uOttawa

GNG5300: Full Stack Software DevelopmentPhone

Assignment 1 | Phone Book Management

Professor: Masoud Dorrikhteh

Pouria Bahri - 300352271

Fall 2024

1. Introduction

The Phone Book Management System is a Python-based command-line application that provides users with a straightforward way to manage contact information. The project implements essential functionalities, such as adding, searching, updating, and deleting contacts, along with additional features like input validation for phone numbers and emails, as well as data persistence using JSON.

The project is publicly accessible on GitHub on the following repository:

https://github.com/p0ur1a/GNG5300-Phone-book-management

2. System Overview and Objectives

The primary objective of the Phone Book Management System is to provide users with an intuitive contact management tool that ensures persistence between sessions using JSON files. The system emphasizes modularity and clean, maintainable code, allowing for future feature expansion.

Key Objectives:

- To allow the user to manage contacts via the command-line interface (CLI).
- To persist contact data across multiple sessions using JSON.
- To validate inputs (phone numbers, email addresses) and ensure data integrity.
- To enable users to batch-import contacts from CSV files.

3. Features

a) Add Contact

This feature allows users to add new contacts to the phone book. Each contact consists of:

- First Name
- Last Name
- Phone Number
- Email Address (optional)
- Address (optional)

Validation is performed for the phone number and email format to prevent invalid data from being entered. Phone numbers follow a pattern (e.g., digits only, or specific formats like (123) 456-7890). Emails follow the format example@domain.com.

b) Search Contact

The system provides functionality to search for contacts by name or phone number. It returns the relevant contact(s) if found or informs the user if no matching contact is available.

c) Update Contact

This feature allows the user to update an existing contact's details. The system first prompts the user for the name or phone number of the contact to be updated, and then asks for updated values.

d) Delete Contact

The delete function removes a contact from the phone book. Similar to the update feature, the user is prompted to search for the contact, and upon confirmation, the contact is deleted from the phone book.

e) Data Persistence Using JSON

One of the core features of this system is its ability to save contact data using a JSON file (contacts.json). The system loads this file upon initialization, ensuring that contact data is persistent between runs.

f) Batch Import of Contacts from CSV

A bulk import feature allows users to load multiple contacts from a CSV file. The system reads the CSV file and converts the entries into contact objects, adding them to the phone book while ensuring that valid entries conform to the required format.

g) View Logs

The logger.py module records actions performed on the phone book. Every change (such as adding, updating, or deleting a contact) is logged with a timestamp, which can later be reviewed to understand the sequence of actions taken.

h) View existing contacts

The program allows users to view the already existing contacts. It also allows them to sort the contacts either by first or last name

4. Code Structure

The repository consists of several key Python files that define different components of the system.

a) cli.py

This file handles the command-line interface (CLI) of the application. It is the main entry point where the user interacts with the system. The cli.py file contains the logic to read user input, present available options (e.g., add, search, update, delete contacts), and call the corresponding functions from other modules to execute these operations.

b) phonebook.py

This module contains the core functionality of the phone book system. It defines the PhoneBook class, which manages the contacts, and provides the methods for adding, updating, searching, and deleting contacts. The class also handles saving and loading contact information from the contacts.json file.

c) utils.py

The utils.py file contains utility functions that support the main operations of the phone book, like colored printing the messages and file handling.

d) tests.py

This file contains unit tests for the application. The tests ensure that the core functionality (like adding, searching, updating, deleting, and loading contacts) works as expected. By running tests.py, the integrity of the phone book system can be verified.

e) contacts.json

This file is where all contact information is stored. It is a JSON file that holds the data for all contacts added to the phone book. The system reads from this file when initialized and writes to it whenever contact information is modified, ensuring data persistence between sessions.

f) logs.txt

The logs.txt file records logs of user and program actions. Every operation performed (such as adding, updating, or loading contacts) is logged with a timestamp. This helps in tracking the history of operations and provides a record for future reference.

g) sample_contacts.csv

This is a sample CSV file used for demonstrating the batch import functionality. Users can import multiple contacts at once by providing a properly formatted CSV file like this one. The system reads the contacts from the CSV and adds them to the phone book, while also validating each contact's details.

5. System Design

The design of the Phone Book Management System is based on the principles of modularity and separation of concerns. This makes the system flexible, easy to maintain, and extensible.

- **Modular Approach:** Breaking down functionality into different modules ensures that changes to one part of the system do not adversely affect others.
- **Data Persistence**: The choice to use JSON for data storage ensures compatibility with many applications and allows easy human-readability for debugging purposes.

6. Testing and Validation

The system was tested with various input scenarios to ensure its robustness. Validation tests were performed to check whether the system correctly handled invalid phone numbers and emails. JSON file consistency was also ensured across multiple operations (add, delete, update).

7. Conclusion

The Phone Book Management System is a functional and well-structured Python application. It achieves the goals of contact management, data persistence, input validation, and logging.