NAMIBIA UNIVERSITY OF SCIENCE AND TECHNOLOGY

Names	Student Numbers
Shane Swartbooi	223098051
Hofeni Haamandike	224034782
Isack Teofilus	224082760
Jayden Nakakuwa	223025135
Fabregas Dimba	223104884

DSA Group 94 Project

Section A

Project overview

Title: Phonebook Applications

INTRODUCTION:

- A simple phone book management application is a practical tool for storing and managing contact information.

- A phonebook app generally allows users to:

  - Add new contacts with details like name and phone number.
  - View a list of all saved contacts, search for specific contacts, edit existing contact information and delete contacts.

Description of functions

1. Phonebook: This is the constructor that initializes the GUI components for the phonebook application.
2. CreateAddPanel: Creates a panel for adding contacts with text fields for entering contact information.
3. CreateViewPanel: Sets up the panel for viewing and searching contacts.
4. CreateUpdatePanel: Creates a panel for updating and deleting contacts.
5. CreateButton: Helper method to create button with specified text, color, and style.
6. ActionPerformed: Handles actions for buttons throughout the application, specifically the "Add Contact" button in this case.
7. IsUniqueContact: Checks if the provided name or phone number is unique within the contacts list.
8. ViewContacts: Sorts the contacts alphabetically and updates the display list.
9. SearchContacts: Searches for contacts that match the text in the searchField and filters the display list accordingly.
10. DeleteContacts: Deletes the selected contact from the contacts list after user confirmation.
11. UpdateContacts: Allows the user to update the name or phone number of the selected contacts.

Main Application operation

Function 1: Insert Contact:

- User enters name and phone number.
- The Phonebook module stores the new contact in the contacts list.
- The GUI shows confirmation that the contact was added.

Function 2: Search Contact:

- The Phonebook module searches for the contact by name.
- User enters a name in the search field.
- If found, the contact's details are displayed in the GUI.

Function 3: View All Contacts:

- User clicks the "View All" button.
- The Phonebook module retrieves all stored contacts.
- The GUI displays the list of contacts.

Function 4: Delete Contact:

- User enters a name in the search field and the Phonebook module deletes the contact with the matching name.
- The Phonebook module deletes the contact with the matching name and if the contact provided has no match found, it will prompt you to enter the correct contact .
- The GUI shows confirmation if the contact was deleted

Function 5: Update Contact:

- User select a contact to update
- The Phonebook module provide the contact and user updates the phone number if the contact is found.
- The GUI shows confirmation if the contact was updated or an error message if the contact was not found.

Modules Used

Our Phonebook used five modules:

- Module 1: GUI Management
- Module 2: Data Management
- Module 3: Event Handling
- Module 4: Contact Model

Description of modules

Module 1: GUI Management:

- Uses Swing for the graphical user interface (JFrame, JButton, JTextField,JTable, etc.).
- Implements a tabbed interface (DefaultTableModel) with three main panels: 1. Add Contact Panel, 2. View All , 3.Search Panel, 4.Update, 5.Delete Panel

The purpose of this module is to create and manage the graphical user interface of the application.

- It sets up the main frame (JFrame) and organizes the layout using a TABLE MODEL (Default).
- Creates separate panels for different functionalities: adding contacts, viewing contacts, searching contacts, and updating/deleting contacts.
- Handles the visual presentation of data through components like JList and JTextField.
- Ensures a user-friendly interface with properly labelled and organized input fields and buttons.

Module 2: Data Management:

- Uses an ArrayList to store Contact objects.
- Implements the operations (Create, Read,View, Update, Delete) for contacts.

Purpose:

The Data Management module is responsible for manipulating data

- Uses an ArrayList to store Contact objects, providing a dynamic data structure.
- Implements methods to add new contacts, delete existing ones, and update contact information.

Module 3: Event Handling

- Implements the ActionListener interface for button click events.
- Uses EventListeners for any input from the Keyboard on the text fields.

Module 4: Contact Model

- Defines a class Contact to represent individuals contacts.

Purpose:

This module defines the data structure for individual contacts.

- Implements the Contact class as an inner static class.
- Defines the properties of a contact (name and phone number).

Phonebook application pseudocode.

This pseudocode provides an overview of our phonebook software.

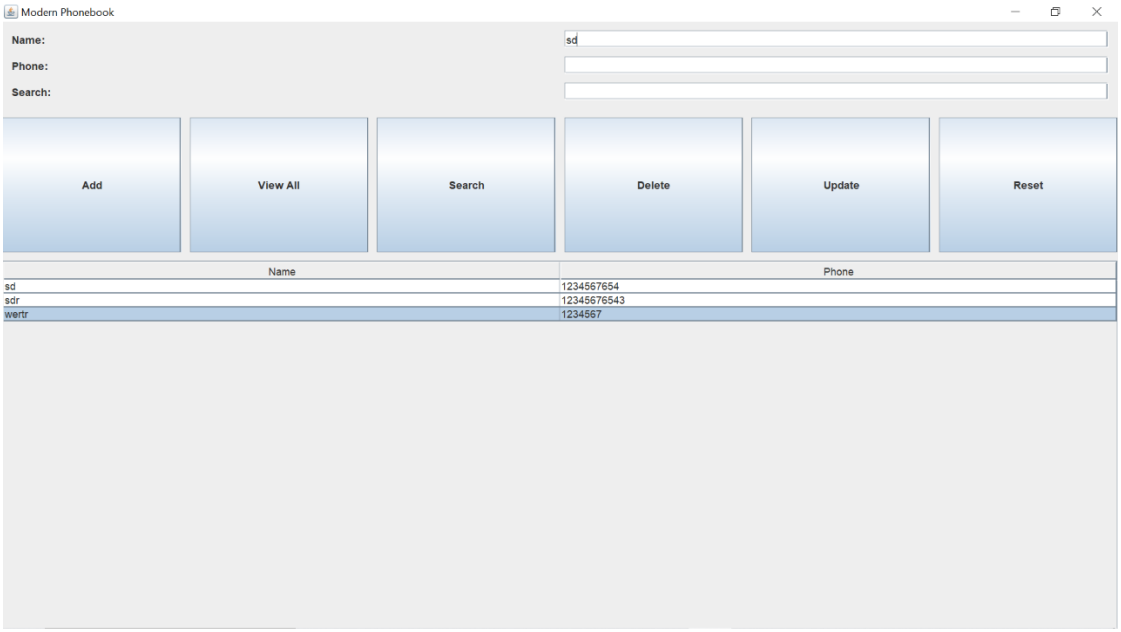
It provides a clear overview of how the different parts of the software interact and what their main responsibilities are.

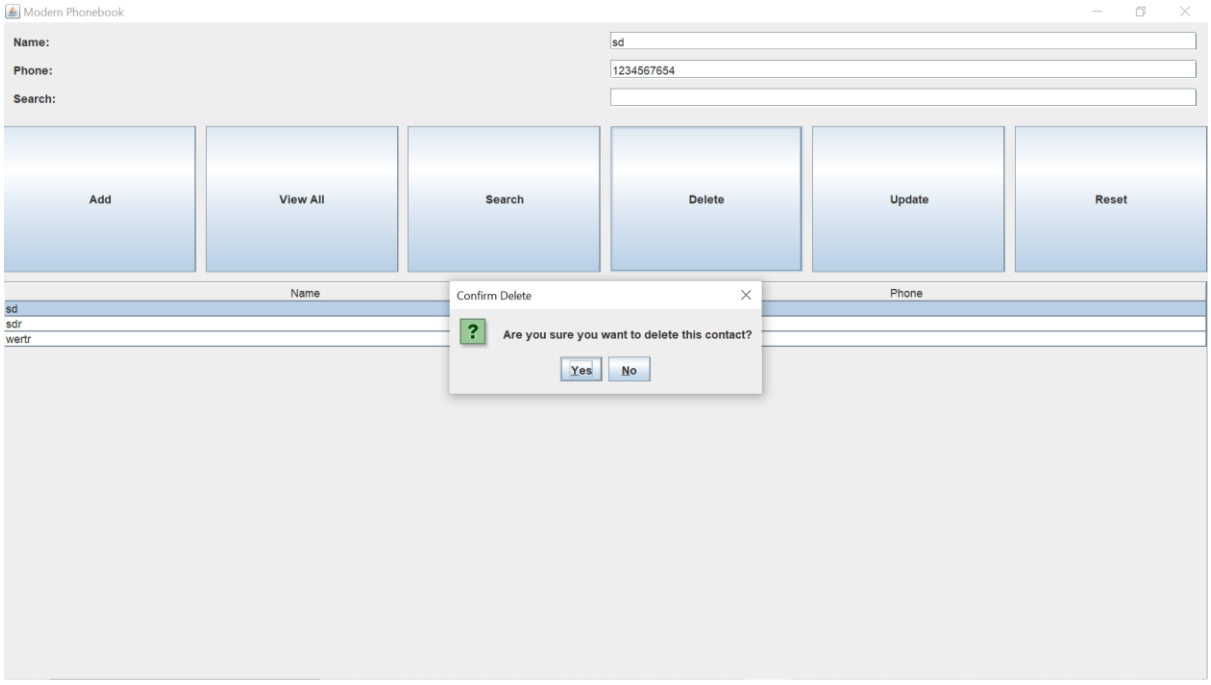
```
Start

    Prompt user for name and phone number

    Get name and phone number

    //Store name and phone number

     If (e.getSource() == addButton) Then

       String name = nameField.getText.trim && String phone = phoneField.getText.trim

       If (isValidName(name) && isValidPhone(phone)) Then

           If ( !isDuplicateContact(name, phone,)

           //New contact added

           contactTableModel.addRow(new Object[](name, phone))

                Display “Contact successfully added”
```

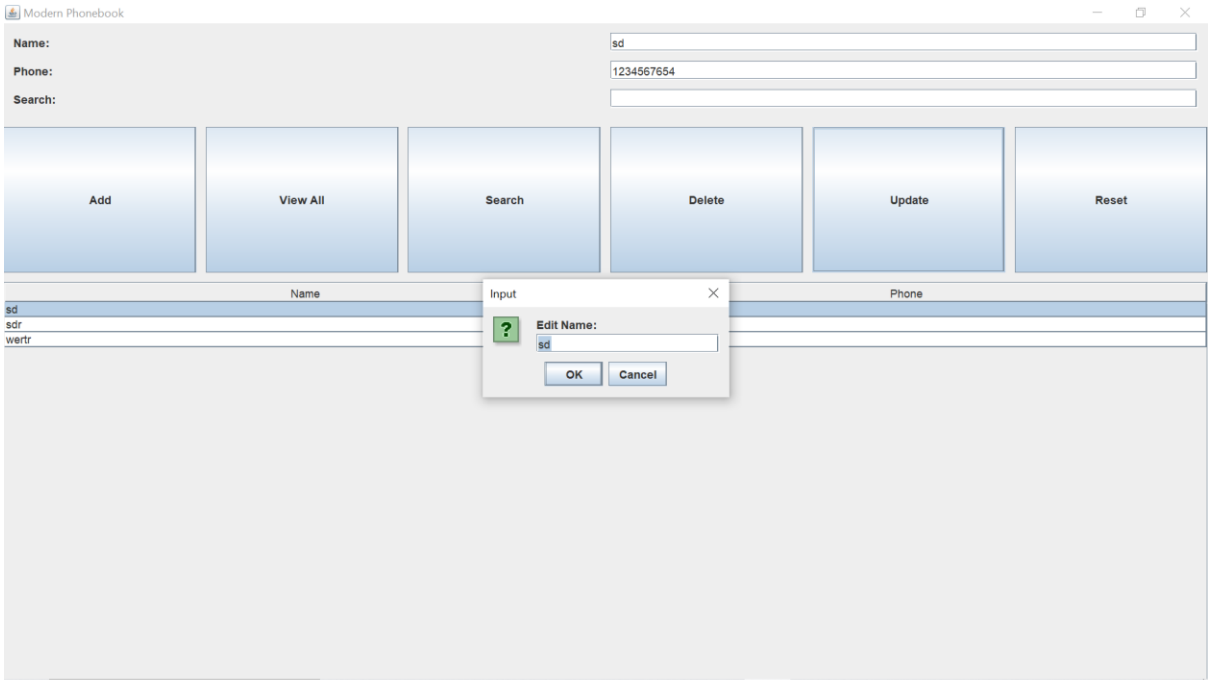
```
            Else
                Display “This contact already exists”
        Else
            Display “Invalid input. Name must contain only letters, and phone must contain only digits."
                Endif
        Endif
    Endif
    // Search stored contacts
    IF (e.getSource() == searchButton) Then
        search = searchField.getText
            If (!search.isEmpty) Then
               searchContact(search)
            Else
                Display “Please enter a name or phone number to search.”
             Endif
      Endif

     // Delete an existing contact
     If (e.getSource() == deleteButton)Then
        selectedRow = contactTable.getSelectedRow
          If (selectedRow != -1)Then
              Display “Are you sure you want to delete this contact?”
                  If (userOption = Yes) Then
                      Display “Contact successfully deleted”
                   Endif
          Endif
      Endif
```

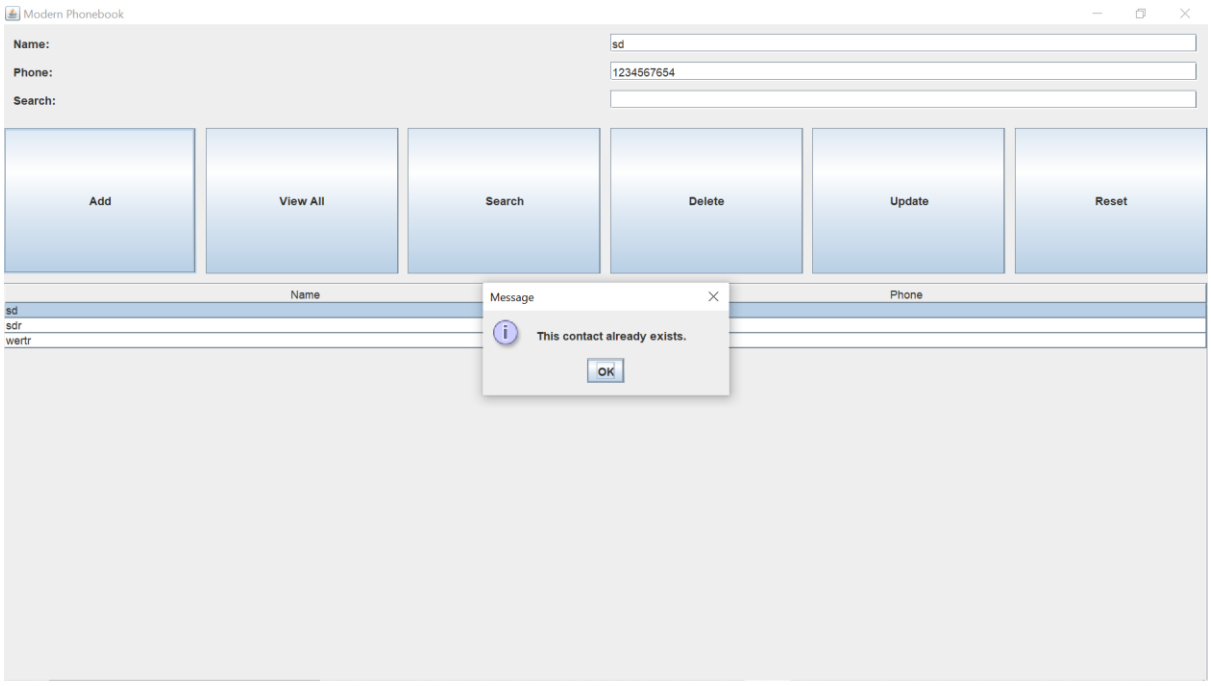
```
// Update an existing contact
If (e.getSource() == updateButton) Then
    selectedRow = contactTable.getSelectedRow
        If (selectedRow  ! = -1)  Then
            newName = (String)
            newPhone = (String)
            if (newName != null && newPhone != null && !newName.isEmpty &&
!newPhone.isEmpty)Then
                    Display “Contact updated successfully”
             Else
                Display “Invalid input”
              Endif
          Endif
    Endif
```

SCREENSHORT FROM THE SOFTWARE


Modern Phonebook
Name:
sd
Phone:
Search:
Add
View All
Search
Delete
Update
Reset
Name
Phone
sd
1234567654
sdr
12345676543
wertr
1234567


Modern Phonebook
Name:
sd
Phone:
1234567654
Search:
Add
View All
Search
Delete
Update
Reset
Name
Phone
sd
sdr
wertr
Confirm Delete
Are you sure you want to delete this contact?
Yes
No


Modern Phonebook
Name:
sd
Phone:
1234567654
Search:
Add
View All
Search
Delete
Update
Reset
Name
Phone
sd
sdr
wertr
Input
Edit Name:
sd
OK
Cancel


Modern Phonebook
Name:
sd
Phone:
1234567654
Search:
Add
View All
Search
Delete
Update
Reset
Name
Phone
sd
sdr
wertr
Message
This contact already exists.
OK

Roles played by Team Members

Shane Swartbooi	Wrote the introduction and description of the project
Hofeni Haamandike	Worked on the code and analyzed the algorithm of the project
Isack Teofilus	Typed the assignment assisted in the algorithm of the project
Jayden Nakakuwa	Worked on the Java code of the project assisted by the entire crew
Fabregas Dimba	Worked on the algorithm of the project assisted by Isack and Hofeni