**JAVA PROJECT REPORT**

**TOPIC- CONTACT MANAGEMENT LIST**

**AADITYA PATEL-60009200072**

**SAMEEP KARIA-60009200082**

**Introduction-**

Our project is to manage a contact list, after taking contact information as input (Name and contact no.) and store it. We have also provided with the search functionality where the searched contact no. or name registered in the system will be display. Various GUI based tools Swing and awt is used to make GUI. Here the contact information is stored in the list.

**Full Code-**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.\*;

import java.util.ArrayList;

import java.awt.event.\*;

public class AddressBook extends JFrame implements ActionListener {

    public static final String FILE\_NAME = "addressBook.dat";

    JTextField searchField = new JTextField(10);

    JTextArea searchResult = new JTextArea();

    JButton searchButton = new JButton("Search");

    JButton resetButton = new JButton("Reset");

    JButton checkAllButton = new JButton("Check All");

    JButton checkNoneButton = new JButton("Check None");

    JCheckBox nameCheckBox = new JCheckBox("Name");

    JCheckBox phoneCheckBox = new JCheckBox("Phone");

    /\*\*

     \* AddressBook Constructor

     \*/

    public AddressBook() {

        // sets the title of the JFrame

        super("AddressBook");

        setLayout(new BorderLayout());

        // default the check boxes to true when the application starts

        setCheckBoxes(true);

        // attach the action listener to the buttons

        searchButton.addActionListener(this);

        resetButton.addActionListener(this);

        // create a panel for the filters

        JPanel filterPanel = new JPanel();

        filterPanel.setLayout(new BoxLayout(filterPanel, BoxLayout.Y\_AXIS));

        filterPanel.add(Box.createVerticalGlue());

        filterPanel.add(new JLabel("Search Filters:"));

        filterPanel.add(Box.createRigidArea(new Dimension(0, 5)));

        filterPanel.add(nameCheckBox);

        filterPanel.add(phoneCheckBox);

        filterPanel.add(Box.createRigidArea(new Dimension(0, 5)));

        filterPanel.add(checkAllButton);

        filterPanel.add(checkNoneButton);

        filterPanel.add(Box.createVerticalGlue());

        // create a panel for the buttons

        JPanel buttonPanel = new JPanel();

        buttonPanel.add(searchField);

        buttonPanel.add(searchButton);

        buttonPanel.add(resetButton);

        // add the components to the JFrame

        add(new JLabel("Address Book"), BorderLayout.NORTH);

        add(searchResult, BorderLayout.CENTER);

        add(filterPanel, BorderLayout.EAST);

        add(buttonPanel, BorderLayout.SOUTH);

        // have the application exit when the window is closed

        this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        // set the size and make the JFrame visible

        this.setPreferredSize(new Dimension(400, 300));

        this.pack();

        this.setVisible(true);

    }

    /\*\*

     \*  Search through the contacts and return a match

     \*/

    private void search() {

        java.util.List<Contact> contacts = (java.util.List<Contact>) readObject();

        String searchText = searchField.getText().toLowerCase();

        searchResult.setText("Found the following results:\n");

        for(Contact contact: contacts) {

            if (nameCheckBox.isSelected()

                    && contact.getName().toLowerCase().contains(searchText)) {

                searchResult.append(contact.toString());

            } else if (phoneCheckBox.isSelected()

                    && contact.getPhone().toLowerCase().contains(searchText)) {

                searchResult.append(contact.toString());

            }

        }

    }

    /\*\*

     \* Resets the state of the application

     \*/

    private void reset() {

        // reset the text fields

        searchField.setText("");

        searchResult.setText("");

        // reset the filters

        setCheckBoxes(true);

    }

    /\*\*

     \* Sets the value of the checkboxes

     \* @param value the boolean value to set the check boxes

     \*/

    private void setCheckBoxes(boolean value) {

        nameCheckBox.setSelected(value);

        phoneCheckBox.setSelected(value);

    }

    /\*\*

     \*

     \* @param e

     \*/

    @Override

    public void actionPerformed(ActionEvent e) {

        JButton sourceButton = (JButton) e.getSource();

        System.out.println(String.format("%s button pressed", sourceButton.getText()));

        if (sourceButton.equals(searchButton)) {

            search();

        } else if (sourceButton.equals(resetButton)) {

            reset();

        } else if (sourceButton.equals(checkAllButton)) {

            setCheckBoxes(true);

        } else if (sourceButton.equals(checkNoneButton)) {

            setCheckBoxes(false);

        } else {

            System.out.println("Invalid button");

        }

    }

    /\*\*

     \* Writes an object to the file

     \* @param o the object to write to the file

     \*/

    public static void writeObject(Object o) {

        ObjectOutputStream outFile = null;

        try {

            // instantiate the file writer

            outFile = new ObjectOutputStream(new FileOutputStream(FILE\_NAME));

            // write the object to the file

            outFile.writeObject(o);

            outFile.close();

        } catch (IOException ioe) {

            ioe.printStackTrace();

            System.out.println("File not found");

        } finally {

            // do nothing

        }

    }

    /\*\*

     \* Reads an object from the file

     \* @return the reconstructed object from the file

     \*/

    public static Object readObject() {

        ObjectInputStream inFile = null;

        Object result = null;

        try {

            // instantiate the file reader

            inFile = new ObjectInputStream(new FileInputStream(FILE\_NAME));

            // read in the object

            result = inFile.readObject();

            inFile.close();

        } catch (IOException ioe) {

            ioe.printStackTrace();

            System.out.println("File not found");

        } catch (ClassNotFoundException e) {

            e.printStackTrace();

            System.out.println("Class not found");

        } finally {

            // do nothing

        }

        return result;

    }

    public static void ContactInfo()

    {

        java.util.List<Contact> contacts = new ArrayList<Contact>();

        // Creating a new frame using JFrame

        JFrame f

            = new JFrame(

                "Create Contact");

        // Creating the labels

        JLabel l1, l5;

        // Creating three text fields.

        JTextField t1, t2;

        // Creating two buttons

        JButton b1, b2;

        // Naming the labels and setting

        // the bounds for the labels

        l1 = new JLabel("Name:");

        l1.setBounds(50, 50, 100, 30);

        l5 = new JLabel("Mobile No:");

        l5.setBounds(50, 120, 70, 30);

        // Creating the textfields and

        // setting the bounds for textfields

        t1 = new JTextField();

        t1.setBounds(150, 50, 130, 30);

        t2 = new JTextField();

        t2.setBounds(150, 120, 130, 30);

        // Creating one button for Saving

        // and other button to close

        // and setting the bounds

        b1 = new JButton("Save");

        b1.setBounds(150, 300, 70, 30);

        b2 = new JButton("close");

        b2.setBounds(420, 300, 70, 30);

        // Action listener to close the form

        b2.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e)

            {

                f.dispose();

            }

        });

        // Adding action listener

        b1.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e)

            {

                String s1 = t1.getText();

                String s2 = t2.getText();

                Contact[] conarr=new Contact[100];

                int i=0;

                conarr[i] = new Contact(s1,s2);

                contacts.add(conarr[i]);

                writeObject(contacts);

                i++;

                JOptionPane

                    .showMessageDialog(

                        f,

                        "Successfully Saved"

                            + " The Details");

            }

        });

        // Default method for closing the frame

        f.addWindowListener(new WindowAdapter() {

            public void windowClosing(WindowEvent e)

            {

                System.exit(0);

            }

        });

        // Adding the created objects

        // to the frame

        f.add(l1);

        f.add(t1);

        f.add(t2);

        f.add(l5);

        f.add(b1);

        f.add(b2);

        f.setLayout(null);

        f.setSize(700, 600);

        f.setVisible(true);

    }

    /\*\*

     \* The entry point to the application

     \* @param args

     \*/

    public static void main(String args[]) {

        ContactInfo();

        // instantiate the address book

         new AddressBook();

    }

}

class Contact implements Serializable {

    private String name;

    private String phone;

    public Contact(String name, String phone) {

        this.name = name;

        this.phone = phone;

    }

    /\*\*

     \*

     \* @return a String representation of the Contact

     \*/

    @Override

    public String toString() {

        return String.format("%s\n\nName: %s\nPhone: %s\n",

                                super.toString(), name, phone);

    }

    public String getName() {

        return name;

    }

    public void setName(String name) {

        this.name = name;

    }

    public String getPhone() {

        return phone;

    }

    public void setPhone(String phone) {

        this.phone = phone;

    }

}

**Output-**



