**Documentação do Código - Integração de transações Clara Cartões e Notion**

**Situação problema**

Este código automatiza a integração entre a **API da Clara Cartões** e o **Notion**. Ele busca transações financeiras recentes da Instituição de Cartão de Crédito Corporativo Clara Cartões, processa os dados recentes e cria entradas correspondentes em um banco de dados do Notion. O objetivo é facilitar a identificação e o controle de despesas, pois para cada transação é criado uma task no Notion para que o responsável preste contas ao financeiro evitando atrasos em conciliação e a falta de gestão das despesas.

**Objetivo**

Este código integra a API Clara Cartões com a API do Notion para:

* Obter transações recentes da Clara.
* Processar as transações.
* Criar novas entradas no Notion para as despesas que precisam ser identificadas.

**Configuração Inicial**

**1. Dependências Necessárias**

* **Bibliotecas Python:**
  + requests: Para realizar chamadas às APIs.
  + msvcrt: Para pausa da execução no Windows.
  + json: Para manipulação de arquivos JSON.
  + re: Para trabalhar com expressões regulares.
  + datetime: Para manipulação de datas e intervalos de tempo.
  + requests.auth.HTTPBasicAuth: Para autenticação básica na API Clara.

**2. Arquivos Necessários**

* **token-data.json:** Contém as credenciais para acesso à API Clara:
  + clientId
  + clientSecret
* **notion\_credenciais.json:** Contém as credenciais para acessar a API do Notion:
  + Clara\_Cartoes\_Key
  + Registro\_de\_Despesas\_ID
* **Certificados e Chaves da API Clara:**
  + Arquivo de certificado público: public-key.crt
  + Arquivo de chave privada: private-key.key
  + Certificado da autoridade certificadora: ca-public-key.pem
* **notion\_users.json:** Mapeamento de e-mails para IDs de usuários do Notion.

Exemplos de Arquivos de Configuração

token-data.json

{

"clientId": "seu\_client\_id\_aqui",

"clientSecret": "seu\_client\_secret\_aqui"

}

notion\_credenciais.json

{

"Clara\_Cartoes\_Key": "sua\_chave\_api\_notion\_aqui",

"Registro\_de\_Despesas\_ID": "id\_do\_banco\_de\_dados\_notion\_aqui"

}

notion\_users.json

{

"usuario1@empresa.com": "id\_usuario1\_notion",

"usuario2@empresa.com": "id\_usuario2\_notion"

}

**3. Configurações Adicionais**

O período de transações a ser processado é configurado automaticamente para os últimos 7 dias, excluindo o dia atual.

**Funções do Código**

**1. convert\_date\_format(date\_obj)**

Converte uma data para o formato YYYY-MM-DD.

**2. get\_access\_token()**

Obtém o token de acesso da API Clara usando autenticação básica.

* **Erro Tratado:** Qualquer erro ao obter o token exibe uma mensagem e finaliza o programa.

**3. get\_api\_data(access\_token)**

Faz uma chamada à API Clara para obter os dados das transações no período definido.

* **Parâmetros:**
  + access\_token: Token de acesso para autenticação.
* **Erro Tratado:** Qualquer falha na requisição exibe uma mensagem e finaliza o programa.

**4. retrieve\_all\_notion\_uuids()**

Busca todos os UUIDs existentes na base de dados do Notion para evitar duplicatas.

* **Iteração:** Realiza paginação para buscar todos os resultados.
* **Erro Tratado:** Qualquer falha na requisição exibe uma mensagem e finaliza o programa.

**5. create\_notion\_entry(transaction, existing\_uuids)**

Cria uma nova entrada no Notion com base nos dados da transação.

* **Parâmetros:**
  + transaction: Dados da transação da API Clara.
  + existing\_uuids: Lista de UUIDs existentes no Notion.
* **Critérios de Ignoração:**
  + Transações do tipo PAYMENT.
  + Transações já existentes no Notion.
  + Transações associadas a um e-mail específico.
* **Propriedades Criadas no Notion:**
  + Orientações
  + UUID
  + Título
  + Nome
  + Data de Pagamento
  + Tipo de Transação
  + Nome do Usuário
  + E-mail
  + Valor
  + Moeda
  + Responsável (se houver correspondência de e-mail).
* **Erro Tratado:** Qualquer falha na criação exibe uma mensagem e segue para a próxima transação.

**6. process\_and\_create\_notion\_entries(data, existing\_uuids)**

Processa todas as transações recebidas e tenta criar entradas no Notion para as que atendem aos critérios.

* **Parâmetros:**
  + data: Dados da API Clara.
  + existing\_uuids: Lista de UUIDs existentes no Notion.
* **Saída:** Exibe o total de entradas criadas.

**Fluxo Principal**

1. Obtém o token de acesso da API Clara.
2. Busca dados de transações recentes da API Clara.
3. Recupera todos os UUIDs já existentes no Notion.
4. Processa as transações e cria novas entradas no Notion para despesas identificáveis.
5. Exibe o total de novas entradas criadas.
6. Aguarda entrada do usuário para encerrar o programa.

**Mensagens de Erro e Tratamento**

* **Erro ao obter token:** Finaliza o programa após exibir mensagem.
* **Erro ao buscar dados da API Clara:** Finaliza o programa após exibir mensagem.
* **Erro ao buscar UUIDs do Notion:** Finaliza o programa após exibir mensagem.
* **Erro ao criar entrada no Notion:** Exibe mensagem e continua com as demais transações.

**Possíveis Extensões**

* Adicionar logs detalhados em vez de apenas mensagens no console.
* Implementar validações adicionais para os dados recebidos da API Clara.
* Incluir suporte para reprocessar entradas ignoradas.
* Criar interface para configurações dinâmicas de período.

**Observações**

* Certifique-se de que os arquivos JSON e certificados estão atualizados e no caminho correto.
* Revise o mapeamento de usuários para garantir a correta associação de responsabilidades no Notion.
* O sistema é projetado para evitar duplicatas no Notion, mas é importante monitorar manualmente as entradas criadas.