Universidade do Minho
Licenciatura em Ciência de Dados

**Unidade Curricular: Ambientes e Linguagens de Programação para Ciência de Dados**

---

Aplicação de conhecimentos:

# Web Scraping

**Realizado por:**

Mariana Costa Pereira, A110552

Joana Rafaela Gomes da Rocha, A110386

Rui Duarte de Barros Braga Simões, A111869

# Índice

# Introdução

O Trabalho Prático nº2 tem como objetivo complementar o projeto iniciado no Trabalho prático nº1, adicionando novas funcionalidades através de interação com a API do *itjobs.pt* e de Web Scraping ao website *Teamlyzer*.

O script desenvolvido (*emprego.py*) funciona como uma CLI que permite pesquisar ofertas de emprego, obter detalhes de um anúncio específico, recolher informação adicional sobre a empresa, gerar estatísticas e identificar skills associadas a diferentes posições.

Para isso, foram utilizados módulos como **requests**, **BeautifulSoup**, **re** e **csv**, permitindo recolher, processar e exportar dados de forma automática. Este relatório descreve a solução implementada e o funcionamento das principais funcionalidades.

# Tecnologias utilizadas

Para o desenvolvimento do TP2 foram utilizadas tecnologias semelhantes às do TP1, mas ampliadas para suportar novas funcionalidades como *Web Scraping* e enriquecimento de dados. A aplicação continua a ser desenvolvida em **Python 3**, tirando partido da sua simplicidade e do vasto ecossistema de bibliotecas dedicadas à extração, análise e tratamento de informação.

A biblioteca **Requests** voltou a ser utilizada para comunicar com a API do *itjobs.pt*, enquanto os módulos **json** e **csv** permitiram processar as respostas e exportar dados estruturados. Como novidade neste trabalho, recorreu-se à biblioteca **BeautifulSoup** para realizar *Web Scraping* no website *Teamlyzer*, permitindo extrair informações adicionais sobre empresas, como rating, descrição ou benefícios. O módulo **re** manteve um papel importante na normalização de texto e identificação de padrões necessários à extração de skills e classificação do regime de trabalho.

Tal como no projeto anterior, o desenvolvimento foi realizado no **Visual Studio Code**, com integração direta com Git. O **GitHub** foi novamente utilizado como plataforma de controlo de versões, assegurando a partilha, sincronização e histórico do projeto ao longo das diferentes etapas de implementação e teste.

# Funcionalidades desenvolvidas

Neste trabalho foram implementadas várias funcionalidades que expandem o projeto do TP1, permitindo não só consultar informação da API do *itjobs.pt*, mas também enriquecê-la com dados recolhidos através de *Web Scraping* ao *Teamlyzer*. Todas as funcionalidades foram integradas numa CLI intuitiva, capaz de produzir resultados em formato JSON ou CSV conforme pretendido pelo utilizador.

A **primeira funcionalidade** corresponde à obtenção detalhada de um anúncio específico, implementada na função **get_job(job_id)**. Para além de consultar o **endpoint *job/get.json***, esta função identifica automaticamente a empresa associada ao anúncio e procura a respetiva página no *Teamlyzer*. Quando encontrada, o programa recolhe informação adicional, como o **rating geral, a descrição, os principais benefícios e referências salariais**. Estes dados são incorporados no objeto JSON final apresentado ao utilizador. Tal como no TP1, esta funcionalidade inclui ainda a possibilidade de exportação para CSV.

A **segunda funcionalidade** introduzida foi a geração de estatísticas sobre vagas por zona e tipo de trabalho, implementada na função **statistics_zone()**. A aplicação recolhe uma lista de anúncios recentes e contabiliza o número de vagas por localização e título da posição. Os resultados são automaticamente exportados para um ficheiro CSV, permitindo ao utilizador analisar de forma clara a distribuição geográfica das ofertas de emprego.

A **terceira funcionalidade**, desenvolvida na função **list_skills(job_title)**, permite identificar as principais skills associadas a uma determinada área profissional. O programa acede ao *Teamlyzer* e analisa o texto das ofertas correspondentes ao título fornecido, contabilizando a frequência de diversas tecnologias, linguagens de programação e ferramentas mencionadas. As dez skills mais frequentes são apresentadas em formato JSON e podem também ser exportadas para um ficheiro CSV.

Por fim, a **quarta funcionalidade** corresponde ao mecanismo de exportação para CSV, utilizado em várias partes da aplicação. A função **export_jobs_to_csv** recebe uma lista de anúncios e grava num ficheiro os campos mais relevantes, garantindo previamente a limpeza do texto das descrições através da função **clean_html**, que remove tags HTML. Este módulo assegura que os dados produzidos podem ser reutilizados ou analisados com facilidade noutras ferramentas.

No seu conjunto, estas funcionalidades permitem recolher, enriquecer, analisar e armazenar informação de forma simples e automatizada, oferecendo ao utilizador uma ferramenta completa para explorar e estudar o mercado de trabalho na área tecnológica.

# Explicação do funcionamento

---

O programa desenvolvido funciona como uma **Command Line Interface (CLI)** que permite ao utilizador executar diferentes operações através de comandos específicos. Cada comando está associado a uma função no ficheiro *emprego.py*, sendo responsável por comunicar com a API do *itjobs.pt*, realizar *Web Scraping* no *Teamlyzer* ou processar informação recolhida.

Quando o utilizador executa o programa através do terminal, o argumento introduzido após *python emprego.py* determina qual das funcionalidades será acionada. O ficheiro contém uma estrutura condicional que identifica o comando solicitado e encaminha a execução para a função correspondente. Todos os comandos aceitam parâmetros adicionais, necessários para especificar filtros, limites ou caminhos de ficheiros CSV.

A comunicação com a API é realizada através da biblioteca **Requests**, enviando pedidos HTTP aos *endpoints* disponibilizados pelo *itjobs.pt*. As respostas são devolvidas em formato JSON e processadas pelo programa de forma a extrair apenas a informação relevante. Nos casos em que é necessário complementar os dados com informações externas, como acontece no comando *get*, o

programa acede ao website *Teamlyzer* utilizando **BeautifulSoup** e extrai automaticamente elementos presentes no HTML, como ratings, descrições ou benefícios.

O processo de filtragem, análise de texto e deteção de padrões é realizado com o auxílio do módulo **re**, permitindo identificar termos associados a regimes de trabalho ou contabilizar a ocorrência de determinadas tecnologias. Sempre que o utilizador solicita a exportação para CSV, a função **export_jobs_to_csv** organiza os campos relevantes e grava os dados num ficheiro estruturado, garantindo que a informação é limpa e consistente.

De forma geral, o funcionamento da aplicação assenta na combinação de três pilares:

1. **Recolha de dados** através da API e de *Web Scraping.*
2. **Processamento e análise** da informação obtida.
3. **Apresentação ou exportação** dos resultados em formatos legíveis.

Esta abordagem modular torna o programa fácil de utilizar, manter e expandir, permitindo que cada função execute uma tarefa bem definida dentro do fluxo global da aplicação.

# Resultados

---

Após a implementação das funcionalidades, o **programa foi testado com diversos cenários** para verificar o seu correto funcionamento. Os testes mostraram que a aplicação responde adequadamente a cada um dos comandos implementados.

**a. Informação detalhada de um jobID**

Para validar a funcionalidade associada ao comando **get**, foi executado o comando *python emprego.py get 506697*, cujo propósito é apresentar todos os dados referentes a um anúncio específico disponibilizado pela API do *itjobs.pt* e enriquecê-los com informação adicional recolhida através de Web Scraping ao *Teamlyzer*. A figura 1 mostra parte da resposta obtida, exibida em formato JSON estruturado.

A aplicação começou por estabelecer comunicação com o ***endpoint*** **job/get.json**, obtendo corretamente todos os campos fornecidos pela API, como o **título da oferta**, a **empresa** responsável, a **descrição** da vaga, a **localização**, o **tipo de contrato** e a **data de publicação**. Estes campos surgem integralmente no resultado apresentado ao utilizador, preservando a totalidade das informações disponibilizadas pelo portal.

Em seguida, a **aplicação identificou** automaticamente **a empresa associada ao anúncio** (**We Are Meta**) e **procurou a respetiva página no *Teamlyzer***. Como a empresa foi encontrada, foram extraídos elementos adicionais, nomeadamente:

- o **rating geral** da empresa,
- a **descrição** apresentada na plataforma,
- os **benefícios** destacados,
- e uma referência relativa ao **salário**.

Estes campos foram integrados no JSON final devolvido, demonstrando que o processo de Web Scraping está a funcionar corretamente e que as informações provenientes das duas fontes foram combinadas com sucesso.

```
PS C:\Users\Joana\Desktop\TP1_ambientes> python emprego.py get 506697
>>
C:\Users\Joana\Desktop\TP1_ambientes\emprego.py:371: DeprecationWarning: The 'text' argument to find()-type methods is deprecated. Use 'string' instead.
  for tag in soup.find_all(text=re.compile(r'\d+[.,]\d+')):
C:\Users\Joana\Desktop\TP1_ambientes\emprego.py:401: DeprecationWarning: The 'text' argument to find()-type methods is deprecated. Use 'string' instead.
  for tag in soup.find_all(text=re.compile(r'(salário|salary|€|\$)', re.IGNORECASE)):
{
  "id": 506697,
  "company": {
    "id": 6433,
    "name": "We Are Meta",
    "logo": "https://static.itjobs.pt/images/companies/db/ec0/6433/logo.png",
    "description": "We Are META was born to prove that Tech Talent Consultancy can be great for IT careers.\r\n \r\nWe empower IT businesses by creating a CRASH: a grou
p of highly skilled and motivated tech professionals that extends our client's team and charges at every obstacle, fearlessly! Recruiting amazing tech talent demands br
ight minds and honest people, and at META, we have both!\r\n \r\nJoin the CRASH and become a Rhino!",
    "address": "Av. Almirante Reis, 54, 6th floor\r\n1150-019, Lisboa",
    "phone": "+351 211 601 380",
    "email": "sayhello@wearemeta.io",
    "url": "https://wearemeta.io",
    "url_linkedin": "https://www.linkedin.com/company/we-are-meta",
    "slug": "we-are-meta"
  },
  "companyId": 6433,
  "title": "PHP Symfony Senior Developer (French speaker)",
  "body": "<p><strong>Seeking a PHP </strong><strong> Symfony Senior Developer (French speaker): Rhino, are you there? </strong></p><p style=\"text-align:center;\">At W
E ARE META, we focus on finding the perfect match between our Rhinos and our clients.</p><p><strong><em>Why join us? </em></strong></p><ol><li><strong>Expand your netwo
rk: </strong> As a <strong> consultant</strong> at We Are META, you'll have access to a network of national and international partners across <strong> diverse</strong>
 sectors of the <strong> tech</strong> industry.</li><li><strong>Enjoy our perks: </strong> When you join the <strong> crash</strong> you'll have an array of <strong> be
nefits</strong>, to help you achieve the best work-life balance.</li><li><strong>Get your perfect match: </strong> Our <strong> recruiters</strong> do their best to get
 the best position for you, whether in terms of <strong> working policy</strong>(remote, hybrid, on-site) or the company's <strong> culture</strong>.</li><li><strong>Pe
rsonalized support: </strong> With our <strong> career coach</strong>, we assure that all your needs are being met and you're being provided everything you need to exce
l in your professional life because the <strong> well-being</strong> of our Rhinos is our number one priority.</li></ol><p><strong><em>Perks of becoming a Rhino: </em><
/strong></p><ul><li>You'll get a welcome kit </li><li>Opportunities for career progression </li><li>Live on the edge with our health insurance </li><li>Coverflex meal c
ard </li><li>Other protocols and special discounts</li></ul><p><strong><em>For this position, the perfect Rhino should have the following skills: </em></strong></p><ul>
  "allowRemote": false,
  "wage": null,
  "types": [
    {
      "id": "1",
      "name": "Full-time"
    }
  ],
  "contracts": [
    {
      "id": "1",
      "name": "Fixed term"
    },
    {
      "id": "2",
      "name": "Permanent"
    },
    {
      "id": "4",
      "name": "Freelance"
    }
  ],
  "locations": [
    {
      "id": "18",
      "name": "Porto"
    }
  ],
  "publishedAt": "2025-11-09 17:21:48",
  "updatedAt": "2025-11-09 17:21:48",
  "slug": "php-symfony-senior-developer-french-speaker",
  "teamlyzer_rating": 4.5,
  "teamlyzer_description": "Reviews e opiniões da Swiss Post - Lisbon IT Campus. Conhece o processo de recrutamento, vê salários e obtêm feedback de outros informáticos
| Teamlyzer",
  "teamlyzer_benefits": "EXPLORAR; Nova review; Procurar reviews; Prémios 2025; Ranking de empresas",
  "teamlyzer_salary": "Calculadora de salários TI"
}
```


**Figura 1** - Resposta JSON obtida após a execução do comando *python emprego.py get 506697*, contendo os dados do anúncio e a informação adicional recolhida no Teamlyzer.

### b. Contagem de vagas por tipo/nome da posição e por região.

Para validar a funcionalidade **statistics zone**, foi executado o comando: **python emprego.py statistics zone**

Após a execução, o programa estabeleceu ligação ao endpoint *job/list.json*, processou os anúncios devolvidos e contabilizou o número de vagas por localização e por título da posição. No terminal, como é possível verificar na figura 2, foi apresentada a mensagem **"Ficheiro de exportação criado com sucesso."**, indicando que o processamento decorreu sem erros e que o ficheiro CSV foi gerado corretamente.

```
PS C:\Users\Joana\Desktop\TP1_ambientes> python emprego.py statistics zone
>>
Ficheiro de exportação criado com sucesso.
```

**Figura 2** - Output obtido no terminal após executar o comando *python emprego.py statistics zone*.

O ficheiro criado, *statistics_zone.csv*, contém três colunas — **Zona**, **Tipo de Trabalho** e **Número de Vagas** — organizando todas as combinações encontradas nos anúncios. A figura apresentada mostra apenas o início do ficheiro, onde surgem algumas das primeiras cidades processadas (como Aveiro e Braga). No entanto, o CSV completo inclui várias outras zonas do país, abrangendo todas as localizações presentes nos anúncios recolhidos.

Entre as primeiras entradas do ficheiro observam-se, por exemplo:

- Aveiro — *Generative AI Engineer* — 1 vaga
- Aveiro — *System Administrator Linux* — 1 vaga
- Braga — *Android Developer* — 1 vaga
- Braga — *Customer Support Consultant – VoIP* — 1 vaga

A análise confirma que a funcionalidade cumpre integralmente o seu objetivo: percorre todos os anúncios devolvidos pela API, identifica as localizações associadas e gera automaticamente um ficheiro CSV que resume a distribuição geográfica das ofertas de emprego, incluindo todas as cidades presentes nos dados originais

emprego.py | statistics_zone.csv U

statistics_zone.csv > data

```
1   Zona,Tipo de Trabalho,Nº de vagas
2   Aveiro,Generative AI Engineer,1
3   Aveiro,Project Manager (CRM Dynamics),1
4   Aveiro,Python Engineer,1
5   Aveiro,Senior IT Engineer,1
6   Aveiro,Software Engineer (Python + React),1
7   Aveiro,System Administrator Linux,1
8   Aveiro,Technical Team Lead,1
9   Aveiro,Web Developer,1
10  Aveiro,Web Developer (Angular),1
11  Braga,AI Engineer,1
12  Braga,Android Developer,1
13  Braga,Backend Developer (NodeJS/.NET),1
14  Braga,Cloud  Architect,1
15  Braga,Customer Support Consultant - VoIP,1
```

**Figura 2** – Resultado do comando python emprego.py statistics zone e excerto inicial do ficheiro *statistics_zone.csv*.

```
Lisboa,Commercial Intelligence,1
Lisboa,Consultor SRE,1
Lisboa,Data Analyst,2
Lisboa,Data Engineer,4
Lisboa,Data Engineer (AWS),1
Lisboa,Data Engineer (English speaker),1
Lisboa,Data Engineer- Azure,1
```

**Figura 3** - Outro excerto do csv do ficheiro *statistics_zone.csv*.

### c. Identificação das principais skills associadas a um cargo

Para validar o funcionamento da funcionalidade **list skills**, foi executado o comando: **python emprego.py list skills "data scientist"**

O objetivo deste comando é identificar, através de Web Scraping ao *Teamlyzer*, as tecnologias mais frequentes nas ofertas relacionadas com o cargo indicado. Após aceder às páginas associadas ao termo "data scientist", o programa extraiu o texto relevante e aplicou um processo de contagem baseado em expressões regulares.

```
PS C:\Users\Joana\Desktop\TP1_ambientes> python emprego.py list skills "data scientist"
[
  {
    "skill": "ruby",
    "count": 2
  },
  {
    "skill": "ai",
    "count": 2
  },
  {
    "skill": "postgresql",
    "count": 1
  },
  {
    "skill": "elasticsearch",
    "count": 1
  },
  {
    "skill": "github",
    "count": 1
  },
  {
    "skill": "react",
    "count": 1
  },
  {
    "skill": "python",
    "count": 1
  },
  {
    "skill": "machine learning",
    "count": 1
  },
  {
    "skill": "redis",
    "count": 1
  },
  {
    "skill": "pytorch",
    "count": 1
  }
]
```

**Figura 4 -** Resultado do comando python emprego.py list skills "data scientist", apresentando as skills mais mencionadas e respetiva contagem.

A figura 4 apresenta o conjunto de skills devolvido pela aplicação em formato JSON. Para o cargo analisado, as tecnologias mais mencionadas foram:

- **ruby** – 2 ocorrências
- **ai** – 2 ocorrências
- **postgresql** – 1 ocorrência
- **elasticsearch** – 1 ocorrência
- **github** – 1 ocorrência
- **react** – 1 ocorrência
- **python** – 1 ocorrência
- **machine learning** – 1 ocorrência
- **redis** – 1 ocorrência

Estes resultados mostram que as menções são variadas e abrangem não apenas linguagens de programação, mas também ferramentas de versionamento, frameworks e tecnologias de bases de dados. Apesar de o cargo estar associado tradicionalmente a ferramentas como Python e machine learning, o scraping revelou a presença de outras tecnologias relevantes no conjunto de ofertas analisadas.

O teste confirmou que a funcionalidade opera corretamente, tal como nas restantes funcionalidades, é ainda possível exportar estes resultados para CSV mediante indicação de um nome de ficheiro adicional.

### d. Exportação para CSV

A funcionalidade de exportação para ficheiros CSV foi validada através dos comandos **get** e **list skills**, que permitem gerar automaticamente um ficheiro sempre que o utilizador indica um nome de saída.

Quando foi executado o comando **python emprego.py get 506697 resultado.csv**, a aplicação apresentou no terminal o objeto JSON com toda a informação recolhida da API e enriquecida através do Teamlyzer, e gerou simultaneamente o ficheiro *resultado.csv*.


**Figura 5** - Ficheiro *resultado.csv* gerado pelo comando **python emprego.py get 506697 resultado.csv,** contendo a informação do anúncio enriquecida com dados do Teamlyzer.

Este ficheiro, ilustrado na **Figura 5**, reúne os dados essenciais do anúncio, como o identificador, o título, a empresa, a localização e a data de publicação, bem como os campos resultantes do Web Scraping, nomeadamente o rating da empresa, os benefícios identificados e a descrição associada. A análise da figura confirma que os dados são organizados de forma clara, permitindo uma leitura imediata dos elementos mais relevantes.

De forma semelhante, a execução do comando **python emprego.py list skills "data scientist" skills.csv** originou o ficheiro *skills.csv*, apresentado na **Figura 6**. Este ficheiro contém a lista de skills identificadas para o cargo pesquisado, acompanhadas do número de ocorrências de cada uma. Tal como evidenciado na figura, o ficheiro reflete fielmente os resultados devolvidos no terminal, demonstrando que o processo de exportação é consistente e corresponde exatamente aos valores calculados pela aplicação.

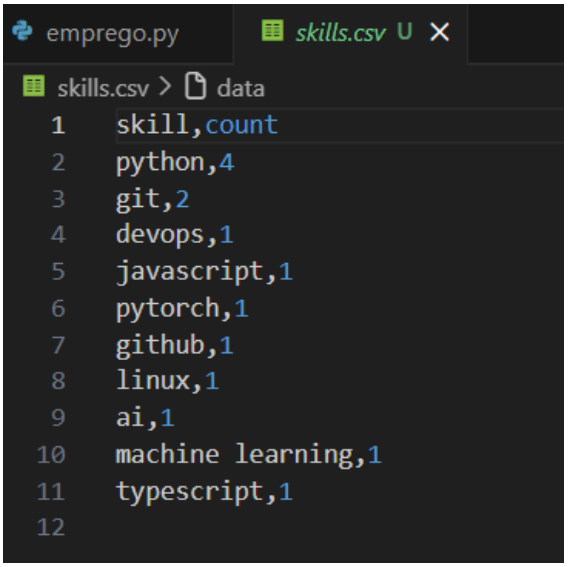


**Figura 6** - Ficheiro *skills.csv* gerado pelo comando **python emprego.py list skills "data scientist" skills.csv**, apresentando as skills identificadas e o número de ocorrências.

Antes de guardar os dados, o programa aplica ainda um conjunto de tratamentos, nomeadamente a remoção de tags HTML presentes nas descrições dos anúncios e a substituição automática de valores em falta por "não especificado". Este pré-processamento garante que os ficheiros finais são legíveis, limpos e prontos para utilização em ferramentas externas.

Os testes realizados permitiram confirmar que a exportação para CSV funciona de forma estável em todos os cenários avaliados, produzindo os ficheiros corretamente mesmo quando o número de resultados é reduzido. Conclui-se, assim, que esta funcionalidade cumpre integralmente os requisitos definidos e contribui significativamente para a utilidade prática da aplicação.

# Conclusão

O trabalho permitiu integrar dados obtidos através da API do *itjobs.pt* com informação adicional recolhida por *Web Scraping* ao Teamlyzer, reforçando competências de recolha, processamento e análise automática de dados.

A aplicação desenvolvida cumpre todos os requisitos do enunciado, disponibilizando uma CLI intuitiva capaz de apresentar resultados em formato JSON ou CSV. As funcionalidades implementadas mostraram-se robustas e complementares: a consulta detalhada de anúncios demonstrou a capacidade de combinar múltiplas fontes; a geração de estatísticas permitiu analisar a distribuição geográfica das ofertas; e a identificação das principais skills revelou tendências tecnológicas do mercado. A possibilidade de exportação para CSV aumentou a utilidade prática do sistema, facilitando análises externas.

No conjunto, o projeto evidenciou a importância do pré-processamento, da automação e da estruturação de dados na Ciência de Dados, com os testes a confirmarem o funcionamento consistente e correto da aplicação.