Here's a README for your PhoneBook Management System project. This document explains the purpose, functionality, and usage of the application.

---

# PhoneBook Management System

## Overview

The PhoneBook Management System is a Java Swing application designed to manage a list of contacts. Users can add, delete, update, search, and display contacts through a simple graphical user interface (GUI). The application uses a linked list to store contacts, ensuring efficient management of data.

## Features

- \*\*Add Contact\*\*: Allows users to add new contacts by entering a name and a phone number.

- \*\*Delete Contact\*\*: Enables users to remove an existing contact by name.

- \*\*Update Contact\*\*: Users can update the name and phone number of an existing contact.

- \*\*Search Contact\*\*: Users can search for a contact by name and view their details.

- \*\*Display All Contacts\*\*: Displays a list of all contacts stored in the phonebook.

## Technologies Used

- Java

- Swing (for GUI)

- LinkedList (for contact management)

## Classes

### Contact

This class represents a single contact with two properties:

- `name`: The name of the contact.

- `phoneNumber`: The phone number of the contact.

### PhoneBook

This class manages the collection of contacts using a `LinkedList`. It includes methods for adding, deleting, updating, searching, and displaying contacts.

#### Methods

- `addContact(String name, String phoneNumber)`: Adds a new contact to the phonebook.

- `deleteContact(String name)`: Deletes a contact by name.

- `updateContact(String oldName, String newName, String newPhoneNumber)`: Updates a contact's details.

- `searchContact(String name)`: Searches for a contact by name.

- `displayContacts()`: Returns a string containing all contacts.

### PhoneBookSystemDsa

This is the main class that creates the GUI and handles user interactions. It uses `JFrame`, `JTextArea`, `JTextField`, and various `JButton` components to create a user-friendly interface.

## How to Use

1. \*\*Run the Application\*\*: Compile and run the `PhoneBookSystemDsa` class.

2. \*\*Add a Contact\*\*:

- Enter the contact's name and phone number.

- Click the "Add Contact" button.

3. \*\*Delete a Contact\*\*:

- Enter the contact's name.

- Click the "Delete Contact" button.

4. \*\*Update a Contact\*\*:

- Enter the contact's name.

- A prompt will appear to enter the new name and phone number.

5. \*\*Search for a Contact\*\*:

- Enter the contact's name.

- Click the "Search Contact" button to view the details.

6. \*\*Display All Contacts\*\*:

- Click the "Display All Contacts" button to see the full list.

## Requirements

- JDK (Java Development Kit)

- IDE (Integrated Development Environment) such as Eclipse, IntelliJ IDEA, or NetBeans

## Conclusion

This PhoneBook Management System provides a simple yet effective way to manage contacts. It can be further enhanced by adding features such as data persistence (saving contacts to a file or database), input validation, and advanced search options.

---

Feel free to modify any sections to better suit your project's specifics or to add any additional features that you plan to implement!

Ryan Kaseke

224045210

Jordan Hashikuni

224081306

KOMBUNGU RITJANOVIPO

223058246

224071408 Junias L.M Philipus

Nestor Amukwaya Iipinge

224034022