ESPECIFICAÇÃO DO PROCESSO

Processo: Extração de dados dos filmes **Data**: 28/04/2022

Histórico de alterações:

Data	Versão	Descrição	Autor
28/04/2022	**1.0**	**Processo de automatização dados dos filmes**	**Willian Damasceno**

Descrição Geral

Objetivo:
Automatizar o fluxo de extração de dados dos filmes marcados como favoritos no site IMDb.

Detalhamento do Processo:

Macroprocesso:

1- Acessar a página de favoritos;
2- Fazer download do arquivo CSV com a lista de favoritos;
3- Fazer o WebScraping das páginas referente aos filmes;
4- Gerar um excel com os dados informados dos filmes;

Detalhes do Processo:

- Passo-a-Passo para a **Extração de vagas e envio:**
  - Acessar a página IMDb
  - Fazer login com as credenciais informadas
  - Obter a lista em CSV dos filmes marcados como favoritos
  - Tratar os dados obtidos e retorná-los
  - Transformar os dados retornados em um DataFrame (pandas)
  - Gerar um excel a partir deste Dataframe (Pandas)

ESPECIFICAÇÃO DO PROCESSO

Método responsável pelo acesso ao site e download do CSV.

```
class getCsvFile:

    def __init__(self, driver, url, login, pwd):
        self.driver = driver
        self.url = url
        self.login = login
        self.pwd = pwd

    def site(self):
        self.driver = webdriver.Chrome(self.driver)
        self.driver.get(self.url)

        a=1

    def login_site(self):
        self.driver.find_element_by_xpath("//div[@class='ipc-button__text'][normalize-space()='Fazer login']").click()
        self.driver.find_element_by_xpath("//span[normalize-space()='Sign in with IMDb']").click()
        self.driver.find_element_by_xpath("//input[@id='ap_email']").send_keys(self.login)
        self.driver.find_element_by_xpath("//input[@id='ap_password']").send_keys(self.pwd)
        self.driver.find_element_by_xpath("//input[@type='submit']").click()
        a=1

    def clearFolder(self,folder):
        files = glob.glob(folder)
        for f in files:
            os.remove(f)

    def download_favs(self):
        self.driver.find_element_by_xpath("//div[contains(text(),'Lista de favoritos')]").click()
        self.clearFolder(DOWNLOAD_FILE)
        self.driver.find_element(By.XPATH, "//a[normalize-space()='Export this list']").click()
        time.sleep(2)
        a=1
```

Classe responsável por gerar o data frame inicial e pegar os dados na página do filme

```
class dataExport:
    def __init__(self, csv):
        self.csv = csv
        self.df2 = None
        self.dataframe = pd.read_csv(csv)
        self.movie_url = []

    def get_url(self):
        self.df2 = self.dataframe.reset_index()
        for index, row in self.df2.iterrows():
            self.movie_url.append(row[7])

        return self.movie_url

    def generate_final_file(self):

        fav_movies_url = self.get_url()
        img = []
        actors = []
        title = []

        for movie_url in fav_movies_url:
            movie_data   = get_film_data(movie_url)
            img.append(movie_data[0])
            title.append(movie_data[1])
            actors.append(movie_data[2])

        self.dataframe['Portuguese Title'] = title
        self.dataframe['Actors'] = actors
        self.dataframe['URL Image']  = img


        pd.DataFrame((self.dataframe.rename(columns=COLS)), columns=COLS2).to_excel(FINAL_REPORT+r'\filmes.xlsx')
```

Método responsável pelo web scraping

```
def get_film_data(link):
    pagina = requests.get(link)
    conteudo = pagina.content
    soup = BeautifulSoup(conteudo, 'html5lib')
    image = soup.img.get_attribute_list('src')[0]
    titulo = soup.find("h1",{"data-testid":"hero-title-block__title"}).text

    art = [x.text for x in soup.select("li[data-testid='title-pc-principal-credit']:nth-child(3) > div > ul > li")]
    arts = " - ".join(list(set(art)))

    return image, titulo, arts
```

Requisitos

() Python instalado na máquina e instalação do requirements.txt

Declaro este documento de requisitos em acordo com escopo do projeto.

Aprovação:

Aprovador por:	Assinatura	Data