



Universidade do Minho
Escola de Ciências

Trabalho Prático

CLI para consulta da API itjobs.pt

**Ambientes e Linguagens de Programação
para Ciência de Dados**

TRABALHO REALIZADO POR:
GABRIELA SOARES - A112282
LARA SACRAMENTO - A110333
RODRIGO ALVES - A111408

Contexto do Projeto

Objetivo: criar uma ferramenta capaz de consultar, analisar, enriquecer e exportar dados reais sobre ofertas de emprego;

Foco principal:

- TP1 → Acesso estruturado à API do itjobs.pt;
- TP2 → Enriquecimento da informação via Web Scraping (Teamlyzer);

Arquitetura modular, baseada em Typer + Requests + Regular Expressions + BeautifulSoup;

Os Desafios

- Como recolher informação fiável, atual e estruturada sobre empregos em TI?
- Como integrar diferentes fontes de dados (API + Website)?
- Como extrair padrões relevantes em textos (regime, skills)?
- Como enriquecer ofertas com informação sobre empresas?
- Como organizar tudo numa CLI simples, robusta, validada e exportável?

Autenticação Necessária para Aceder à API itjobs.pt

- O itjobs.pt exige uma API Key, fornecida após registo por email;
- A chave deve ser incluída em todos os pedidos, caso contrário a API devolve erro;
- Este mecanismo garante:
 1. controlo de acesso;
 2. prevenção de abuso;
 3. consistência dos dados;

```
API_KEY = "d160d8c93e8e49486873b9f6f60d3822"
```

Funcionamento do Acesso à API

A aplicação comunica com a API via HTTP GET/POST, usando requests;

Cada pedido segue o mesmo fluxo:

1. Construção da rota + parâmetros;
2. Envio da API Key;
3. Validação do código de resposta;
4. Conversão para JSON;
5. Tratamento e filtragem;

Rotas principais (TPI)

- /api/jobs → listar anúncios recentes
- /api/jobs/{id} → detalhe de um trabalho
- /api/jobs/search → filtro por empresa/localidade

Campos relevantes

- title, company, locations, type, publishedAt
- body (para análise textual)
- allowRemote (regime de trabalho)

A aplicação agrupa estes dados localmente, permitindo filtragem, análise e exportação.

Agrupamento de Requests: Rotas e Campos

Justificação Técnica

O código foi estruturado assim porque:

- Typer → cria comandos organizados, claros e escaláveis;
- Requests → comunicação HTTP simples e robusta;
- Regex (re) → extrair padrões não diretamente fornecidos pela API;
- BeautifulSoup → recolher informação no TP2 não disponível por API;
- CSV/JSON → formatação, persistência e reutilização dos dados;
- Estrutura modular → facilita manutenção e evolução do TP1 para o TP2;

Vista Geral

TP1

- Lista as N ofertas mais recentes;
- Filtra ofertas part-time por empresa/localidade;
- Deduz regime remoto/híbrido/presencial;
- Conta skills entre duas datas;
- Exporta dados para CSV;

TP2

- Enriquecimento das ofertas com dados do Teamlyzer;
- Estatísticas por zona e tipo de trabalho;
- Top skills por profissão (scrapping Teamlyzer);
- Exportações CSV adicionais;

Caminho que o Programa Percorre

- 1.O utilizador chama um comando da CLI;
- 2.Typer valida argumentos;
- 3.É feita comunicação com a API ou com o Teamlyzer (scraping);
- 4.O programa processa JSON ou HTML;
- 5.Aplica filtros, regex, agrupamentos;
- 6.Formatada saída em JSON;
- 7.Exporta opcionalmente para CSV;

Top N mais recentes

```
url = "https://www.itjobs.pt/api/jobs"  
jobs = requests.get(url, params={"api_key": API_KEY}).json()
```

TPI: Endpoints Utilizados

Regime de trabalho

```
url = f"https://www.itjobs.pt/api/jobs/{job_id}"  
job = requests.get(url, params={"api_key": API_KEY}).json()
```

Pesquisa Part-time

```
url = "https://www.itjobs.pt/api/jobs/search"  
res = requests.get(url, params={  
    "company": empresa,  
    "location": localidade,  
    "api_key": API_KEY  
}).json()
```

TP2: HTML Utilizada no Scraping

Fonte: Teamlyzer

- Ranking de empresas
- Página da empresa
- Página de ofertas filtradas por skills

Campos extraídos:

- Rating
- Descrição
- Benefícios
- Salários médios

```
html = requests.get(url, headers=headers).text  
soup = BeautifulSoup(html, "html.parser")
```

```
rating = soup.find("div", class_="rating").text  
benefits = [b.text for b in soup.select(".benefits li")]
```

Expressões Regulares (Regime + Skills)

Deteção do regime de trabalho (TP1)

```
if re.search(r"\b(remote|teletrabalho)\b", texto, re.I):  
    return "Remoto"  
if re.search(r"\b(híbrido)\b", texto, re.I):  
    return "Híbrido"
```

Contagem de skills (TP1/TP2)

```
skills = re.findall(r"\b(python|java|sql|docker)\b", texto, re.I)
```

Paginação no TP2 (Scraping)

Percorrer páginas do Teamlyzer

```
page = 1  
skills = []
```

```
while True:
```

```
    url = f"{base_url}?page={page}"  
    soup = BeautifulSoup(requests.get(url,  
headers=headers).text, "html.parser")
```

```
    entries = soup.find_all("div", class_="job-entry")  
    if not entries:  
        break
```

```
    skills.extend(extrair(entries))  
    page += 1
```

Dificuldades Encontradas

API itjobs.pt:

- Falta de uniformidade nos campos;
- Problemas de filtragem nativa (empresa/localidade);
- Dependência da API Key;

Scraping Teamlyzer:

- HTML sem estrutura oficial;
- Elementos escondidos ou dinâmicos;
- Mudança de classes;
- Necessidade de validar resultados nulos;

Regex e Análise Textual:

- Textos inconsistentes entre anúncios;
- Casos ambíguos no regime de trabalho;

Validações:

- Datas mal formatadas;
- job_id inexistentes;



Reflexão Final

- Este projeto permitiu integrar API + Web Scraping numa solução única;
- TP1 criou a base de aquisição e análise textual;
- TP2 expandiu para enriquecimento e estatísticas avançadas;
- A aplicação tornou-se modular, robusta e extensível;



Reforçámos competências em:

- consumo de serviços web;
- processamento de JSON e HTML;
- expressões regulares;
- construção de CLIs;



O resultado final é uma ferramenta que transforma dados dispersos em informação útil, permitindo analisar o mercado de trabalho de forma mais completa;