George Sheridan Terminal App

Musician Contact Book

Overview

The mcontacts app is designed to function as a standard contact app would but with an additional search capability.

The search function is configured to allow standard search by name but also gives you the ability to search by 'instrument or job (eg producer or sound engineer)' and location. If you are a producer or musical director and need a specific player for a show or session in a place thats unfamiliar you, rather than manually scroll through your phone contacts trying to remember who is who, you will be able to bring up a list of all the relevant people you know efficiently.

Overview 2

The app starts with a main menu. 6 options are presented to the user:

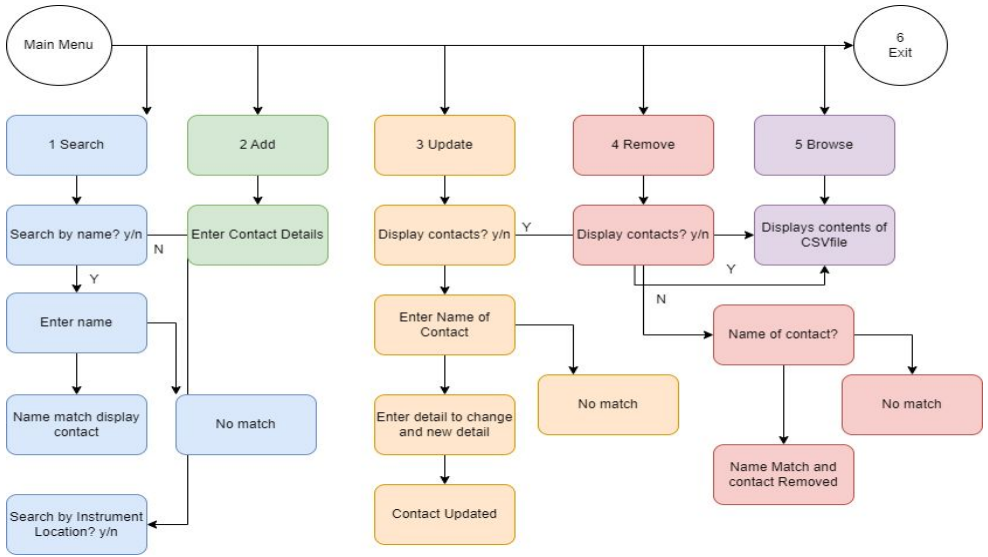
1. Search Contacts
2. Add Contacts
3. Update Contacts
4. Remove Contacts
5. Browse Contacts
6. Exit the program

Application Navigation

Heres a rough diagram of how the app is set up:

All functions are running through the main menu loop which only ends when the user selects option 6. ie Each function returns to the main menu once it has reached the end of its path.

More steps are included in the code to prevent infinite looping and bad user input. For example in the add function, if statements do not allow the user to enter a blank field.



Functions

The app first uses a try except block to create a new csv file if there is none present.

Then the create menu function is called and the user sees the main functions displayed in the terminal.

1 Search - As explained the search function acts as you would expect but also provides the ability to search via instrument and location. This is achieved by a series of if and else statements to get user input and then using the csv module to match the relevant row strings (ie either name, or instrument and location), and then display the row or rows from the csv file in the terminal.

2 Add - This prompts the user to add a name, phone, email, instrument, and city detail to create a new contact. This is done by asking for the user input and using the csv module to write the input to the csv file. There are a number of if statements contained in this function to prevent the user from adding blank entries. Blank details in an entry can cause confusion if later the user wants to update or search for that entry.

Functions 2

3. Update - This function allows a user to search for an entry and then change any of the details of that entry. It first gives the option to display all the entries in the address book for ease of reference. Then asks for the contact name, and if it matches with an entry, displays the full entry. Then it asks which detail (meaning the actual entry contents, not ‘phone, email etc’, perhaps that could be changed in v2.0), and then what to change it to. The swap is achieved by searching the entry row for the detail and swapping it with the new detail. Then adding that entry to a list, then adding all the other entries to the list, then writing the list into the csv file (overwriting the previous contents).

4. Remove - This allows the user to remove a contact. It works in the same way as the other functions. Offering to display all contacts and then getting the name of the entry to remove, matching it, then adding all the other names in the contact file to a list, and overwriting that new list to the csv file.

5. Browsing - This is a simple function that uses the csv module to open the contact file and display all of it contents in the terminal. I’ve used the ‘*’ operator(?) before ‘row’ to ensure that no brackets or commas are displayed in the terminal for a better aesthetic.

Modules

Rich - Ive used this module to add some colour and emojis to parts of the output to make the app experience slightly more pleasing for the user. Note this caused major headaches when designing a test for the create menu function. Because pytest was testing to see if the text displayed at output matched the code, it threw errors because it was not excepting the ansi code from this module. I used this article with the ‘re’ module, and a lot of trial and error to make it work: https://stackoverflow.com/questions/14693701/how-can-i-remove-the-ansi-escape-sequences-from-a-string-in-python#14693789

Art - I used this to display some fun ASCII text on exiting the program.

Csv - Used extensively to interact with the csv file.

Time - Used in the progress bar feature of the rich module, enabled me to control the speed of the progression.