Contact Book

We come across lots of people daily. We make acquaintances and friends. We get their contacts to keep in touch later on. Sadly, keeping the received contact details can be hard. One way to do this is to write the contact details down. But this is not secure as the physical book can easily be lost.

This is where the Contact Book project comes in. A contact book is a tool for saving a contact's details, such as name, address, phone number, and email address. With this contact book project, you can build a software tool that people can use to save and find contact details.

With the contact book project idea, users can save their contacts with less risk of losing the saved contact details. It'll always be accessible from their computer, through the command-line.

Examples of Contact Book Tools

There are Contact Book applications, but it's rare to find command-line Contact Book products, as most are web, mobile, or GUI applications.

Note: For an in-depth explanation of how to build a GUI-based contact book, check out Build a Contact Book With Python, PyQt, and SQLite.

Here are some implementations of the Contact Book idea:

- Simple Contacts
- Pobuca Connect

Technical Details

The main objective of this project is to save contact details. It's important that you set up the commands users can use to enter the contact details. You can use the argparse or click command-line frameworks. They

abstract a lot of complex stuff, so you only have to focus on the logic to be run when executing commands.

Some features you should implement include the commands to delete contacts, update contact information, and list saved contacts. You can also allow users to list contacts using different parameters, such as alphabetical order or contact creation date.

Since it's a command-line project, the SQLite database will be fine for saving contacts. SQLite is user-friendly to set up. You may save the contact details in a file, but a file will not offer the benefits you can gain from using SQLite, such as performance and security.

To use the SQLite database in this project, the Python sqlite3 module will be very useful.

Extra Challenge

Remember how the database is stored on the user's computer? What if something happens, like the user losing their files? It means they'll also lose the contact details.

You can challenge yourself further and backup the database to an online storage platform. To do this, you can upload the database files to the cloud at certain intervals.

You can also add a command that allows users to backup the database themselves. This way, the user can still have access to the contacts if the database file is lost.

You should note that you may need some form of identification, so the contact book can tell which database file belongs to which user. Implementing a user authentication feature is one way to go about it.

Directory Tree Generator

Directories are like family trees: each directory has a particular relationship with other directories. No directory ever stays on its own, except an empty root directory.

When you're working with files and directories, it is difficult to see the relationship between directories, as you can only see what exists in the current directory. You're either using a file manager or working from the command-line.

With a Directory Tree Generator, you can see the relationship between files and directories like a tree or a map.

This makes it easier to understand the positioning of files and directories. A directory tree map is important when you're explaining certain concepts, and a Directory Tree Generator makes it easier to get a visual representation of the file and directory relationships.

Examples of Directory Tree Generators

Here are some implementations of the Directory Tree Generator idea:

- Tree
- Dirtreex

Technical Details

The main objective of the Directory Tree Generator is to visualize the relationships between files and directories. The os library can be very useful in listing the files and directories in a chosen directory.

Using a framework such as docopt or argparse helps abstract a lot of stuff, allowing you to focus on writing code for the application's logic.

In the application's logic, you can decide how you want to represent files or directories. Using different colours is a brilliant way to go about it. You can use the colored library to print the files and directories in different colours.

You can also decide how deep you'd like the Directory Tree Generator to go. For example, if a directory has children directories twelve levels deep, you may decide to go only as deep as the fifth level.

If you wish, you can also let the user decide how deep they want the Directory Tree Generator to go.

Extra Challenge

Since the results of the generated directory tree will be on the command-line, you can go one step further. You can have the generator create images of the directory tree, so it'll basically turn the text into an image.

You'll find the pillow library useful for doing this.

Bulk File Rename Tool

Sometimes, you need to name all the files in a directory according to certain conventions. For example, you can name all the files in a directory with File0001.jpg, where the numbers increase based on the number of files in the directory. Doing this manually can be stressful and repetitive.

The Bulk File Rename Tool allows users to rename a large number of files, without having to manually rename files.

This saves users a lot of time. It spares them the trouble of having to do boring repetitive work and make mistakes. With the Bulk File Rename Tool, users can rename files in a couple of seconds without any mistakes.

Examples of Bulk File Rename Tools

Here are some implementations of the Bulk File Rename idea:

- Ren
- Rename

Technical Details

The main objective of this project idea is to rename files. So, the application needs to find a way to manipulate the target files. The os, sys, and shutil libraries will be useful for a large part of this project.

Your users will be able to rename all the files in the directory, using naming conventions. Therefore, they should be able to pass in the naming convention of choice. The regex module will help match the required naming patterns, if you understand how regex works.

A user may want to pass in a naming convention such as myfiles as part of the commands and expect that the tool renames all the files like

myfilesXYZ, where XYZ is a number. They should also be able to choose the directory where the files to be renamed are.

Extra Challenge

The major challenge in this project is to rename all the files in a directory. But users may only need to name a certain number of files. To test your skills, you can implement a feature to allow users to choose the number of files to be renamed, instead of all the files.

Note that renaming only a certain number of files will require the tool to sort the files based on alphabetical order, time of file creation, or file size, depending on the user's requirements.