Contact Book Management System

Project Description:

This is a Command Line Interface (CLI) application for managing a contact book. The program allows users to add, delete, search, and display contacts interactively, with data stored persistently in a CSV file. The application ensures input validation and prevents duplicate phone numbers and emails. I googled a bit for beautiful visualization and directory handling with the Python os module. And the project is made using techniques learnt from Ostad classes and using the knowledge learnt from the Library Management project.

Features :

1. **Add Contacts**
    - Saves contact details: Name, Phone, Email, Address.
    - Validates inputs (phone numbers must be digits, and emails must contain `@` and `.`).
    - Prevents duplicate phone numbers or emails.
2. **View Contacts**
    - Displays all saved contacts in a clean, tabular format.
3. **Search Contacts**
    - Search by name or phone number (case-insensitive).
4. **Delete Contacts**
    - Removes a contact by matching the name and phone number.
5. **Data Persistence**
    - Automatically saves all contacts to a CSV file in the current directory.
    - Loads data automatically at program start.

File Structure:

📂 ContactBook
├── main.py # Main driver script with the interactive menu
├── fileHandler.py # Handles file operations (load, write, and check duplicates)
├── validation.py # Validates phone numbers and emails
├── functionality.py # Contains core functionalities (save, delete, search, display contacts)
├── contactBook.csv # Stores all contacts (auto-created if it doesn't exist)
└── README.docx # Documentation for the project

Design Plan :

To achieve modularity and encapsulation I divided the code into 4 .py files/modules.

The program is structured into these four main modules:

1. **Main Driver (`main.py`)**:
    - Handles the user interface and menu system.
2. **Functionality of phonebook(`functionality.py`) :**
    - Holds all the functionality codes .
3. **File Handler (`fileHandler.py`)**:
    - Manages file operations (read, write, check duplicates).
4. **Validation (`validation.py`)**:
    - Ensures data validity for inputs like phone numbers and emails.

.

Here is a diagram showing interaction between the modules :