Documentação: Módulo de Scraping e Análise de Aulas

Este documento detalha o funcionamento dos componentes principais do projeto, que automatizam a coleta, preparação e registro de aulas.

Fluxo de Trabalho Geral

O processo completo é dividido em três fases principais, projetadas para serem executadas em sequência:

1. **Fase 1: Coleta (`scraper.py`)**: O robô acessa o portal da Seduc, faz login e extrai todos os registros de aulas **já existentes**. Isso cria uma base de dados atualizada do que já foi feito, salvando tudo em `data/aulas_coletadas.json`.
2. **Fase 2: Preparação (`preparar_planos.py`)**: Este script inteligente analisa os dados coletados, compara com o cronograma oficial e identifica qual a **próxima disciplina mensal** que precisa ser registrada para cada turma. Em seguida, ele cria arquivos de texto "esqueleto" (`.txt`) na pasta `aulas/`, um para cada aula faltante daquela disciplina.
3. **Fase 3: Registro (`registrar_aulas.py`)**: Após você preencher os arquivos `.txt` com o conteúdo e a estratégia da aula, este script assume o controle. Ele lê os arquivos preenchidos, acessa o portal e cadastra cada aula, uma por uma, de forma totalmente automática.

Adicionalmente, o `analise_aulas.ipynb` pode ser usado a qualquer momento para validar a coleta e entender o panorama geral dos registros.

---

1. `scraper.py` - O Coletor de Dados

Este script utiliza a biblioteca **Selenium** para automatizar um navegador Chrome, simulando as ações de um usuário para coletar dados de forma estruturada.

Funcionalidades Principais

- **Inicialização e Login**: Configura o navegador e realiza o login no portal usando as credenciais fornecidas.

- **Navegação Inteligente**: Seleciona o perfil de "Professor" e a instituição correta, lidando com a complexidade de `iframes` (páginas dentro de páginas).
- **Mapeamento de Turmas**: Utiliza arquivos JSON para traduzir os nomes "curtos" das turmas (ex: `1º DS`) para os nomes completos encontrados no portal (ex: `EMI-INT CT DES SIST-1ª SÉRIE -I-A`).
- **Coleta Cíclica**:
  i. Itera sobre cada turma e, dentro dela, sobre cada disciplina associada.
  ii. Clica em "Registro de aulas" para cada disciplina.
  iii. Acessa a tabela de aulas registradas.
- **Paginação**: Uma vez na tabela, o scraper navega por todas as páginas de resultados, clicando no botão "Próxima" até que não haja mais páginas, garantindo que **todos** os registros sejam coletados.
- **Tratamento de Erros**: Tira screenshots automaticamente em caso de falhas e possui mecanismos para tentar se recuperar de erros de navegação.

Estrutura de Arquivos Necessária

Para que o `scraper.py` funcione, os seguintes arquivos devem estar corretamente configurados no diretório `data/`:

- `config.json`: Contém o nome do professor para o qual os dados serão coletados.

```
{
  "professor": "Nome Sobrenome"
}
```

- `credentials.json`: Armazena o usuário e a senha de acesso ao portal.

```
{
  "username": "seu_usuario",
  "password": "sua_senha"
}
```

- `horarios_semanais_oficial.json`: Define quais turmas pertencem ao professor. O scraper lê as chaves dentro do objeto do professor (ex: "1º DS", "1º PJ").
- `mapa_turmas.json`: O "dicionário" que traduz o nome completo da turma (chave) para o nome curto (valor). Essencial para a navegação.

```
{
  "EMI-INT CT DES SIST-1ª SÉRIE -I-A": "1º DS",
  "ENS FUND II-9º ANO-I-B": "9º B"
}
```

- `turmas_com_disciplinas.json`: Mapeia o nome completo da turma às suas respectivas disciplinas.

Como Executar

1. Certifique-se de que todos os arquivos de configuração acima estão preenchidos.
2. Abra o terminal na raiz do projeto (`B:\Dev\Aulas_pygui>`).
3. Ative o ambiente virtual: `.venv\Scripts\activate`.
4. Execute o script:

```
python tools/scraper.py
```

5. Ao final, o arquivo `data/aulas_coletadas.json` será criado ou atualizado com os dados coletados.

---

2. Preparação e Registro de Aulas

Esta é a parte central do fluxo de automação, composta por dois scripts que trabalham em conjunto com a sua intervenção para planejar as aulas.

2.1. `preparar_planos.py` - O Assistente de Planejamento

Este script funciona como um assistente que prepara o terreno para você. Em vez de você ter que descobrir quais aulas faltam, ele faz isso automaticamente.

Como funciona?

1. Ele lê as aulas já coletadas (`aulas_coletadas.json`).
2. Para cada turma, ele verifica as disciplinas mensais e, com base na carga horária (ex: 40h), encontra a primeira que ainda não foi completada.
3. Ele então calcula as datas e horários das próximas aulas necessárias para completar essa disciplina, pulando fins de semana e feriados.

4. Por fim, ele cria arquivos `.txt` na pasta `aulas/`, organizados por turma. Ex: `aulas/1_DS/FUND_DB_20250901.txt`.

Como Executar:

```
# 1. Certifique-se de que o scraper.py foi executado recentemente.
# 2. Execute o preparador:
python tools/preparar_planos.py
```

2.2. Tutorial: Preenchendo os Planos de Aula

Após executar o `preparar_planos.py`, a pasta `aulas/` conterá os arquivos que você precisa editar. Esta é a sua principal tarefa manual.

Abra um dos arquivos gerados, por exemplo, `aulas/1_DS/FUND_DB_20250901.txt`. Você verá um modelo como este:

```
# Data: 01/09/2025
# Horário: 08:20 - 09:20

[CONTEUDO]
Preencher o conteúdo abordado aqui.

[ESTRATEGIA]
Preencher a estratégia metodológica aqui.
```

Sua tarefa é substituir os textos de preenchimento:

1. **Bloco `[CONTEUDO]`**: Descreva aqui o que será ensinado na aula. Você pode usar múltiplas linhas.
   - **Exemplo Preenchido:**

     ```
     [CONTEUDO]
     Introdução à Modelagem de Dados.
     - Conceitos de Entidade e Atributo.
     - Apresentação de Chaves Primárias.
     ```

2. **Bloco `[ESTRATEGIA]`**: Descreva como a aula será conduzida.
   - **Exemplo Preenchido:**

```
[ESTRATEGIA]
Aula expositiva com uso de slides, seguida por atividade prática de identificação
```

Salve o arquivo após preencher. Repita o processo para todos os arquivos gerados na pasta `aulas/`.

2.3. `registrar_aulas.py` - O Robô de Registro

Depois que você preencheu os planos de aula, este script faz o trabalho pesado.

Como funciona?

1. Ele varre a pasta `aulas/` em busca de todos os arquivos `.txt` que foram preenchidos.
2. Para cada arquivo, ele extrai os dados (data, horário, conteúdo, estratégia) e as informações da turma/disciplina.
3. Ele faz o login no portal, navega até a tela de registro da disciplina correta e preenche todos os campos do formulário, incluindo o anexo do link do material de apoio, se aplicável.
4. Se o registro for bem-sucedido, **o arquivo `.txt` correspondente é automaticamente excluído**. Se falhar, o arquivo é mantido para que você possa corrigir ou tentar novamente.

Como Executar:

```
# Execute o registrador após preencher os arquivos .txt:
python tools/registrar_aulas.py
```

2.4. Gerenciando Links de Recursos

Para evitar repetir links, o arquivo `data/recursos_links.json` centraliza os materiais de apoio. O `registrar_aulas.py` usa o nome curto da disciplina (ex: `FUND_DB`) para encontrar o link correspondente e anexá-lo na Aba 4 do formulário.

---

3. `analise_aulas.ipynb` - O Painel de Análise

Este é um Jupyter Notebook que serve como um dashboard interativo para explorar os dados coletados pelo scraper. Ele utiliza as bibliotecas **Pandas** para manipulação de dados e **Plotly** para visualização.

Análises Geradas

1. **Carregamento e Limpeza**:
   - Lê o arquivo `aulas_coletadas.json`.
   - Converte as colunas de data para um formato que permite análises temporais.
2. **Estatísticas Gerais**:
   - **Aulas por Disciplina**: Um gráfico de barras que mostra o total de aulas coletadas para cada disciplina. É a ferramenta mais importante para identificar falhas na coleta (ex: uma disciplina com muito menos aulas que as outras).
   - **Aulas por Turma**: Um gráfico de pizza que mostra a proporção de aulas por turma.
   - **Aulas ao Longo do Tempo**: Um gráfico de barras que agrupa as aulas por mês, ajudando a visualizar a distribuição ao longo do ano letivo.
3. **Análise Detalhada**:
   - **Tabela Cruzada e Mapa de Calor**: Mostra a quantidade exata de aulas coletadas para cada disciplina dentro de cada turma. É ideal para encontrar "buracos" específicos na coleta.

Como Utilizar

1. Certifique-se de que o `scraper.py` já foi executado e o arquivo `data/aulas_coletadas.json` existe.
2. No terminal, com o ambiente virtual ativado, instale as dependências para análise:

   ```
   pip install notebook pandas plotly nbformat
   ```

3. Inicie o servidor Jupyter:

   ```
   jupyter notebook
   ```

4. Seu navegador abrirá uma nova aba. Navegue até `tools/` e clique em `analise_aulas.ipynb`.
5. Dentro do notebook, você pode executar cada célula de código para gerar as tabelas e gráficos. Use o menu "Cell" > "Run All" para executar tudo de uma vez.

Investigando Problemas

Se a análise revelar que o número de aulas coletadas (ex: 704) é menor que o esperado (ex: 780), use os gráficos para responder às seguintes perguntas:

- **Qual disciplina tem menos aulas?** O gráfico de "Aulas por Disciplina" mostrará isso claramente.
- **A coleta falhou em um mês específico?** O gráfico de "Distribuição por Mês" pode indicar isso.

Com essas informações, você pode focar a investigação, por exemplo, rodando o `scraper.py` novamente para uma única disciplina e observando o console em busca de erros de paginação ou `Timeout`.