Password Manager Online

Documentação

Contexto

Após desenvolver um gestor de *passwords* em modo *offline* deparei-me com um problema:
Sempre que quisesse aceder às minhas contas precisava de andar com uma *pen* com o programa e base de dados atrás de mim pois era lá que tinha todas as contas atualizadas. Ora certo dia precisava de entrar numa conta enquanto estava na universidade e reparei que tinha deixado a *pen* no computador de casa. Surgiu então o problema da indisponibilidade dos dados, ou seja, não conseguia aceder aos meus dados quando precisava.

Motivado por este problema ter surgido, decidi recriar o meu antigo gestor de contas *offline* para uma versão *online.*

Como foi desenvolvido o novo *Password Manager*?

Tal como o modelo anterior, este programa também foi desenvolvido em *Python* e adota algumas funções e estruturas de código do programa anterior.

A maior diferença está certamente na forma como a base de dados foi implementada. Enquanto anteriormente era utilizada uma base de dados *SQL,* nesta nova versão é utilizado um Sistema de Gestão de Bases de Dados *MongoDB*.

O *MongoDB* é uma alternativa aos SGBD's relacionais tradicionais como o *MySQL* ou *SQLite* que era o SGBD utilizado na versão anterior.

O *MongoDB* é especialmente conhecido por ser um SGBD orientado a documentos, isto é, todos os dados são armazenados em documentos no formato *BSON* (*Binary JSON*), que é uma representação binária do *JSON* (*JavaScript Object Notation*). Cada documento é uma unidade independente de dados, semelhante a um registo ou linha em bases de dados relacionais, mas com uma estrutura de dados flexível, permitindo que os documentos de uma coleção possam ter campos diferentes.

Resumidamente, enquanto numa base de dados "comum" /relacional temos registos guardados em tabelas, em *MongoDB* temos documentos guardados em coleções em que cada documento não tem necessariamente de ter os mesmos campos que todos os outros.

	MySQL	MongoDB
Use Case	Legacy applications or applications that require multi-row transactions (i.e. accounting systems)	Real-time analytics, content management, internet of things, mobile apps.
Data Structure	Structured data with clear schema.	No schema definition required.
Risk	Risk of SQL injection attacks.	Less risk of attack due to design.
Analysis	A great choice if you have structured data and need a tranditional relational database.	A great choice if you have unstructred and/ or structured data with the potential for rapid growth.

Figura 1 - Diferenças entre MySQL e MongoDB

Transporte dos dados de *SQLite* para *MongoDB*

Para além de ter de implementar um novo sistema de base de dados também tinha que manter os meus dados que já estavam registados em *SQLite* mas o MongoDB apenas aceita a importação de ficheiros .csv ou .json.

Para exportar os meus dados de *SQLite* para .csv utilizei o programa *DB Browser for SQLite* e abri o meu ficheiro de base de dados nele:

	site	email	username	password	twofa
	Filter	Filter	Filter	Filter	Filter
1	Microsoft	gAAAAABkU-ZfqxD_JMdrLT4Qxgf-...	gAAAAABkU-...	gAAAAABkU-...	gAAAAABkU-...
2	Apple	gAAAAABkU-baQxJ98r_1YmdGha42tR-...	gAAAAABkU-...	gAAAAABkU-bigPqTh7mVxdaJYuOz6-...	gAAAAABkU-bkM3IXjSpPz3-...
3	Mega 365	gAAAAABkU-dTrn9jvU_Uv8mbA-...	gAAAAABkU-...	gAAAAABkU-...	gAAAAABkU-...
4	Mega 26	gAAAAABkU-...	gAAAAABkU-d590FE9nIsBsD-...	gAAAAABkU-...	gAAAAABkU-d7W-...
5	Discord	gAAAAABkU-eLC46dSoOCxOuKaqAMbgISKiTWdK...	gAAAAABkU-eMo4HtG68q-...	gAAAAABkU-...	gAAAAABkU-...
6	Facebook	gAAAAABkU-...	gAAAAABkU-...	gAAAAABlMC7XmKxSHqix7AGjbLTikDIpWONAMW...	gAAAAABkU-...
7	Instagram	gAAAAABkU-...	gAAAAABkU-fioZZ2bx3XCCMRt4-...	gAAAAABkU-...	gAAAAABkU-...
8	Twitter	gAAAAABkU-...	gAAAAABkU-...	gAAAAABkU-...	gAAAAABkU-...
9	Snapchat	gAAAAABkU-gXrY0d2-...	gAAAAABkU-gZrfOHU12gLqBOlMwGQHbBHS6v-...	gAAAAABkU-...	gAAAAABkU-...
10	Twitch	gAAAAABkU-...	gAAAAABkU-...	gAAAAABkU-...	gAAAAABkU-...
11	Netflix	gAAAAABkU-hMvu-xfzJmJRLtI8_lPG-...	gAAAAABkU-...	gAAAAABkU-...	gAAAAABkU-...
12	LinkedIn	gAAAAABkU-hrmRysP5BXbuj3bnVYDg1-...	gAAAAABkU-...	gAAAAABkU-...	gAAAAABkU-htK-...
13	Europass	gAAAAABkU-mDg5tAgJ17s-...	gAAAAABkU-mEfpLeZiQR002szlcmqKE-...	gAAAAABkU-...	gAAAAABkU-mFgIJYB-...
14	Pobre.tv	gAAAAABkU-...	gAAAAABkU-mXoAFkCzfzaC-...	gAAAAABkU-mXO7iLz_9--...	gAAAAABkU-mYuA04N_zE8YGypZcY-...
15	Steam 2	gAAAAABkU-msyBZEokMfjsQhLiKNgdu8bZo-...	gAAAAABkU-...	gAAAAABkU-...	gAAAAABkU-...
16	Steam 26	gAAAAABkU-...	gAAAAABkU-ncxPnO-krQjWEiu-...	gAAAAABkU-nd3w0vlxMUM_9vNVa8blJ3VBs9ZyH-...	gAAAAABkU-...
17	Epic Games	gAAAAABkU-...	gAAAAABkU-...	gAAAAABkU-...	gAAAAABkU-oYNWKSxEd7DSJ-5p-...

Figura 2 - Base de dados SQLite em DB Browser

Após selecionar todos os registos, foi só exportar para CSV:

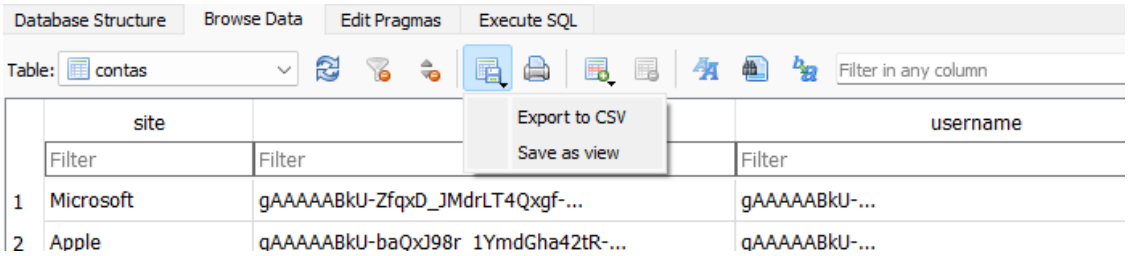

Figura 3 - Exportar para CSV

MongoDB Atlas

O *MongoDB Atlas* é onde toda a base de dados é criada e armazenada de forma online. Com a ajuda de uma das plataformas disponíveis (*Amazon Web Server, Azure* ou *Google Cloud Platform*), o *MongoDB Atlas* disponibiliza-nos a nossa base de dados em *cloud.*

Há várias formas de conseguir aceder à base de dados do *MongoDB Atlas,* uma delas é pelo *MongoDB Compass* que é o programa do *MongoDB* que nos ajuda a fazer tratamentos de dados por exemplo.

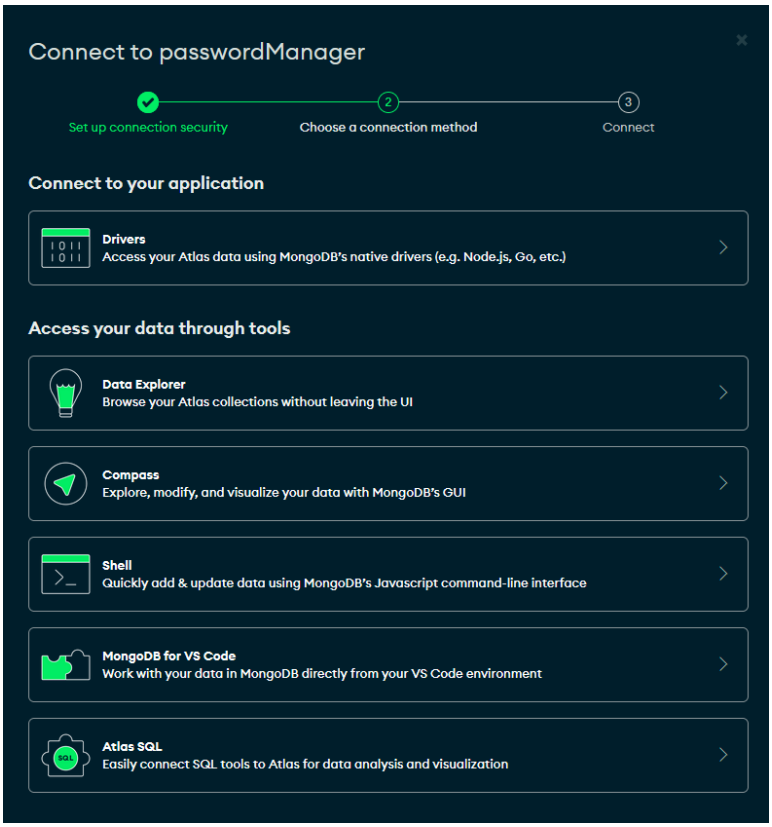

Figura 4 - Opções de ligação ao MongoDB Atlas

Para fazer a ligação do *MongoDB* ao *python* foi utilizada a biblioteca *pymongo.*

Para obter o link de ligação para o *python* basta copiar o "*connection string*" do *MongoDB Compass* e inserir na linha 8 do ficheiro *functions.py*

Criptografia

Criptografia não é mais do que pegar num pedaço de texto e transformá-lo num pedaço de texto ilegível / incompreensível com uma chave secreta. Os dados devem ser de tal forma encriptados que a única maneira possível de voltar a ler o texto original seja através da chave.

Tudo o que está relacionado com a internet precisa de criptografia: comunicações, transferências de ficheiros, bases de dados, ligações a sites, emails, …

Como o novo *password manager* passou a ter o seu núcleo hospedado na internet, é fundamental que o seu conteúdo seja encriptado para uma maior segurança.

Se na versão anterior já estava encriptada porque nunca se sabe quando é que alguém nos vai pôr a mão na *pen,* esta versão não seria diferente porque nunca sabemos quando é que alguém pode tentar entrar na nossa conta do *MongoDB Atlas.*

Para prevenir situações indesejáveis, os dados guardados na base de dados têm de ser obrigatoriamente encriptados. Como os meus dados já estão encriptados da versão anterior então não preciso de o voltar a fazer, apenas continuei a aplicar o mesmo algoritmo de encriptação, o *Fernet*.

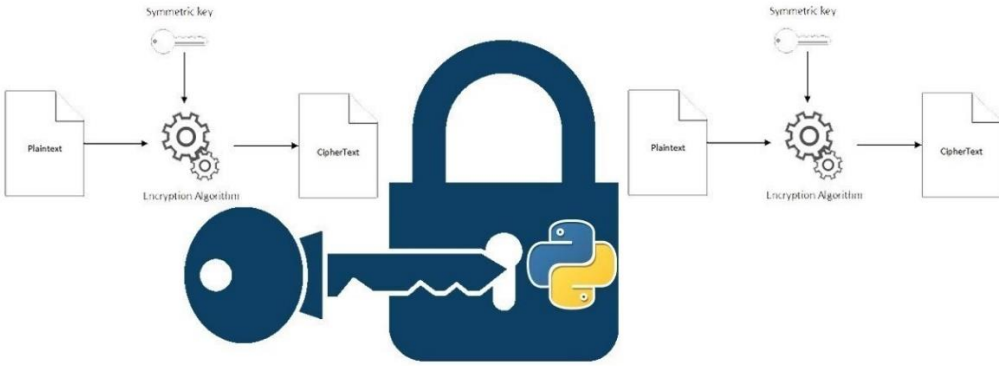

Figura 5 - Fernet

O *Fernet* é uma biblioteca de criptografia simétrica em *Python* que fornece criptografia de dados, especificamente para garantir a confidencialidade dos dados. Foi projetado para ser fácil de usar e oferecer segurança forte. O *Fernet* é construído em cima do *AES* (*Advanced Encryption Standard*), um algoritmo de criptografia amplamente utilizado e confiável.

Esta biblioteca é útil para criptografar e descriptografar dados sensíveis, como *passwords*, *tokens* de autenticação e informações de configuração, antes de armazená-los ou transmiti-los através de uma rede. Com o *Fernet*, é possível proteger facilmente os dados da aplicação de forma eficaz. É popular no mundo *Python* por causa da facilidade de uso e pela segurança robusta que oferece.

Funcionamento do programa

SECRET KEY

O programa está desenhado para funcionar pela linha de comandos. Após seguir as instruções do ficheiro *README.md* e fazer as configurações necessárias vamos ser confrontados com a seguinte situação:

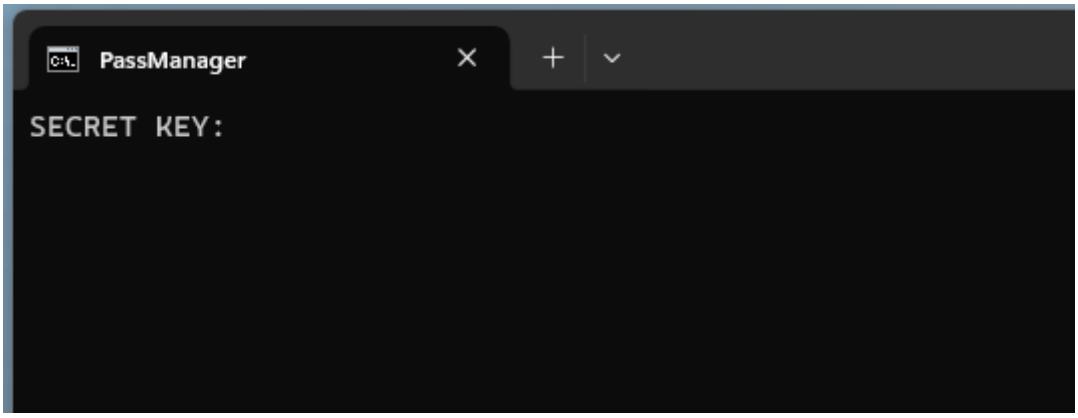

Figura 6 - Inserir Secret Key

A *secret key* refere-se à chave que foi gerada ao executar a função *generate_key()* e que foi imprimida no ecrã. Esta chave é crucial para o correto funcionamento da aplicação. Esta chave é a que vai ser usada pelo algoritmo de criptografia para encriptar e desencriptar os dados da base de dados.

NOTA: Esta chave é crucial para o funcionamento do *password manager.* Uma vez perdida não será possível desencriptar os dados guardados na base de dados. Guarde com cuidado.

Menu de opções

Após a inserção da *secret key* o programa fica pronto a ser utilizado e é então mostrado o seguinte menu de opções:

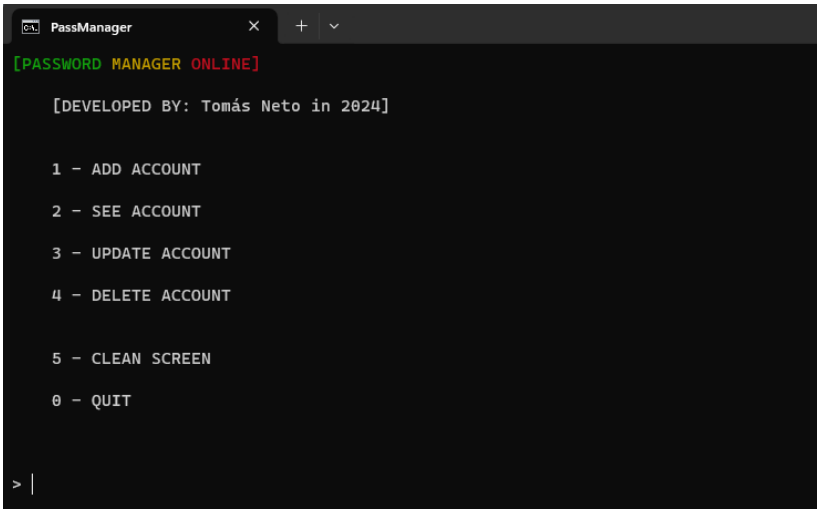

Figura 7 - Menu de opções

Cada opção tem a sua função associada e são autoexplicativas.

1. ADD ACCOUNT → Fazer novos registos na base de dados;
2. SEE ACCOUNT → Visualizar contas existentes e selecionar uma conta para ver o conteúdo;
3. UPDATE ACCOUNT → Atualizar dados de uma conta;
4. DELETE ACCOUNT → Eliminar uma conta da base de dados;
5. CLEAN SCREEN → Apagar o ecrã. Serve por exemplo para situações em que a privacidade é escassa e não queremos que ninguém veja a nossa *password,* nessa situação o ecrã pode ser limpo rapidamente com esta função;
0. QUIT → Sair do programa.

Posto isto, o programa está pronto a ser utilizado. Os nossos dados são guardados de forma segura e é possível aceder-lhes a qualquer momento desde que tenhamos o programa à mão, ligação à internet e não tenhamos perdido a *secret key.*

Código

Para explicações sobre o código posso ser contactado pelo LinkedIn ou email que se encontram no meu portefólio.

Conclusão

Para além do objetivo de colocar o serviço *online* também tinha como objetivo simplificar o código do programa. Ambos os objetivos foram concluídos como é possível verificar pelo código no repositório do Github.

Neste momento já não preciso de andar com *pens* atrás de mim. Tendo o programa nos dois computadores tenho acesso à mesma informação desde que tenha ligação à internet e a *secret key* comigo. Uma dica para guardar a *secret key* pode ser enviá-la por email para nós mesmos e quando for preciso basta ir ao email buscá-la.

Apesar de utilizar uma versão deste programa para uso próprio não significa que este sistema seja infalível. Certamente o código poderia estar mais bem estruturado ou mais seguro, apenas desenvolvi para uso próprio e para mostrar as minhas competências. Se alguém conseguir meter as mãos no programa é possível que consiga aceder à base de dados utilizada, felizmente para nós os dados estão encriptados de tal forma que com a tecnologia existente atualmente seria impossível fazer um ataque de *brute force.* Levaria milhões de anos …

De qualquer das formas não me responsabilizo por qualquer perca ou danos de dados. O código é *open source* e é para ser utilizado por quem o bem entender.

Vale a pena também notificar que algo “completamente seguro” é impossível de atingir, o **foco da segurança é cada vez mais no processo em vez do objetivo**. Deste modo, segurança pode ser definida como os **passos necessários para proteger algo do perigo** e a nossa informação está protegida.