

|  |
| --- |
| You have to develop an application to solve a small business problem. The problem requires a graphical user interface with features that required reading / writing data from text file, working with a collection of data (searching for item / min / max / sum / etc.). The application must handle errors so that it will not crash at end user side. The application also need to be fully tested before the production phase.  You need to write a technical report about the development of the application. Content of the report should cover design, implementation and testing.  In the end you need to demo your application, explain your code and answer technical questions. |

|  |
| --- |
| Learning Outcomes and Assessment Criteria |
| **LO1** Understand basic programming skills and OOP paradigm  **LO2** Understand how to detect errors and handle errors  **LO3** Understand how to working with files in applications  **LO4** Understand how to build GUI application |
| **To get Pass (5 – 6.5 points)**  - Student can design and implement GUI for the application solve a specific problem  - Student knows how to load and save data from file.  - Student knows how to handle errors by using exceptions  - Student knows how to write test plan, execute test cases and log results. |
| **To get Merit** (**7 – 8.5 points**)  - The application is well designed, user friendly and has logical flow of actions.  - Can apply MVC in the application, can apply JUnit to test automatic  - Errors are well handle to avoid program crashing, the test can cover as many as possible the errors in program |
| **To get Distinction (9 – 10 points)**  The application must show excellent design & implementation, runs without any errors, all inputs are validated, all errors are well handled including recover choice, rich features showing unique ideas, algorithms. |

Table of Contents

[Assessment Brief 3](#_Toc143865060)

[I. Introduction. 7](#_Toc143865061)

[II. Requirement. 7](#_Toc143865062)

[III. Design. 8](#_Toc143865063)

[1. UI design. 8](#_Toc143865064)

[2. Use case. 10](#_Toc143865065)

[IV. Implementation. 11](#_Toc143865066)

[1. Project Structure. 11](#_Toc143865067)

[2. Explain class. 11](#_Toc143865068)

[a. Phone class. 11](#_Toc143865069)

[b. Account class. 13](#_Toc143865070)

[3. Explain code: 14](#_Toc143865071)

[4. Explain how to handle errors 21](#_Toc143865072)

[V. Test. 26](#_Toc143865073)

[VI. Result. 29](#_Toc143865074)

[VII. Conclusion. 54](#_Toc143865075)

[VIII. References. 56](#_Toc143865076)

# Introduction.

KD Store, a retailer of various smartphone models, is planning to build a branch in Can Tho. They wanted us to develop an application so that administrators may manage branches more easily since they were having trouble creating a product and personnel management system. Staff can only handle products using an account that has been given to them by administrator, while administrators may manage both employee accounts and goods.

# Requirement.

After agreeing on the requirements with the customer, I summarize the specific requirements as follows:

* The administrator who wants to enter the management page must enter the correct "Username" and "Password" of the administrator into the login interface.
  + Username
  + Password
* Administrators manage employee accounts through the "Staff Account" interface, where administrators can Create accounts, Update accounts, and Delete staff's accounts.
  + Username
  + Password
  + Staff name
  + Date birth
  + Gender
  + Address
  + Phone number
* Administrators and staff manage products through the “Phone Management” interface, where administrators and employees can Add new products, Update product information, and Delete products.
  + Phone ID
  + Phone name
  + Describe
  + Quantity
  + Price
* The program has the following functions:
  + Create/Add: used to create/add new employee and product accounts.
  + Update: used to update product information of staff accounts, products.
  + Delete: used to delete products and unused accounts.

# Design.

## UI design.

* ***This is the Login interface.***

A screen shot of a login screen

Description automatically generated

*Login interface.*

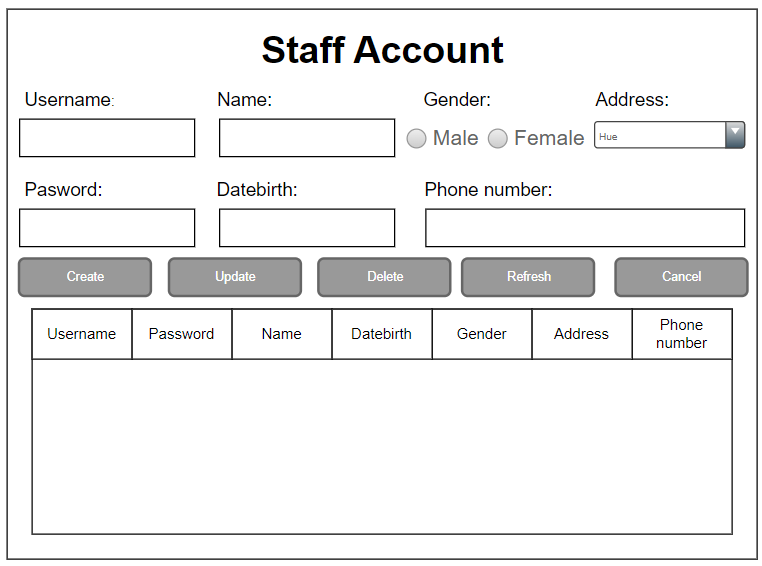
* ***This is Admin interface.***

A screenshot of a computer screen

Description automatically generated

*Admin page (only for admin).*

* ***This is Staff Account interface.***



*Staff Account page (Admin is used to manage employee accounts).*

* ***This is Phone Management interface.***

A screen shot of a phone management form

Description automatically generated

*Phone Management page used for product management.*

## Use case.

A diagram of a company

Description automatically generated

*Use case diagram of project.*

# Implementation.

## Project Structure.

My project consists of 4 main Source Packages: Data, Model, View and asm\_java. Data is a package used to store account information and product information. Model is used to hold the Phone and Account objects. View contains the Login interface, Admin interface, Phone Management interface and Staff Account interface. Finally, asm\_java contains the main functionality of the project.

A screenshot of a computer

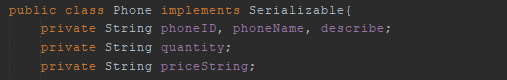
Description automatically generated

## Explain class.

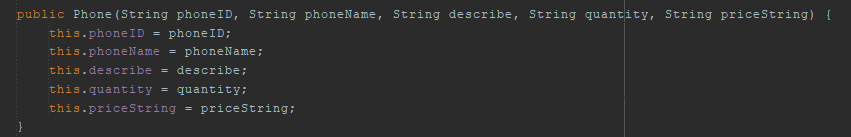
### Phone class.

Phone class are used to get product information. In this class, I import java.io.Serializable so that the class can receive and run the program's interface. The implement keyword is an important keyword for the interface to be able to receive objects. Next, create the property variables for the class.

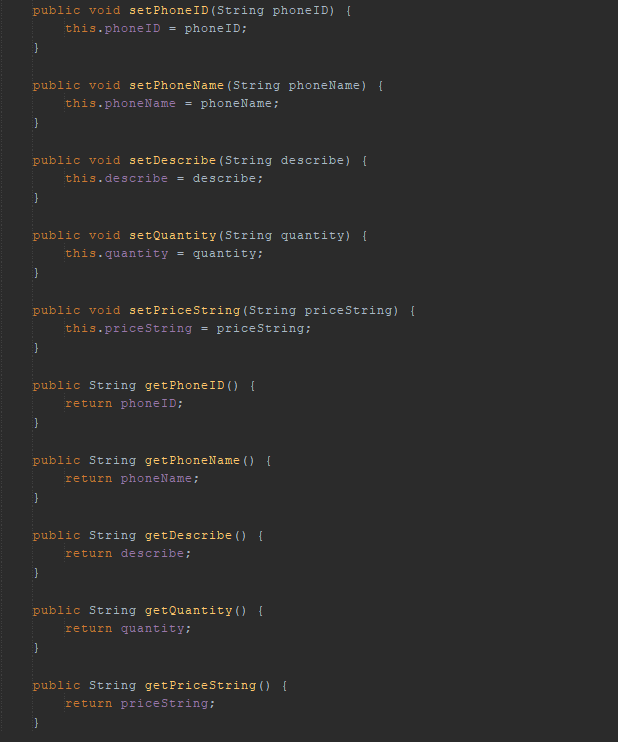




The constructor creates a data object on the property variables declared above to receive data about the product's information.



The set functions are responsible for taking product information entered by the user and assigning them to the properties of the object. At the same time, the get function is used to get data from the object when the user needs it.



### Account class.

Account class are used to get staff account information. I import java.io.Serializable so that the class can receive and run the program's interface. The implement keyword is an important keyword for the interface to be able to receive objects. Next, create the property variables for the class.



A screen shot of a computer

Description automatically generated

The constructor creates a data object on the property variables declared above to receive data about the staff account's information.

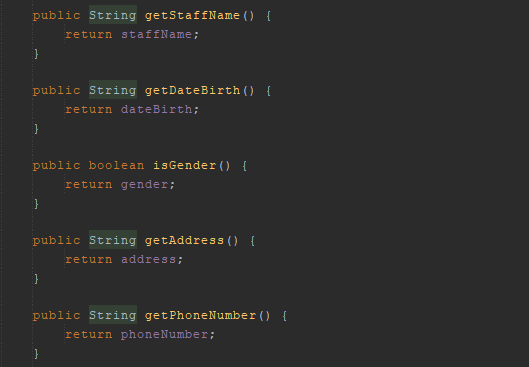
A computer screen shot of a code

Description automatically generated

The set functions are responsible for taking staff account information entered by the user and assigning them to the properties of the object. At the same time, the get function is used to get data from the object when the user needs it.

A screenshot of a computer program

Description automatically generated



## Explain code:

* **Login button:**

Firstly, declare two variables user and pass to receive 2 values username and password.

A black screen with text

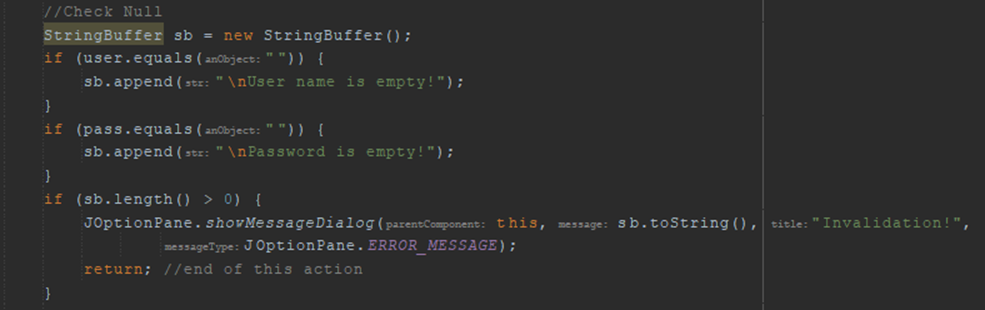
Description automatically generated

The "for-each" statement has a function to create two ArrayList containing the staff's username and password:

A screen shot of a computer

Description automatically generated

StringBuffer and "if" statements are used to check if username and password are empty or not:

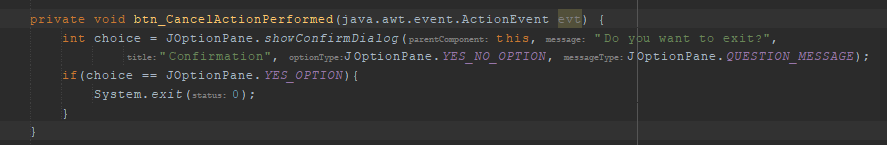


The login button implements a function to check if the login account is indeed a valid account by using an “if-else if” statement to check the conditions.

A screen shot of a computer program

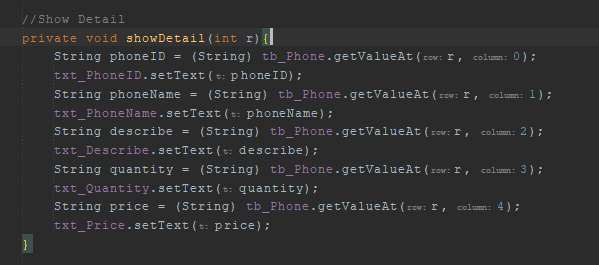
Description automatically generated

* ***Cancel button:*** The function of the Cancel button is to close the program when the user wants to exit the application.



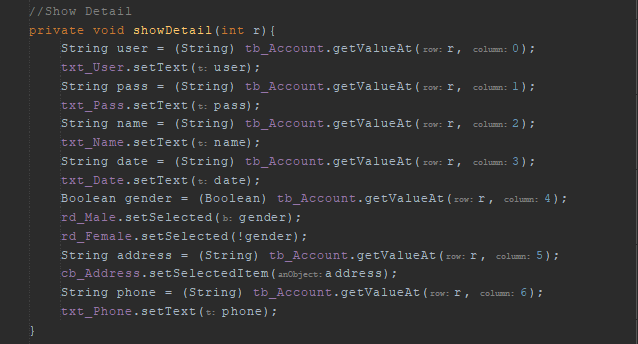
First, declare a variable choice to receive the application exit confirmation message. If choice = YES\_OPTION it will close the application immediately.

* showDetail: perform the function of displaying the data of the selected object in the table.



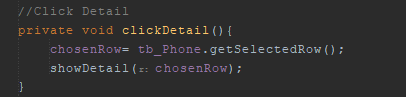
*Show detail function of Phone Management interface.*

Declare the variables that receive data values in the table in turn. (Show detail of Staff Account interface is the same too).



*Show detail function of Staff Account interface.*

* clickDetail: has the function of displaying selected line information on the screen.



*Click detail of Phone Management.*

First assign the data of the selected row to the chosenRow variable. Then call the showDetail function to display the data of the selected chosenRow on the screen. (Click detail of Staff Account interface is the same too).

A screen shot of a computer code

Description automatically generated

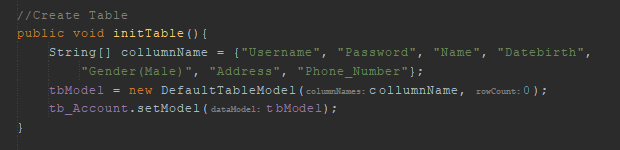
*Click detail of Staff Account interface.*

* initTable: create table. Initialize the String[] array to get the data fields of the object. Next, initialize the object table and then assign the data fields to the table.

A screen shot of a computer code

Description automatically generated

*initTable of Phone Management.*



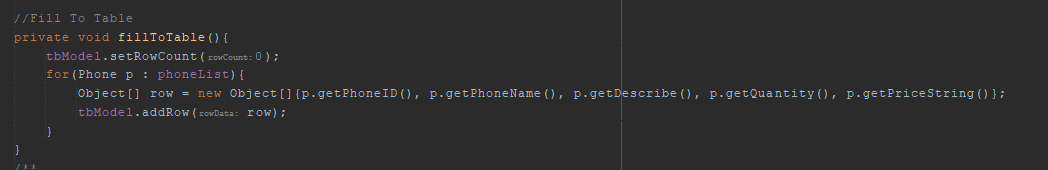
*initTable of Staff Account.*

* initAddress: create combo box. Initialize the combo box object, then create a String[] array to get the staff address. Next, use a for-each loop to assign the values of the String[] array to the combo box.

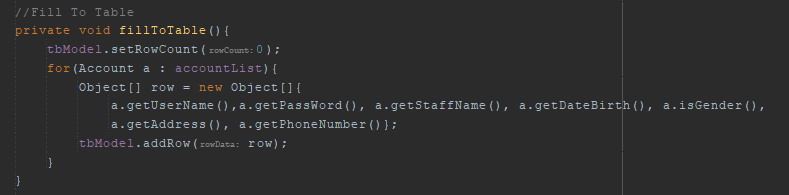
A computer screen shot of text

Description automatically generated

* fillToTable: add Phone/Staff account data to the table using for-each loop. The for-each loop is used to assign product/employee information to an object. Then add the object's data to the table.



*Fill to table of Phone Management.*



*Fill to table of Staff Account.*

* addPhone/ addAccount: There is a function to add new/ create products/ staff accounts. Use if-else if statement to check input conditions, format, etc. If none of the conditions are satisfied, create a new object to save the data to the file and display it in the table.

A screen shot of a computer code

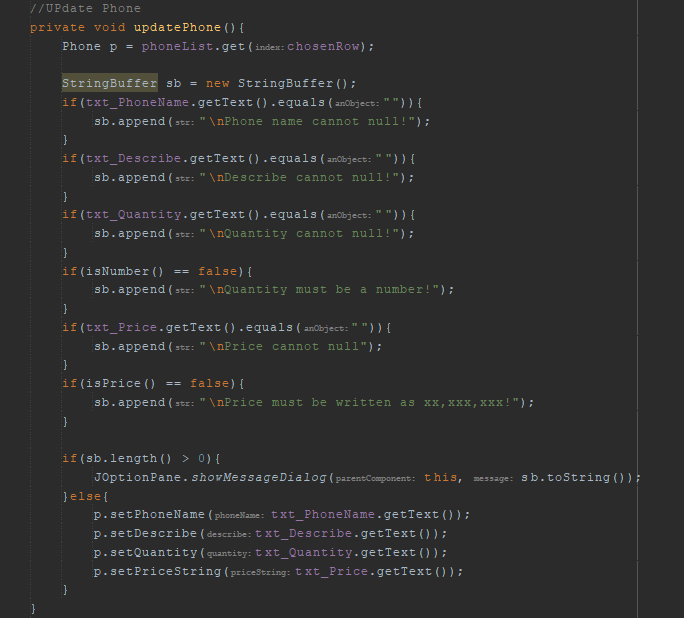
Description automatically generated

*Add Phone.*



*Add Account.*

* updatePhone/ updateAcc: Update information. First create a phoneList that receives data from the selected row in the table. Then use StringBuffer to check for errors when updating information. If there are no errors, then the data will be updated to the table.

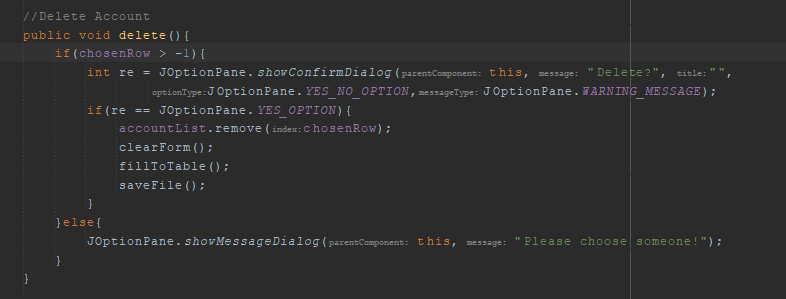


*Update Phone.*



*Update Account.*

* delete: Delete data in the table. If choosenRow > -1, declare the variable re to receive a message as YES\_NO\_OPTION asking if the user wants to remove it. If re = YES\_OPTION then delete and save the file. In case the chosenRow does not meet the condition, the message "Please select someone!" is displayed.



*Delete phone/ account.*

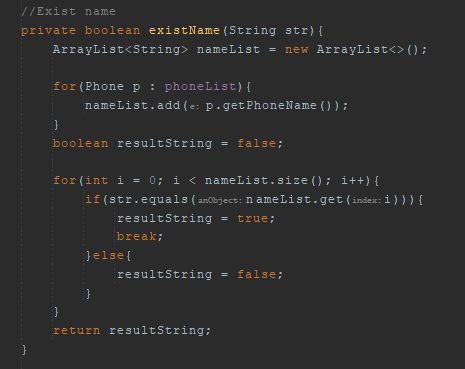
## Explain how to handle errors

* existID/ existName/ existUsername/ existPhone: Check if phone id, phone name, username, phone number are already exist using regular expression. Declare an ArrayList of object ids. Then use for-each to assign an id to the newly created list. Next, create a variable resultString with a boolean data type that takes a false. Use a for loop to check the ids in the list, if the id entered is the same as the existing id, return true and close the program. Otherwise, return false and return resultString. The remaining cases are similar.

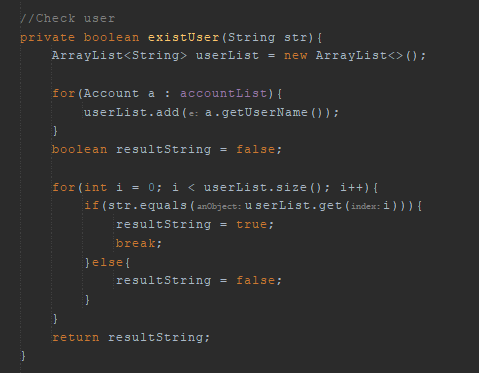
A computer screen shot of a program code

Description automatically generated

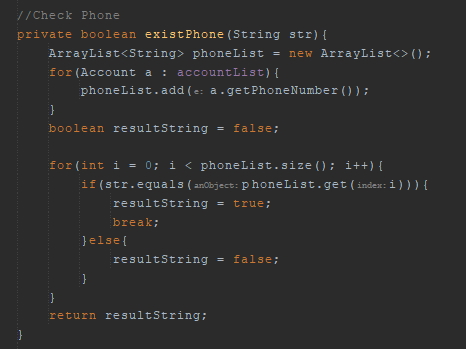
*existID.*



*existName.*



*existUser.*



*existPhoneNumber.*

* isMixedID: Initialize the id variable with the format id "^[GC]{2}\\d{2}$", making the variable re of the boolean data type. Create a variable str that takes input from the keyboard. Then match str with id and return the value re.

A screen shot of a computer code

Description automatically generated

* isPrice: Initialize the price variable with the format price "^[0-9]{2},[0-9]{3},[0-9]{3}$", making the variable re of the boolean data type. Create a variable str that takes input from the keyboard. Then match str with price and return the value re.

A screen shot of a computer program

Description automatically generated

* isNumber: Initialize the number variable with the format number "[0-9]{2}$", making the variable re of the boolean data type. Create a variable str that takes input from the keyboard. Then match str with number and return the value re.

A computer screen shot of a number

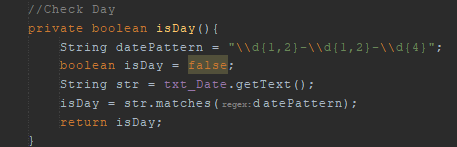
Description automatically generated

* isPhoneNumber: Initialize the phone variable with the format phone "^[0][0-9]{2}[0-9]{3}[0-9]{4}$", making the variable re of the boolean data type. Create a variable P that takes input from the keyboard. Then match P with phone and return the value re.

A screen shot of a computer code

Description automatically generated

* isDay: Initialize the dayPattern variable enter the format day "\\d{1,2}-\\d{1,2}-\\d{4}", create the isDay variable to receive the value false. Create a variable str to receive data from the keyboard, then match str with dayPattern and return isDay.

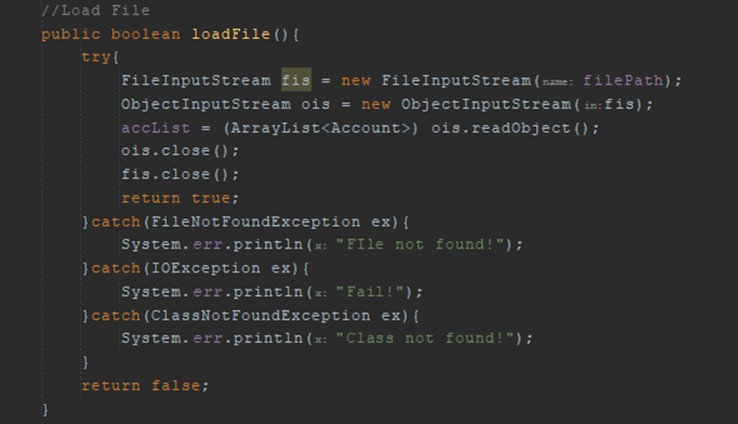


* isString: Initialize the name variable with the format name "[a-zA-Z ]+", making the variable re of the boolean data type. Create a variable str that takes input from the keyboard. Then match str with name and return the value re.

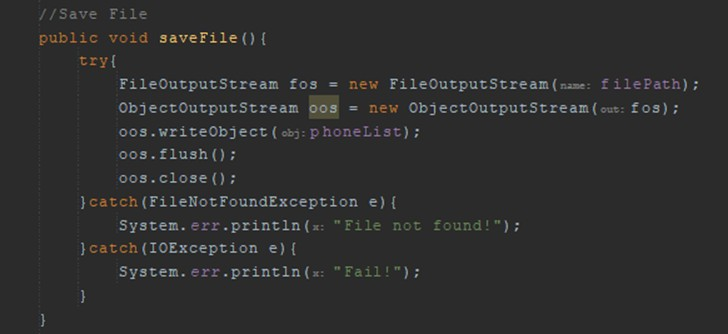
A screen shot of a computer code

Description automatically generated

* loadFile: There is a function to read input data streams from filePath, check for exceptions that occur during system running.
  + FileNotFoundException will be thrown by FileInputStream constructor when file with the specified pathname does not exist.
  + IOException occurs when an input or output operation fails.
  + ClassNotFoundException occurs when the Java Virtual Machine (JVM) cannot find the class specified in the code.



* + ClassNotFoundException occurs when the Java Virtual Machine (JVM) cannot find the class specified in the code.
* saveFile: This function performs 3 functions saving data to the specified file path, displaying data in a table and catching exceptions.



* + First, in the try statement initialize a FileOutputStream (fos) object that receives data from the "filePath" path and an ObjectOutputStream (oos) object that receives a fos object. Then write the data to the ArrayList and display the data to the table. Finally, close the oos object. If the process occurs one of the exceptions, the catch statement will be executed.

# Test.

* Test plan:
  + Creator: Dinh Dinh Khoi.
  + Test date: 21/08/2023.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ***No.*** | ***Test case*** | ***Function*** | ***Test data*** | ***Expected output*** | ***Actual output*** | ***Result*** |
| 1 | Verify that the screen shows the message "Login successful!" and "Welcome Administrator!" when admin enter the correct Username and Password of the account for admin. | Login | -Username: Admin  -Password: Admin123@ | When the login is successful, the message "Login successful!" and "Welcome Admin!", then will switch to the "Admin page" interface. | When the login is successful, the message "Login successful!" and "Welcome Admin!", then will switch to the "Admin page" interface. | Pass |
| 2 | Verify that the screen shows the message " Login success" when staff enter the correct Username and Password of the account for staff. | Login | -Username: dinhkhoi15  -Password: khoidinh123 | When the login is successful, the message "Login successful!" and switch to the "Phone Management" interface. | When the login is successful, the message "Login successful!" and switch to the "Phone Management" interface. | Pass |
| 3 | Verify that, phone information will be saved to the table if admin or staff enter all correct information. | Add | -Phone ID: GC01  -Phone name: Samsung s23  -Describe: 8gb RAM  -Quantity: 50  -Price: 20,000,000 | The phone information will be saved to the table. | The phone information will be saved to the table. | Pass |
| 4 | Verify that the screen shows the error "Phone ID must be written as GCxx ( x is a number)!" if the administrator or employee enters the wrong phone ID format. | Add | -Phone ID: 001  -Phone name: Iphone 14  -Describe: 128gb  -Quantity: 50  -Price: 20,000,000 | The screen will display the error "Phone ID must be written as GCxx ( x is a number)!". | The screen will display the error "Phone ID must be written as GCxx ( x is a number)!". | Pass |
| 5 | Verify that the screen shows the error "Phone name already exists!" if the administrator or employee enters the phone name that appeared before. | Add | -Phone ID: GC04  -Phone name: Samsung s23  -Describe: 64gb  -Quantity: 50  -Price: 15,000,000 | The screen will display the error "The phone name already exist!". | The screen will display the error "The phone name already exist!". | Pass |
| 6 | Verify that the screen shows the error "Price must be write as xx,xxx,xxx!" if admin or staff enter wrong price format. | Add | -Phone ID: GC05  -Phone name: Samsung X  -Describe: 32gb  -Price: 2000sdad | The screen will display the error "Price must be write as xx,xxx,xxx!". | The screen will display the error "Price must be write as xx,xxx,xxx!". | Pass |
| 7 | Verify that the phone's information is deleted from the board when an administrator or employee presses the "Delete" button. | Delete | Click on the phone information line containing the ID GC03 | The data of the phone ID GC03 will be deleted from the table. | The data of the phone ID GC03 will be deleted from the table. | Pass |
| 8 | Verify that the data of the selected line will be updated after the administrator or employee presses the "Update" button. | Update | -Phone ID: GC02  -Phone name: Samsung SX  -Describe: 8gb RAM  -Quantity: 50  -Price: 20,000,000 | The phone name of ID GC02 will change to “Samsung SX”. | The phone name of ID GC02 will change to “Samsung SX”. | Pass |
| 9 | Verify that the screen shows the error "Quantity must be a number!" if admin or staff enter wrong quantity type. | Update | -Phone ID: GC02  -Phone name: Samsung SX  -Describe: 8gb RAM  -Quantity: asd  -Price: 20,000,000 | The screen will display the error " Quantity must be a number!". | The screen will display the error " Quantity must be a number!". | Pass |
| 10 | Verify that the screen shows the error " Price must be written as xx,xxx,xxx!" if admin or staff enter wrong price format. | Update | -Phone name: Samsung SX  -Describe: 8gb RAM  -Quantity: asd  -Price: asdada | The screen will display the error " Price must be written as xx,xxx,xxx!". | The screen will display the error " Price must be written as xx,xxx,xxx!". | Pass |

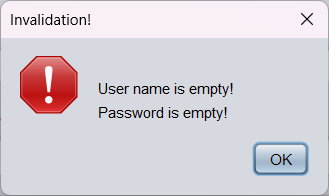
# Result.

* Login interface:
* When the Administrator or Staff runs the program, the first interface that appears is the login interface that requires them to enter "Username" and "Password" to log in to the application.

A screenshot of a login screen

Description automatically generated

* If the Administrator or Staff do not enter “Username” and “Password” but presses the Login button, the program will show error message:

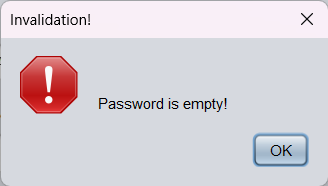


* If the Administrator or Staff do not enter “Username”, the program will show error message:

A screenshot of a computer screen

Description automatically generated

* If the Administrator or Staff do not enter “Password”, the program will show error message:

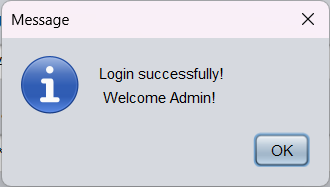
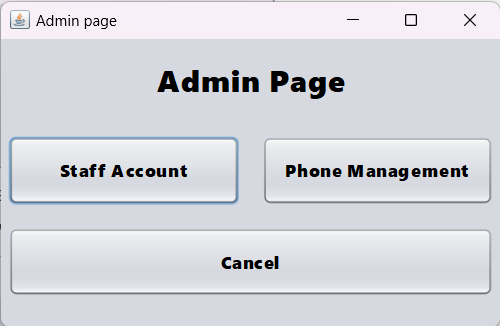


* If the Administrator enter wrong “Username” and “Password” or both, the program will show error message:

A screenshot of a computer screen

Description automatically generated

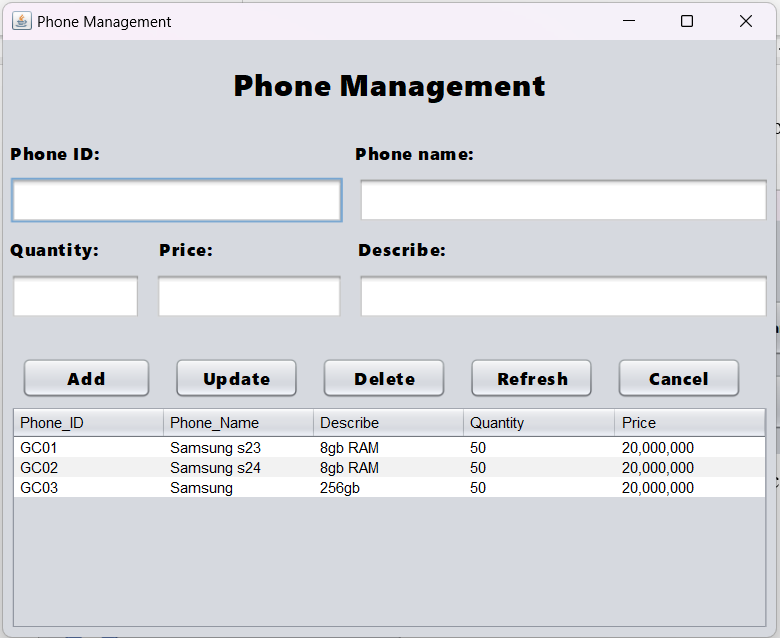
* In case the administrator enters the correct administrator account, the program will notify "Login successful!" and “Welcome Admin!”, then switch to the Admin page interface:

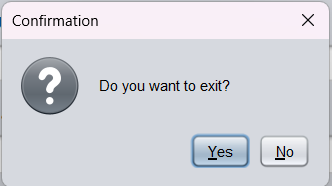
* In case the staff enters the correct staff account, the program will notify "Login success", then switch to the Phone Management interface:

A screenshot of a login screen

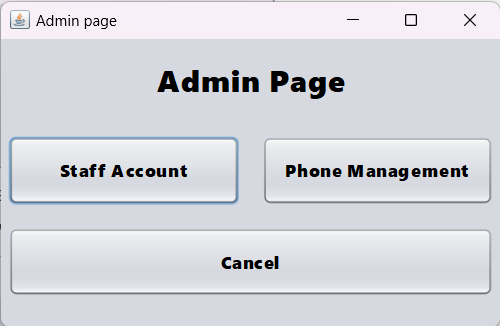
Description automatically generated



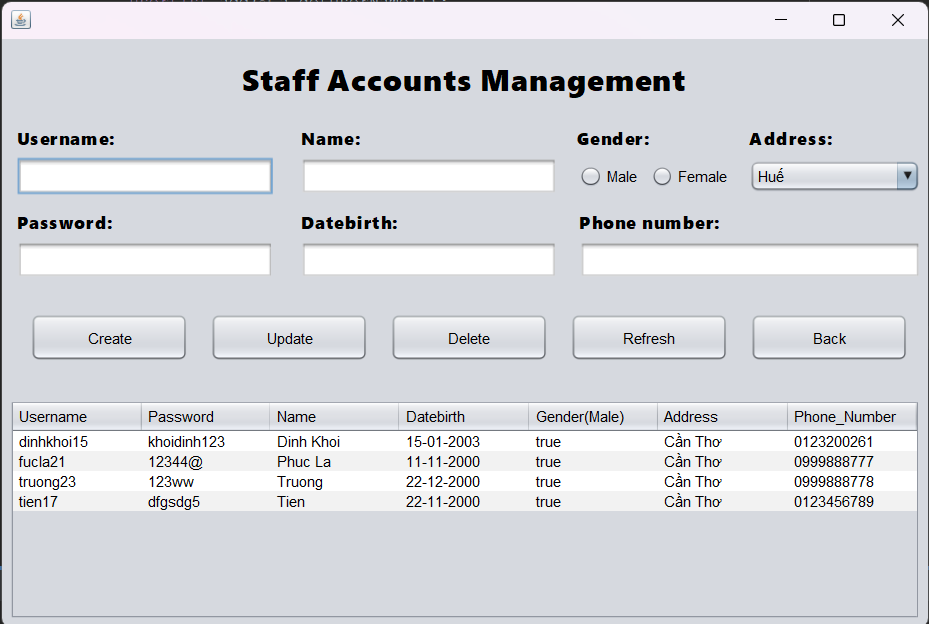
* If the Administrator or Staff want to exit the program, press the “Cancel” button:



* Admin interface:
* At the "Admin page" interface, the admin can choose the management functions "Staff Account" or product management "Phone Management".



* When the Administrator choose Staff Account button, the program will switch to Staff Account interface:



* When the Administrator choose Phone Management button, the program will switch to Phone Management interface:

A screenshot of a phone management

Description automatically generated

* When the Administrator or Staff want to exit to login interface, press “Cancel” button:

A screenshot of a computer

Description automatically generated

* Phone Management interface:
* This is the product management interface called "Phone Management". Here, Administrator and Employees can perform the functions of Add new products, Update product information and Delete products.

A screenshot of a phone management

Description automatically generated

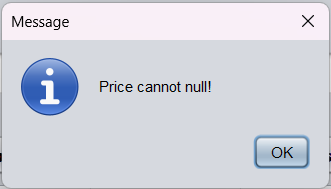
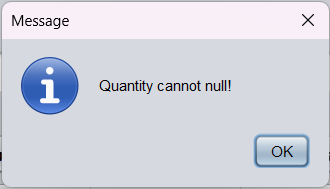
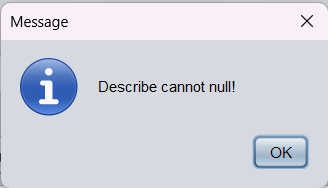
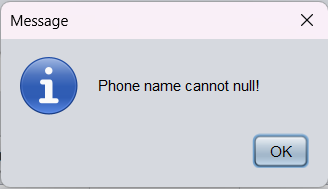
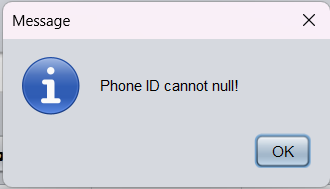
* ***Add button:***
* When the Administrator or Staff enter the correct product information, press the “Add” button, the product data will be saved in the information table below:

A screenshot of a phone management

Description automatically generated



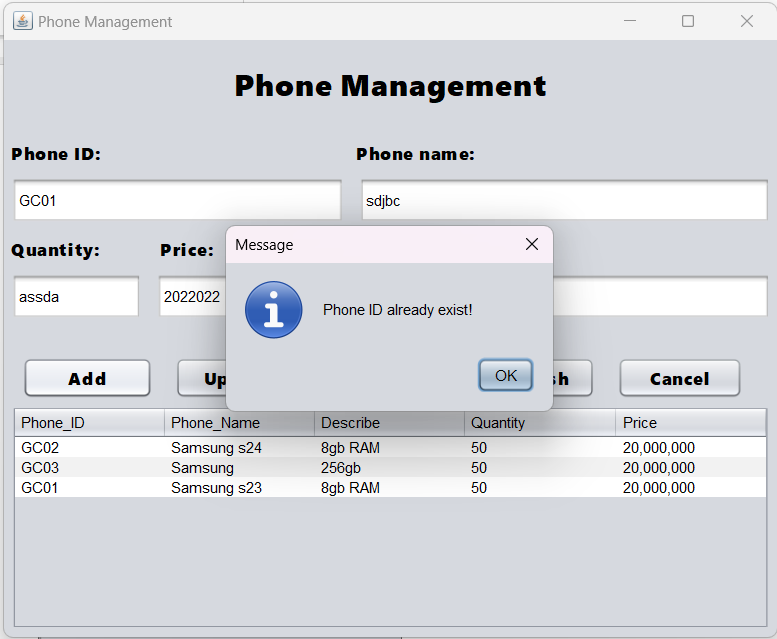
* If the Administrator or Staff leaves 1 or more blank cells, the system will display an error message as follows:



* If the Administrator or Staff enter wrong ID format, the program will show error message:



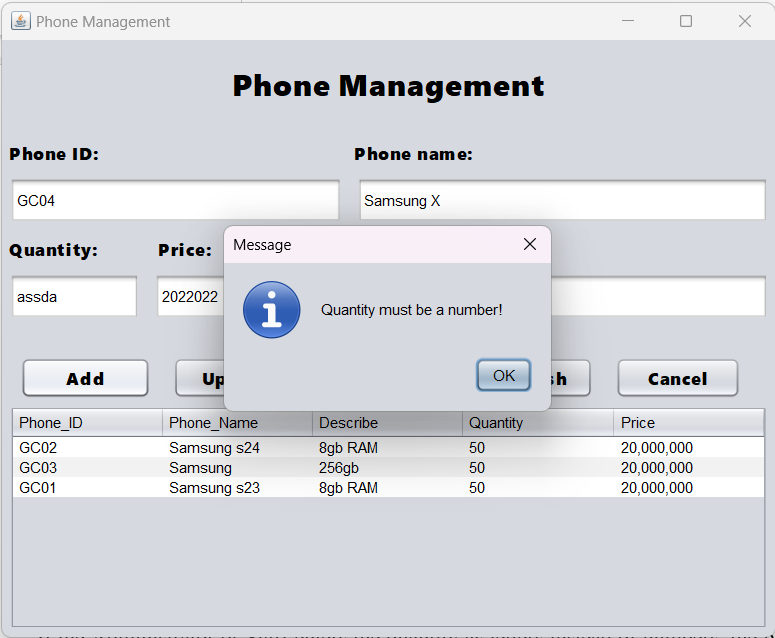
* If the Administrator or Staff enter an existed ID, the program will show error message:



* If the Administrator or Staff enter an existed phone name, the program will show error message:



* If the Administrator or Staff enters the quantity as letters instead of numbers, the program will show error message:



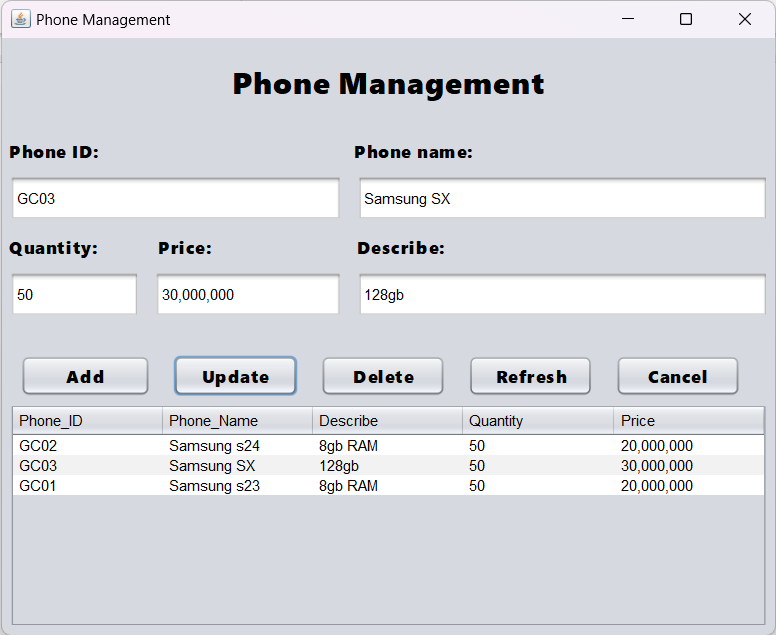
* If the Administrator or Staff enters wrong price format, the program will show error message:



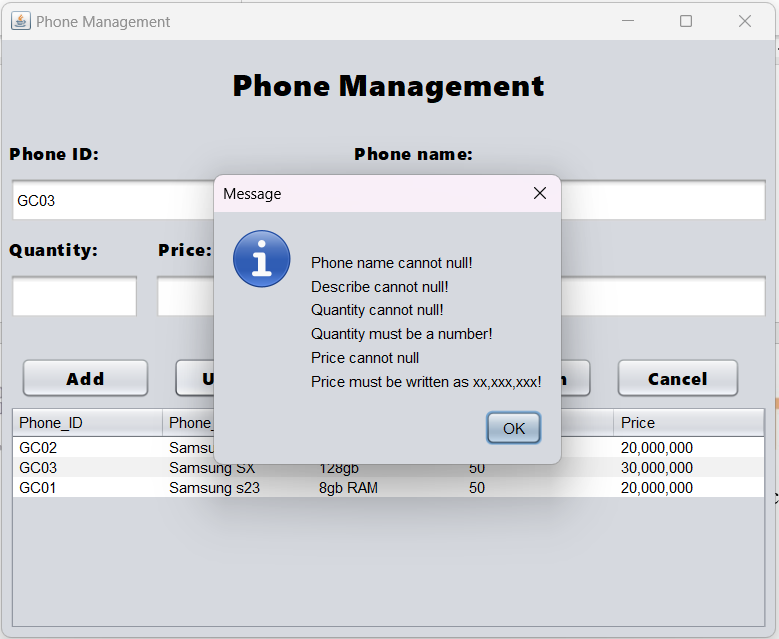
* ***Update button:***
* When the Administrator or Staff enter the correct product information need to update, press “Update” button, the product data will save to the table below:

A screenshot of a phone management

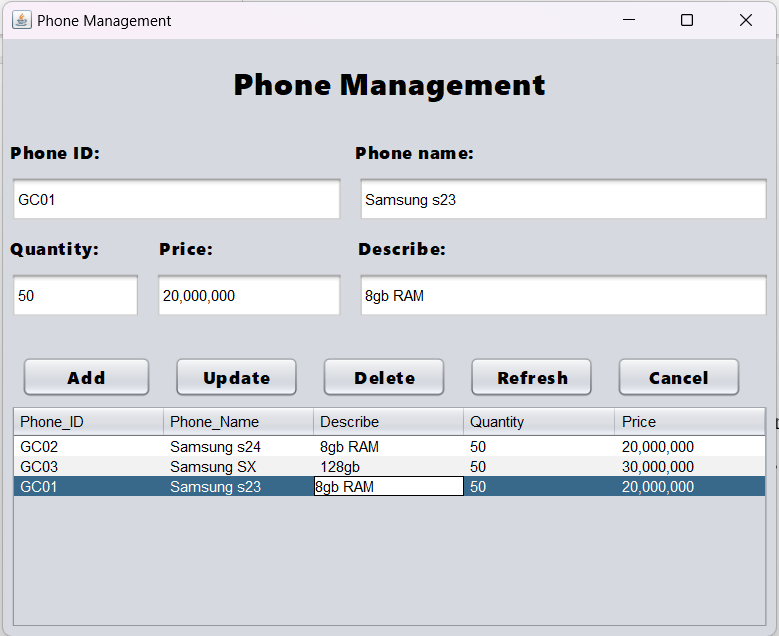
Description automatically generated

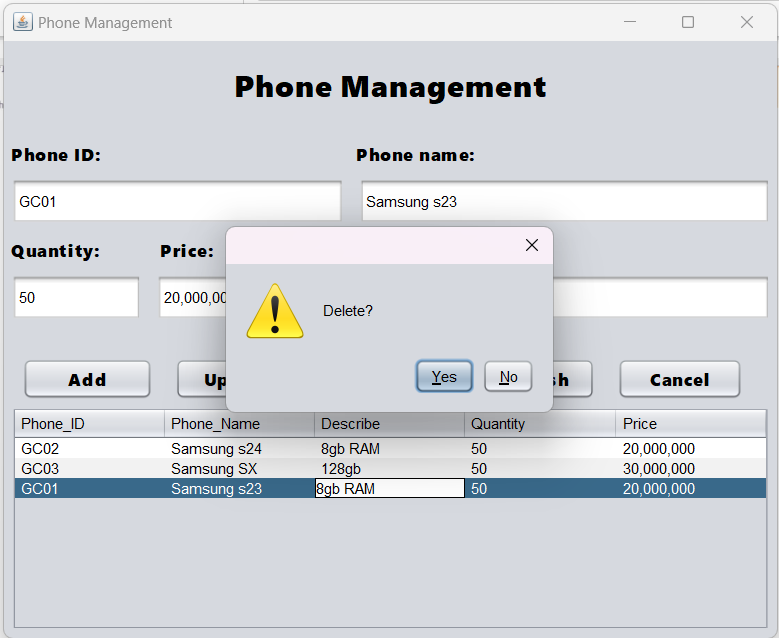


* If Administrator or Staff leave blank phone name, describe, quantity, product price, wrong price format and quantity format, the program will show error message:



* ***Delete button:***
* To delete a product, Administrator or Staff need to select the product line they want to delete, then press the “Delete” button. The program will ask the Admin or Staff if they want to delete, press "Ok" to delete:







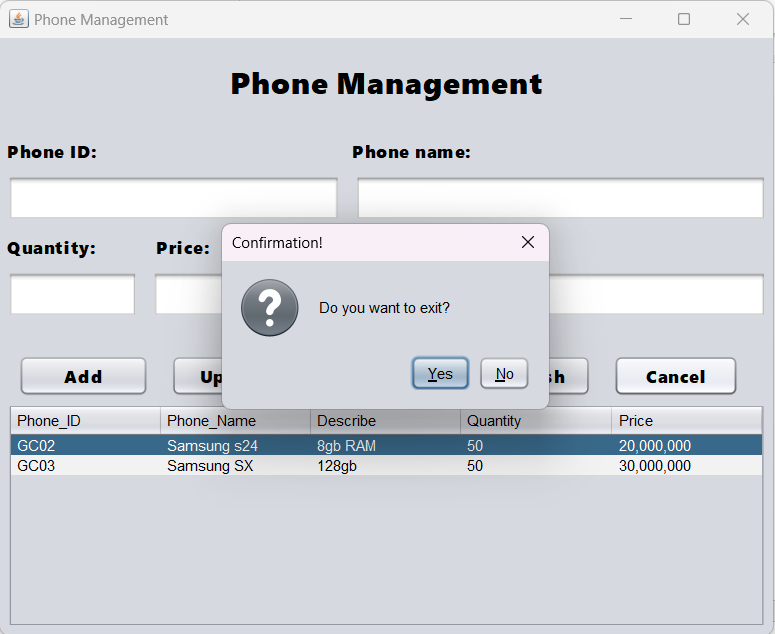
* ***Refresh button:***
* “Refresh” button is used to clear all input data in the text fields:

A screenshot of a phone management

Description automatically generated



* ***Cancel button:***
* If the Administrator or Staff want to exit to login interface, press “Cancel” button:



A screenshot of a login screen

Description automatically generated

* Staff Account interface:
* This is the staff account management interface called "Staff Account". Here, Administrator can perform the functions of Create new account, Update account information and Delete accounts.

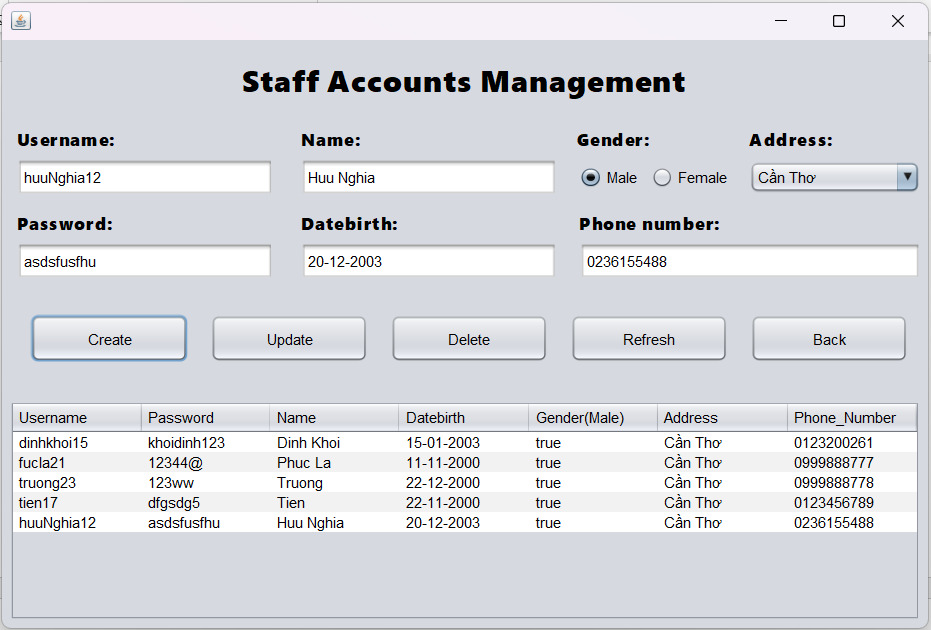
A screenshot of a computer

Description automatically generated

* ***Create button:***
* When the Administrator enters the correct account information, press the “Create” button, the account data will be saved in the information table below:

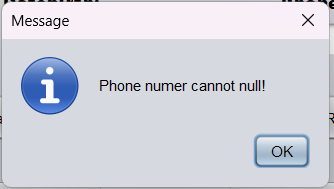
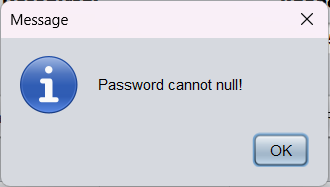
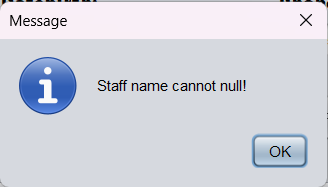
A screenshot of a computer

Description automatically generated

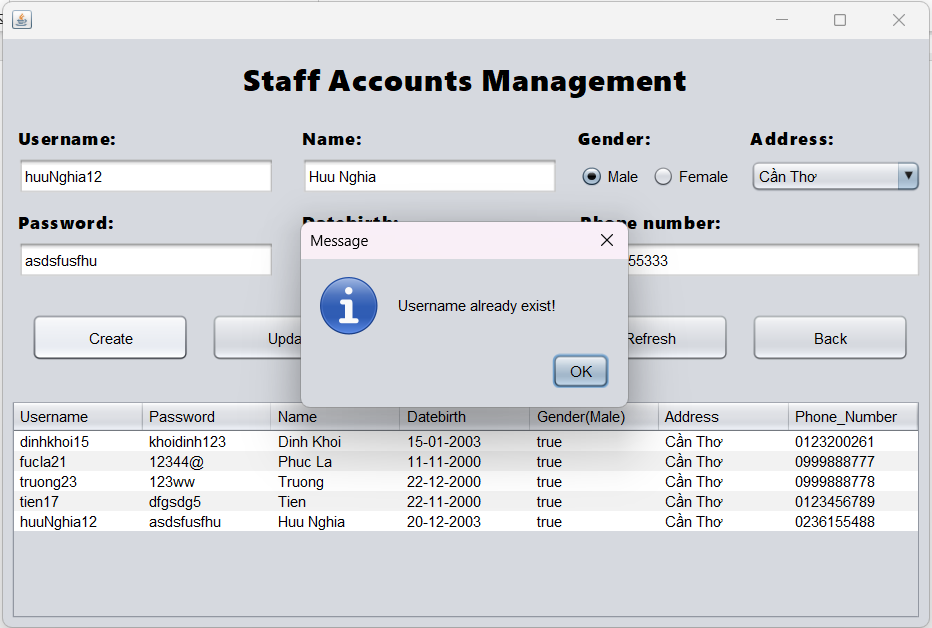


* If the Administrator leaves 1 or more blank cells, the system will display an error message as follows:

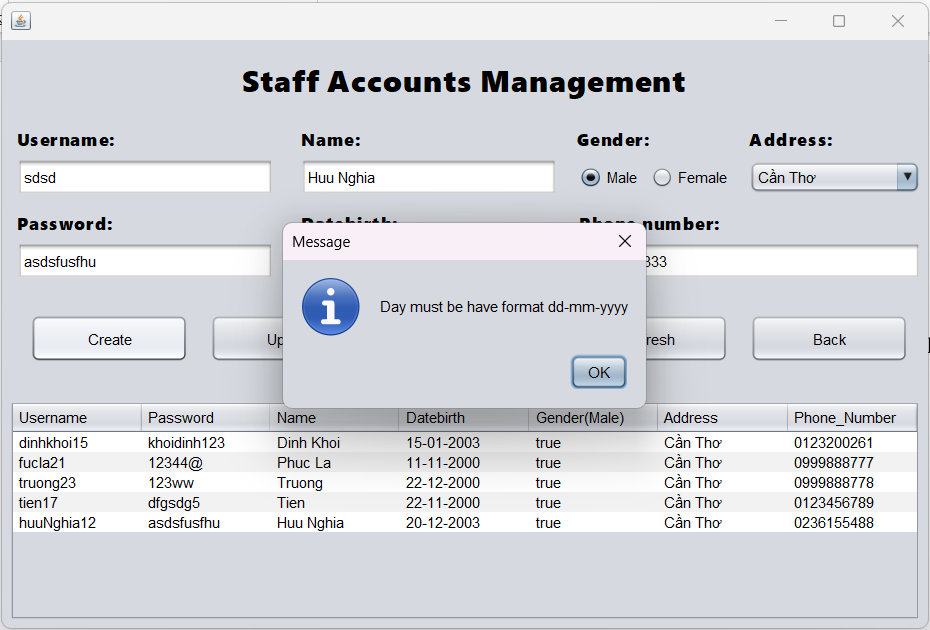
A screenshot of a computer screen

Description automatically generated

* If the Administrator enters an existed Username, the program will show error message:



* If the Administrator leaves the Date birth blank or enters wring format date, the program will show error message:

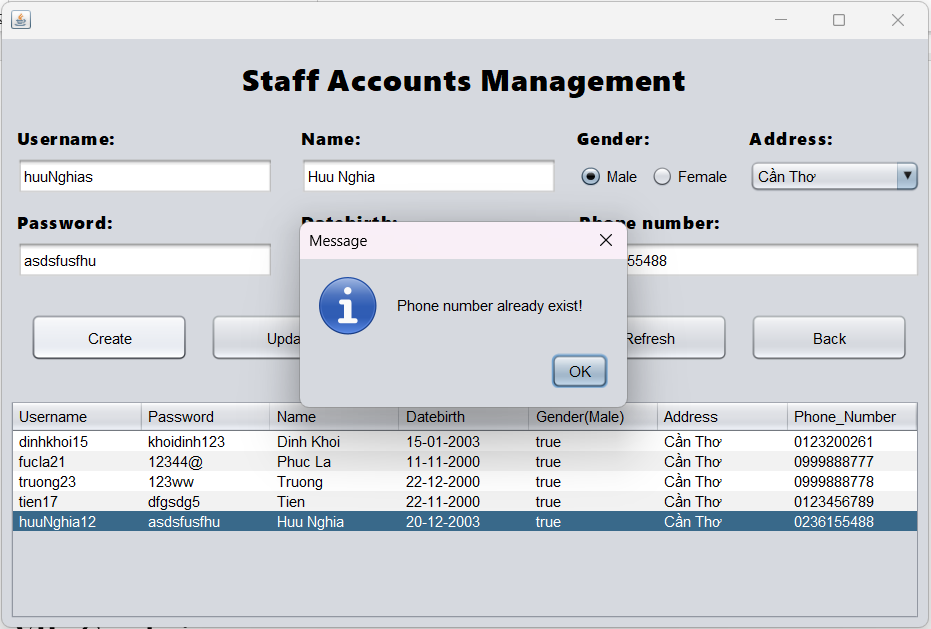


* If the Administrator enters the staff name as numbers instead of letters, the program will show error message:

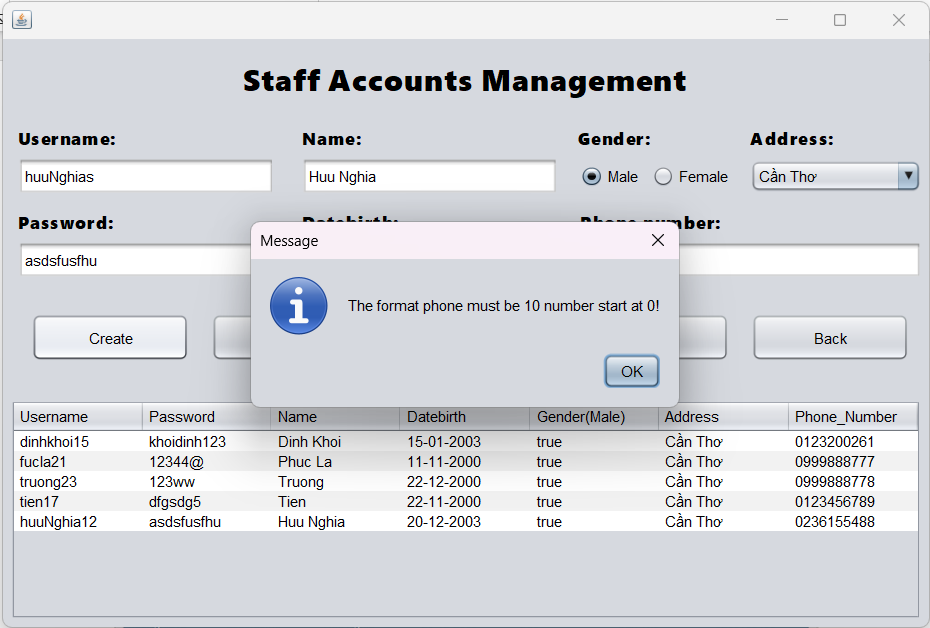
A screenshot of a computer

Description automatically generated

* If the Administrator enters an existed phone number, the program will show error message:



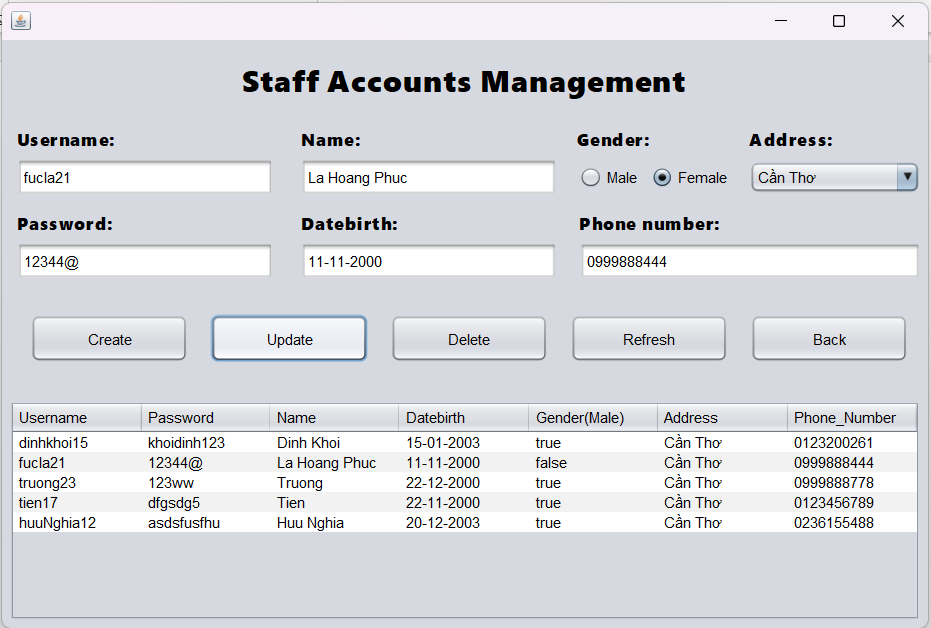
* If the Administrator enters wrong phone number format, the program will show error message:



* ***Update button:***
* When the Administrator enters the correct staff account information need to update, press “Update” button, the account data will save to the table below:

A screenshot of a computer

Description automatically generated



* If Administrator leaves blank username, staff name, password, date birth, phone number, wrong date format, staff name format, and phone number format, the program will show error message:

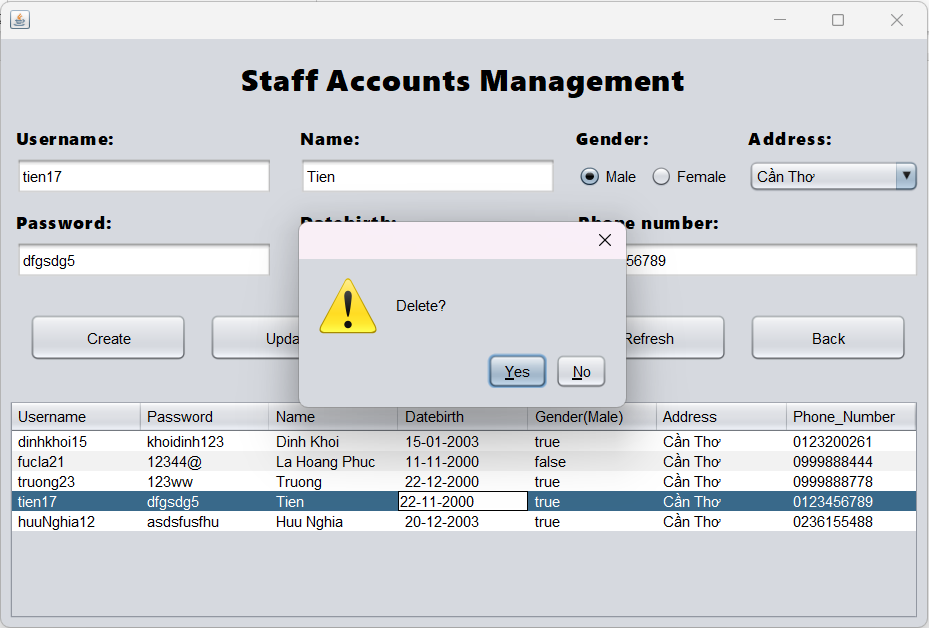
A screenshot of a computer

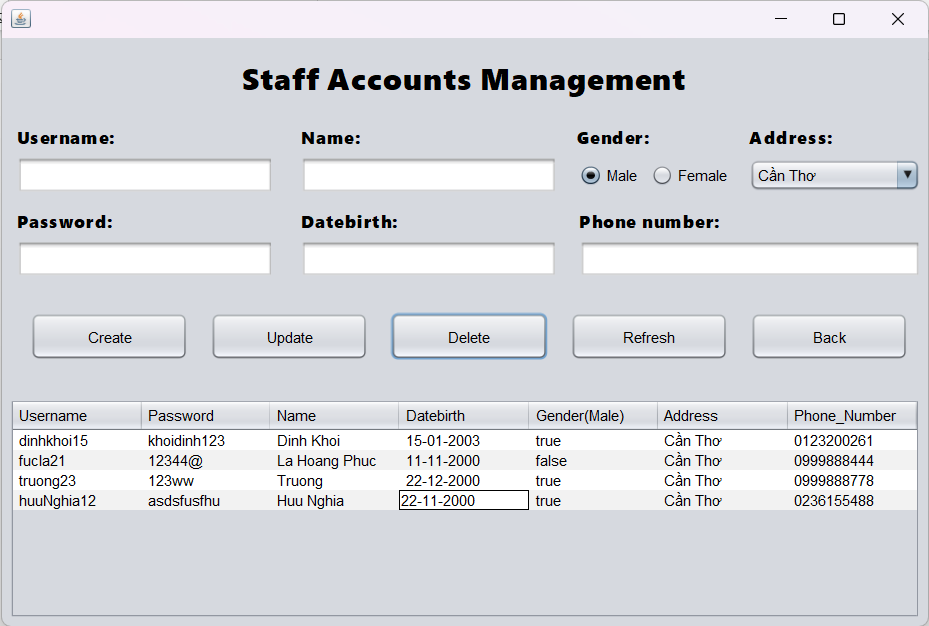
Description automatically generated

* ***Delete button:***
* To delete a account, Administrator needs to select the account they want to delete, then press the “Delete” button. The program will ask the Admin if they want to delete, press "Ok" to delete:

A screenshot of a computer

Description automatically generated

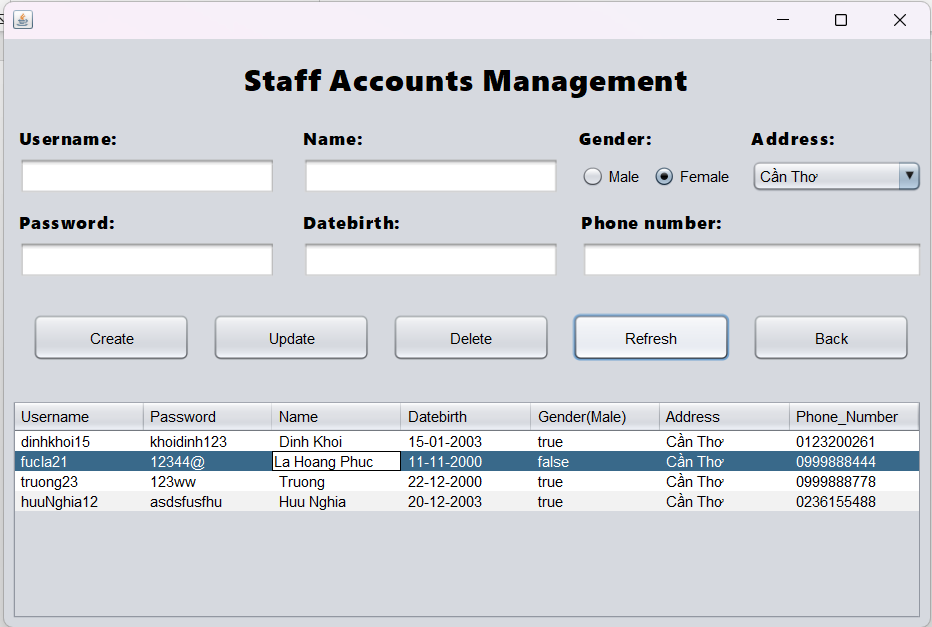




* ***Refresh button:***
* “Refresh” button is used to clear all input data in the text fields:

A screenshot of a computer

Description automatically generated



* ***Back button:***
* If the Administrator want to exit to login interface, press “Back” button:

A screenshot of a computer

Description automatically generated

A screenshot of a login screen

Description automatically generated

# Conclusion.

* Strength: Accomplish most of the main functions of the application: Sign in, Add, Update, Delete, etc. Understand the basics of creating an application using the Java programming language. Several regular expressions can be used to check for formatting errors of some data fields. Understand the basics of the Java language.
* Weakness:
  + When logging in, the password from the file cannot be checked.
  + There is no data connection between the interfaces.
  + The design is sketchy.

# References.

FLM, n.d. [Online]   
Available at: https://flm.greenwich.edu.vn/gui/role/student/SyllabusDetails?sylID=2581