PYTHON

PMS (Password Management System) - (GUI Based)

Scenario

You have been assigned the task of developing a password manager application using Python. The application should allow users to securely store, edit, and delete their passwords for various online accounts. Additionally, it should have a Material Design user interface (UI) for an enhanced visual experience. Furthermore, the application should provide the ability to generate a backup of the stored passwords in a CSV file format.

Task Description

Your task is to create a password manager application with the following features:

- **User Registration:** Users should be able to create an account by providing a unique username and a master password.
- **Password Storage:** The application should securely store user passwords for different online accounts, such as email, social media, and banking.
- **Password Editing:** Users should be able to edit their stored passwords, allowing them to update login credentials when needed.
- **Password Deletion:** Users should have the ability to delete stored passwords for accounts they no longer wish to manage.
- **Material UI:** Implement a visually appealing user interface following the Material Design guidelines to enhance the user experience. Consider using a Python library that supports Material UI, such as PyMaterial or KivyMD, for UI development.
- **CSV Backup:** Provide functionality to generate a backup of the stored passwords in a CSV file format. Users should be able to export their password data for backup purposes or for importing into other password management tools.

Requirements:

- Develop the application using Python programming language.
- Utilize appropriate libraries for encryption, password hashing, and CSV file manipulation.
- Implement a secure authentication mechanism to verify the master password.
- Store the encrypted passwords in a secure manner, either using a local database or file.
- Use a Python library that supports Material UI, such as PyMaterial or KivyMD, for designing the user interface.
- Include features for password editing and deletion to provide flexibility to users.
- Implement the CSV backup functionality to allow users to export their password data.

Libraries (just for Help):

- **Python:** The application should be developed using the Python programming language.
- **Cryptography:** A library for secure encryption and password hashing.
- **bcrypt:** A library for secure password hashing.
- **PyMaterial (optional):** A Python library for creating Material Design user interfaces.
- **KivyMD (optional):** A Python library for creating Material Design user interfaces.
- **CSV (built-in):** Python's built-in CSV module for handling CSV file manipulation.

Explanation

Developing a password manager with a Material UI requires knowledge of Python programming, encryption techniques, secure password storage, and CSV file manipulation. Use appropriate libraries for encryption and password hashing, such as cryptography or bcrypt, to ensure the security of stored passwords. Consider utilizing a Python library that supports Material UI, such as PyMaterial or KivyMD, to design a visually appealing and intuitive user interface. Implement the required functionalities for password storage, editing, deletion, and CSV backup.

Note: Please ensure that all the required files for the project are organized within a single folder. The folder should contain the source code files, an output recorded video (if applicable), an executable file (if applicable), a list of libraries used, and a Readme file. The Readme file should provide clear instructions on how to install and run the application or project, including any necessary dependencies or configurations. By following this organization, it will be easier to review and evaluate the project