Avaliação Técnica

Desenvolvedor Python

Olá!

Seja bem vindo(a) ao nosso processo seletivo.

Como parte da sua avaliação, preparamos algumas tarefas que devem ser concluídas em até **7 dias** a partir de quando você recebê-las.

Assim que as concluir, envie seu projeto e/ou respostas para o Github e nos envie o link por e-mail para talentos@cotefacil.com, gabriel.gobbi@cotefacil.com e informe o recrutador(a).

Nos projetos, o repositório deve ter um README.md contendo instruções claras de como executar e uma breve visão técnica.

Também fará parte da sua avaliação: organização do repositório do projeto (branchs, commits, etc).

Em caso de dúvidas, nos contate através dos e-mails acima.

Boa sorte!

Desafio Técnico - Desenvolvedor Python (Web Scraping com Scrapy + Filas + API)

Para o cargo de Desenvolvedor Python na Cotefácil, você será desafiado a desenvolver uma aplicação que envolva web scraping usando Scrapy, processamento assíncrono com filas e integração com uma API de retorno de resultados.

Este desafio é dividido em três níveis de complexidade, permitindo que você demonstre suas habilidades em Python, Scrapy e arquitetura de software.

Objetivo

Este desafio tem como objetivo avaliar sua capacidade de:

- Trabalhar com Scrapy para realizar web scraping autenticado.
- Processar tarefas assíncronas por meio de filas.
- Integrar o fluxo de scraping e pedidos com uma API de retorno de resultados.
- Projetar código limpo, modular e manutenível.

Você deverá desenvolver uma aplicação que acesse o site do fornecedor Servimed, autentique-se com usuário e senha, busque todos os produtos disponíveis e realize um pedido, retornando seu código de confirmação.

- Site: https://pedidoeletronico.servimed.com.br/
- Usuário: juliano@farmaprevonline.com.br
- Senha: a007299A

Estrutura do desafio

A aplicação será dividida em 3 etapas, com níveis de complexidade crescentes.

Nível 1 - Básico

Objetivo:

Criar um projeto Scrapy que:

- Faça login no site;
- Acesse a listagem de produtos;
- Extraia as seguintes informações de cada produto:
    - GTIN (EAN);
    - Código;
    - Descrição;
    - Preço de fábrica;
    - Estoque.

Requisitos:

- Utilizar Scrapy (sem Selenium, Playwright, etc);
- Armazenar os dados extraídos em JSON local.

Entregáveis:

- Projeto Scrapy funcional;
- Script de execução com parâmetros de login.

Nível 2 - Intermediário

Objetivo:

Criar um processo assíncrono que:

- Receba tarefas de scraping via fila (por exemplo, Redis ou RabbitMQ).
- Execute o scraping de forma independente para cada solicitação.
- Envie os dados de produtos encontrados para a API de callback;
    - É necessário autenticar (`/oauth/token`) para enviar os dados para a API;
    - Crie um usuário na API em `/oauth/signup`.

Requisitos:

- Uso de filas (ex: Celery + Redis, ou RQ, ou outra ferramenta similar).
- As tarefas devem ser enfileiradas com os seguintes dados:

```
{
  "usuario": "fornecedor_user",
  "senha": "fornecedor_pass",
  "callback_url": "https://desafio.cotefacil.net",
}
```

- O worker deve processar essa tarefa e fazer POST para `/produto` com os dados extraídos;
    - A requisição do callback deve ser um JSON com os produtos encontrados:

```
[
  {
    "gtin": "1234567890123",
    "codigo": "A123",
    "descricao": "Produto A",
    "preco_fabrica": 10.99,
    "estoque": 100
  },
  {
    "gtin": "9876543210987",
    "codigo": "B456",
    "descricao": "Produto B",
    "preco_fabrica": 20.49,
    "estoque": 50
  }
]
```

Entregáveis:

- Código do worker + fila.
- Exemplo de chamada à fila.
- Documentação de como testar localmente.

Nível 3 - Avançado

Objetivo:

Além da extração, realizar um pedido de compra de um produto específico;

- Após o login, buscar o produto pelo código;
- Realizar um pedido via formulário no site;
- Retornar o código do pedido para a API de callback;
    - É necessário autenticar (`/oauth/token`) para enviar os dados para a API;
    - Crie um usuário na API em `/oauth/signup`.

Requisitos:

- Estender a estrutura atual para realizar a compra (simulação de envio de formulário);
- Chamar a API do desafio para gerar um pedido aleatório;
- As tarefas devem ser enfileiradas com os seguintes dados:

```
{
  "usuario": "fornecedor_user",
  "senha": "fornecedor_pass",
  "id_pedido": "1234",
  "produtos": [
    {
      "gtin": "1234567890123",
      "codigo": "A123",
      "quantidade": 1,
    }
  ],
  "callback_url": "https://desafio.cotefacil.net"
}
```

- O worker deve processar essa tarefa e fazer PATCH para `/pedido/:id` com os dados extraídos;
    - A requisição do callback deve ser um JSON com os dados de confirmação do pedido:

```
{
  "codigo_confirmacao": "ABC987",
  "status": "pedido_realizado"
}
```

Entregáveis:

- Lógica de pedido integrada.
- Simulação de ambiente (pode ser site de testes local ou documentação com prints/mock).
- Testes automatizados (preferencialmente com pytest ou similar).

Regras Gerais

- Pode usar frameworks auxiliares (ex: FastAPI, Celery, Typer) desde que a lógica principal esteja em Python;
- O projeto deve conter um README com instruções de instalação e execução;
- Valorizamos:
    - Boas práticas de código;
    - Modularização;
    - Logs e tratamento de erros;
    - Código testável e documentado.

Dicas

- Veja a documentação em OpenAPI da API de callback em https://desafio.cotefacil.net/docs.
- Utilize ferramentas como Postman ou Insomnia para testar as APIs.
- Consulte a documentação do Scrapy para entender como lidar com autenticação e extração de dados.