Nome: Jhonatan Barboza da Silva nºUSP: 15645049

MANUAL DO USUÁRIO

Este manual tem como objetivo orientar o usuário na utilização do programa *consulta_usp.py*, desde os passos iniciais de configuração do navegador e instalação das bibliotecas necessárias, até o funcionamento completo da aplicação.

Você encontrará instruções para preparar o ambiente, executar a coleta de dados acadêmicos da USP via web scraping e realizar consultas interativas com base nos dados coletados.

1. Configuração do navegador:

Para que o programa consulta_usp.py possa realizar a coleta de dados da web (web scraping), é necessário utilizar um navegador compatível com a automação via Selenium. Neste projeto, foi utilizado o navegador Google Chrome.

Além do navegador em si, também é necessário o ChromeDriver, que é o componente responsável por permitir que o Selenium controle o Chrome.

1.1 Requisitos

- Google Chrome instalado
- ChromeDriver da mesma versão do seu Chrome
- ChromeDriver colocado no diretório do projeto

1.2 Como verificar se está tudo certo

Verifique a versão do seu Google Chrome:

- Abra o Chrome e acesse: `chrome://settings/help`
- Anote o número da versão (por exemplo: `117.0.5938.92`)

Verifique se o ChromeDriver é compatível:

Execute no terminal: `./chromedriver --version`

O número exibido deve bater com a versão do seu Google Chrome. Se der erro ou não existir o comando, você precisa instalar o ChromeDriver.

1.3 Como instalar o ChromeDriver

Em sistemas baseados em Debian/Ubuntu:

```
sudo apt update
```

```
sudo apt install chromium-chromedriver
```

Obs: Nem sempre essa versão será exatamente compatível com o seu Chrome. Para total controle, veja o próximo método.

Manualmente (recomendado para garantir a versão correta):

1. Acesse: https://chromedriver.chromium.org/downloads
2. Baixe a versão correspondente ao seu Google Chrome.
3. Extraia o executável `chromedriver`.
4. Coloque o arquivo `chromedriver` na **mesma pasta do projeto**.

Se tiver algum problema com as configurações é conveniente utilizar uma IA para te auxiliar com a instalação.

2. Bibliotecas:

Para que o programa funcione corretamente, é necessário instalar duas bibliotecas principais:

- selenium
- beautifulsoup4

Você pode instalar essas bibliotecas no sistema global, mas é recomendado usar um ambiente virtual, para isolar dependências e evitar conflitos com outras aplicações Python.

2.1 Linux

Criando e ativando um ambiente virtual:

No terminal, dentro do diretório do projeto:

```
python3 -m venv venv
source venv/bin/activate
```

Ao ativar, você verá (venv) no início da linha do terminal.

Instalando as bibliotecas:

Com o ambiente ativado, execute:

```
pip install selenium beautifulsoup4
```

2.2 Windows

Criando e ativando um ambiente virtual:

No Prompt de Comando, dentro do diretório do projeto:

```
python -m venv venv
venv\Scripts\activate
```

Após ativar, o prompt mostrará (venv) indicando que o ambiente está ativo.

Instalando as bibliotecas:

Com o ambiente ativado, execute:

```
pip install selenium beautifulsoup4
```

3. Como utilizar o programa consulta_usp.py

O programa *consulta_usp.py* permite consultar dados de cursos e disciplinas da USP coletados previamente. Ele funciona com base no arquivo *dados_usp.pkl*, gerado automaticamente se ainda não existir.

Estrutura de arquivos:

Seu diretório do projeto deve conter:

- bot.py: responsável por coletar os dados e salvar no arquivo *dados_usp.pkl*
- class_USP.py: define as classes UnidadeUSP, Curso e Disciplina
- consulta_usp.py: interface de menu para o usuário interagir
- chromedriver → driver do navegador (compatível com sua versão do Chrome)
- dados_usp.pkl → gerado automaticamente após a primeira coleta dos dados ou quando solicitado.

3.1 Coleta de dados

Ao executar consulta_usp.py, o programa inicia o bot.py para coletar dados das unidades da USP via web scraping. O usuário pode definir quantas unidades deseja analisar; se deixar em branco ou inserir um valor acima do total, todas serão incluídas. Os dados coletados são salvos no arquivo dados_usp.pkl para consultas futuras, evitando nova coleta a cada execução, exceto quando solicitado manualmente.

3.2 Menu Principal de consulta

Após, carregar os dados solicitados, aparecerá um menu semelhante a este:

========== MENU DE CONSULTA ==========

0. Consulta detalhada de cursos
1. Listar cursos por unidade
2. Consulta por nome do curso
3. Exibir dados completos de uma unidade
4. Consulta por disciplina
5. Disciplinas que são usadas em mais de um curso
6. Recarregar dados da web'
7. Sair

==========

3.3 Explicando cada funcionalidade:

0. Consulta detalhada de cursos:
O usuário escolhe uma unidade.Depois, escolhe um curso dentro dessa unidade. E por fim, o programa exibe todas as informações do curso: nome, unidade, duração ideal/mínima/máxima e listas de disciplinas obrigatórias, optativas livres e eletivas.

1. Listar cursos por unidade
Mostra todos os cursos de uma unidade específica.

2. Consulta por nome do curso
O usuário digita o nome do curso, e todas as informações são exibidas. Se tiver mais de um curso com o nome digitado irão aparecer as opções para o usuário escolher.

3. Exibir dados completos de uma unidade
O usuário escolherá uma unidade e serão exibidos todos os dados dos cursos que pertencem a ela.

4. Consulta por disciplina
O usuário digita o nome de uma disciplina e serão exibidos todos os dados dela. Se houver mais de uma disciplina com o mesmo nome, aparecerá as opções para o usuário escolher.

5. Disciplinas utilizadas em mais de um curso
O usuário receberá todas as disciplinas que são utilizadas em mais de um curso das unidades que ele carregou da web.

6. Recarregar dados da web
Reexecuta o bot.py e atualiza o arquivo dados_usp.pkl com novas informações da web.

7. Sair
Encerra o programa.

Após selecionar uma opção e receber os dados correspondentes, o usuário retorna automaticamente ao menu principal. Nesse momento, ele pode:

- Escolher outra operação com os dados já carregados;

- Solicitar uma nova coleta de dados atualizados; ou

- Encerrar o programa.

Essa estrutura em loop garante uma experiência interativa até que o usuário decida encerrar o programa.

3.4 Executando o programa
No terminal, com o ambiente virtual ativado (Linux):

```
make
```

Ou diretamente:

```
python consulta_usp.py
```