#include <iostream>

#include <fstream>

#include <vector>

#include <sstream>

using namespace std;

class Contact

{

public:

    Contact(string name, string phone) : name(name), phone(phone) {}

    string getName() const

    {

        return name;

    }

    string getPhone() const

    {

        return phone;

    }

    void setName(string name)

    {

        this->name = name;

    }

    void setPhone(string phone)

    {

        this->phone = phone;

    }

private:

    string name;

    string phone;

};

class PhoneBook

{

    vector<Contact> contactList;

public:

    void addContact()

    {

        string name, phone;

        cout << "Enter name: ";

        cin.ignore();

        getline(cin, name);

        cout << "Enter phone number: ";

        getline(cin, phone);

        contacts.push\_back(Contact(name, phone));

        saveContactsToFile();

        cout << "Contact added successfully!" << endl;

    }

    void searchContact()

    {

        string name;

        cout << "Enter name to search for: ";

        cin.ignore();

        getline(cin, name);

        for (const auto &contact : contacts)

        {

            if (contact.getName() == name)

            {

                cout << "Phone number for " << name << ": " << contact.getPhone() << endl;

                return;

            }

        }

        cout << "Contact not found." << endl;

    }

    void modifyContact()

    {

        string name;

        cout << "Enter name of contact to modify: ";

        cin.ignore();

        getline(cin, name);

        for (auto &contact : contacts)

        {

            if (contact.getName() == name)

            {

                string phone;

                cout << "Enter new phone number: ";

                getline(cin, phone);

                contact.setPhone(phone);

                saveContactsToFile();

                cout << "Contact modified successfully!" << endl;

                return;

            }

        }

        cout << "Contact not found." << endl;

    }

    void deleteContact()

    {

        string name;

        cout << "Enter name of contact to delete: ";

        cin.ignore();

        getline(cin, name);

        for (auto it = contacts.begin(); it != contacts.end(); ++it)

        {

            if (it->getName() == name)

            {

                contacts.erase(it);

                saveContactsToFile();

                cout << "Contact deleted successfully!" << endl;

                return;

            }

        }

        cout << "Contact not found." << endl;

    }

    void displayContacts()

    {

        cout << "Contacts:" << endl;

        for (const auto &contact : contacts)

        {

            cout << contact.getName() << " - " << contact.getPhone() << endl;

        }

    }

    void loadContactsFromFile()

    {

        ifstream file("contacts.txt");

        if (file.is\_open())

        {

            string line;

            while (getline(file, line))

            {

                stringstream ss(line);

                string name, phone;

                getline(ss, name, ',');

                getline(ss, phone);

                contacts.push\_back(Contact(name, phone));

            }

            file.close();

        }

    }

    void saveContactsToFile()

    {

        ofstream file("contacts.txt");

        if (file.is\_open())

        {

            for (const auto &contact : contacts)

            {

                file << contact.getName() << "," << contact.getPhone() << endl;

            }

            file.close();

        }

    }

private:

    vector<Contact> contacts;

};

int main()

{

    PhoneBook phonebook;

    phonebook.loadContactsFromFile();

    int choice;

    do

    {

        cout << "\nPHONEBOOK APPLICATION\n";

        cout << "1. Add Contact\n";

        cout << "2. Search Contact\n";

        cout << "3. Modify Contact\n";

        cout << "4. Delete Contact\n";

        cout << "5. Display Contacts\n";

        cout << "6. Exit\n";

        cout << "Enter your choice: ";

        cin >> choice;

        switch (choice)

        {

        case 1:

            phonebook.addContact();

            break;

        case 2:

            phonebook.searchContact();

            break;

        case 3:

            phonebook.modifyContact();

            break;

        case 4:

            phonebook.deleteContact();

            break;

        case 5:

            phonebook.displayContacts();

            break;

        case 6:

            cout << "Exiting program..." << endl;

            break;

        default:

            cout << "Invalid choice, please try again." << endl;

            break;

        }

    } while (choice != 6);

    return 0;

}

