

Universidade do Minho
Escola de Ciências
Licenciatura em Ciência de Dados

Relatório

Ambientes e Linguagens de Programação para Ciência de Dados

Adriana Couto, a106869

Luana Vilaverde, a106833

Maria Castro, a106828

**Corpo Docente: Pedro Manuel Rangel Santos Henriques, Tiago João
Fernandes Baptista**

Ano Letivo 2024/2025

Índice

1	Introdução	2
2	Bibliotecas Utilizadas	2
3	Funcionalidades Desenvolvidas	2
3.1	Funções	2
3.2	Funcionalidades	4
3.2.1	Listar os N Trabalhos Mais Recentes	4
3.2.2	Listar os Trabalhos Full-Time de uma Empresa numa Localidade	5
3.2.3	Salário Oferecido num Determinado Trabalho	6
3.2.4	Trabalhos que Requerem uma Determinada Lista de Skills . .	6
3.2.5	Obter Informações Detalhadas de um jobID	7
3.2.6	Contagem de vagas	8
3.2.7	Principais skills pedidas num determinado trabalho.	10
3.2.8	Alternativas de Sites: Indeed e SimplyHired	11
3.2.9	Extrair Contacto de uma Vaga de Emprego	11
3.2.10	Extrair Email de uma Vaga de Emprego	12
4	Conclusão	12
5	Referências	13

1 Introdução

Este trabalho consistiu no desenvolvimento de uma CLI (*Command Line Interface*) para explorar e processar ofertas de emprego disponíveis na API `itjobs.pt`, com o objetivo de fornecer informações detalhadas sobre as oportunidades de trabalho. A solução implementada recorreu a diversos recursos da linguagem Python, incluindo requisições a APIs REST, expressões regulares para filtragem e transformação de dados, manipulação de ficheiros CSV e Web Scraping com a biblioteca *Beautiful Soup* para enriquecer as informações obtidas. O projeto foi realizado em duas etapas: a primeira foca-se na extração e processamento de dados da API, enquanto a segunda expande as funcionalidades, incorporando dados de fontes externas e realizando análises mais complexas.

2 Bibliotecas Utilizadas

Para a realização do projeto, foram utilizadas as seguintes bibliotecas:

- **Typer:** Para criar a interface de linha de comando (CLI).
- **Requests:** Para realizar requisições HTTP à API `itjobs.pt`.
- **Beautiful Soup:** Utilizada em *web scraping* para extrair informações de web-sites.
- **re:** Para trabalhar com expressões regulares, facilitando a extração e manipulação de dados textuais.
- **datetime:** Para trabalhar com datas e intervalos de tempo.
- **json:** Para apresentar os dados em formato JSON (incluindo importação e exportação).
- **csv:** Para criar e exportar arquivos CSV.
- **collections:** Utilizada principalmente pela classe `Counter`, que permite a contagem de ocorrências de elementos em coleções de dados como listas ou sequências textuais. Essa funcionalidade foi essencial para identificar padrões e frequências em conjuntos de dados.

3 Funcionalidades Desenvolvidas

3.1 Funções

Para o desenvolvimento das funcionalidades descritas, foram criadas as seguintes funções auxiliares:

- **response(page):** Realiza uma requisição à API, `itjobs.pt`, e retorna os dados da página especificada no formato JSON.

- `exportar_csv(data, filename='jobs.csv')`: Exporta uma lista de dicionários para um ficheiro CSV, criando o cabeçalho com base nas chaves dos dicionários e garantindo que todas as chaves sejam incluídas.
- `exibir_output(jobs)`: Exibe os dados em formato JSON no terminal.
- `fetch_ambitionbox_data(company_name)`: Obtém informações sobre uma empresa no site *AmbitionBox*. Utiliza o método `select_one()` para localizar a tag `div` com o atributo `data-testid="reviewRating"` e extrair o `span` associado, que contém a avaliação da empresa. Para encontrar os principais benefícios da empresa, utiliza o método `find()` para localizar um `div` com a classe `css-175oi2r flex flex-col flex-1`, seguido de `find_all('h4')` para extrair até três elementos `h4`, representando os benefícios. Caso os elementos não sejam encontrados, retorna NA como valor padrão.
- `fetch_indeed_data(company_name)`: Obtém informações sobre uma empresa no site *Indeed*. Localiza a classificação da empresa utilizando o método `find()` para identificar um `span` com o atributo `aria-hidden="true"` e extrai o seu conteúdo com `.text`. Para o setor de atuação, utiliza `find()` com a classe `css-vjn8gb e1wnkr790`. Se não forem encontrados elementos, retorna NA.
- `fetch_hired_data(company_name)`: Obtém informações sobre uma empresa no site *SimplyHired*. Localiza a classificação da empresa utilizando o método `find()` para identificar um `span` com o atributo `aria-hidden="true"` e extrai o texto correspondente com `.text`. Os principais benefícios são procurados em elementos `p` com a classe `chakra-text css-1tluwxv`, e o texto é extraído utilizando `get_text(strip=True)`. Para o setor da empresa, utiliza o método `find(attrs={"data-testid": "cp-industry"})` e localiza o elemento seguinte com `find_next("p")`.
- `get_skills_from_job(job_url)`: Obtém a lista de *skills* de uma vaga de emprego a partir de sua URL no site *AmbitionBox*. Localiza as *tags* que contêm as *skills* utilizando o método `find_all('a', class_='body-medium chip')` (que encontra todos os elementos `a` com a classe especificada). O texto de cada elemento é extraído com o método `get_text(strip=True)` e convertido para letras minúsculas. A função retorna uma lista contendo os nomes das *skills*.
- `get_job_urls(job_title)`: Obtém uma lista de URLs de vagas relacionadas a um título de trabalho no site *AmbitionBox*. Substitui espaços no nome do cargo por hifens (`replace(" ", "-")`) para adequar a URL. Realiza uma requisição utilizando `requests.get()` e processa o HTML com *BeautifulSoup*. Localiza as *divs* contendo informações das vagas usando `find_all('div', class_='jobsInfoCardCont')`. Para cada *div*, encontra o elemento `a` que contém o link para a vaga, extrai o valor do atributo `href` e cria a URL completa concatenando-a com a URL base (`f"https://www.ambitionbox.com"`). A função retorna uma lista de URLs de vagas.

3.2 Funcionalidades

3.2.1 Listar os N Trabalhos Mais Recentes

Para desenvolver esta funcionalidade, foi criada a função `top`, que lista os N trabalhos mais recentes publicados pela `itjobs.pt`. Observando a API, constatou-se que os trabalhos mais recentes estão listados no início de cada página. No entanto, como a API tem um limite de 12 trabalhos por página, foi necessário implementar uma solução que garantisse a recuperação de mais de 12 trabalhos caso o utilizador solicitasse.

A implementação utiliza um ciclo que percorre várias páginas da API até atingir o número total de trabalhos requisitado (N) ou até encontrar uma página sem resultados disponíveis. Este processo garante que todos os trabalhos solicitados sejam obtidos, mesmo em cenários com limites impostos pela API.

Após a coleta dos trabalhos, os N primeiros itens são formatados e exibidos no terminal em formato **JSON**. A Figura 1 apresenta um exemplo de saída dessa funcionalidade no terminal.

[illegible]

Figure 1: Exemplo de output do comando `top` no terminal em formato JSON

O utilizador também tem a opção de exportar os dados para um ficheiro **CSV**. A exportação organiza os dados em colunas pré-definidas, como *Título*, *Empresa*, *Descrição*, *Data de publicação*, *Localização* e *Salário*.

Título,Empresa,Descrição,Data de publicação,Localização,Salário

Product Owner (Banking),Mind Source,"<p>Local: Lisboa (Híbrido)</p>

Intermediate Project Manager,Imaginary Cloud,"<p>if you're passive</p>

Gestor de Projetos,Opensoft - Soluções Informáticas, SA,"<p><st

TIBCO Consultant,IT People Innovation,"<p>[Estás à altura deste d

Data Engineer,Altair.io,"<p>Hello!</p>>Are you

Senior Quality Assurance Engineer,Go Yonder,"<p>ABOUT YON

DevOps Engineer,Bee Engineering,"<p>// Bee motto

Engenheiro de Automação,akapeople,"<p>o nosso cliente é uma empre

Administrador de Sistemas IT,Bleupharma - Indústria Farmacêutica,

React developer,KCS IT,"<p>We're looking for the special,

Senior DevOps Engineer w/GCP,Aubay,"<p>Your connection with Aubay

Java Expert,Devoteam,"<p><u>About Devoteam</u>

Vue.js Developer,KWAN,"<p>At KWAN, we don't just offer jobs - we

Figure 2: Exemplo da estrutura de um ficheiro **CSV** com os dados obtidos

Este processo de exportação garante que caracteres especiais sejam incluídos corretamente no ficheiro através do uso da codificação UTF-8.

A funcionalidade `top` foi projetada para manter o comportamento esperado mesmo em casos onde o número de trabalhos disponíveis é inferior ao requisitado pelo utilizador.

3.2.2 Listar os Trabalhos Full-Time de uma Empresa numa Localidade

Foi implementada a função `search` que serve para imprimir todos os trabalhos do tipo full-time publicados numa determinada empresa, numa determinada localidade. Os parâmetros utilizados incluem `location`, que especifica o nome da localidade onde os trabalhos foram publicados; `company`, que indica o nome da empresa que publicou os trabalhos; `limit`, que é opcional e define o número de trabalhos a exibir; e `export_csv`, também opcional, que permite exportar os dados obtidos para um arquivo CSV. A saída exibe os trabalhos no terminal em formato JSON e, se solicitado, exporta os dados para um arquivo CSV.

A função começa por percorrer cada trabalho retornado pela API e comparar o nome da empresa com o inserido pelo utilizador (isto para todas as páginas da API até encontrar o nome pretendido). Verifica se o tipo de contrato é *full-time* e, em seguida, se a localização do trabalho coincide com a inserida pelo utilizador. Esses trabalhos são adicionados a uma lista. Por fim, se o utilizador definir um limite de trabalhos a exibir, a função imprime apenas os primeiros n trabalhos dessa lista (caso contrário imprime a lista completa).

[illegible]

Figure 3: Exemplo de output do comando `search`

[illegible]

Figure 4: Dados da API correspondentes.

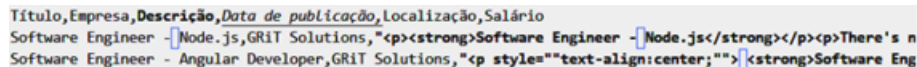


Figure 5: Exemplo da estrutura de um ficheiro CSV com os dados obtidos

3.2.3 Salário Oferecido num Determinado Trabalho

Para esta funcionalidade criamos a função `salary`, que tem como objetivo extrair a informação do salário a partir do `job_id` fornecido, mesmo quando o campo `wage` estiver vazio. Começamos com uma requisição à API, que retorna uma lista de trabalhos. Para encontrar o trabalho desejado, a função percorre as páginas da API até encontrar o trabalho com o `job_id` específico. Primeiramente, a função faz uma requisição à API e verifica se a página contém resultados. Se a página estiver vazia ou não contiver o `job_id` requisitado, o processo termina. Caso o trabalho seja encontrado, a função procura pelo salário diretamente no campo `wage` do trabalho. Se o campo `wage` contiver um valor, o salário é exibido imediatamente. Caso contrário, a função tenta encontrar o salário no `body` do trabalho utilizando uma expressão regular.

```
r"(\d{3,})([.,]\d{3})?\s?(€|\$|USD|£|₹)"
```

Figure 6: Expressão regular utilizada para procurar o salário na descrição do trabalho

A expressão procura por números que tenham pelo menos três dígitos e que estejam seguidos por um símbolo monetário, como €, \$, USD, £ ou . Também tem em conta separadores de milhares como ponto ou vírgula. Caso a expressão regular encontre o valor do salário no `body`, este é exibido. Caso contrário, é apresentada ao utilizador a mensagem "Salário não especificado".

Figure 7: Exemplo de output do comando `salary`

3.2.4 Trabalhos que Requerem uma Determinada Lista de Skills

A funcionalidade `skills` serve para retornar uma lista de trabalhos que requerem uma determinada lista de skills num período específico. Tem como parâmetros uma lista de `skills`, uma `data_inicial`, uma `data_final` e a opção de exportar os resultados para CSV.

Para desenvolver a funcionalidade, começámos por guardar todos os trabalhos da API (de todas as páginas) numa lista, `jobs`. As datas inicial e final inseridas pelo utilizador foram convertidas para objetos `datetime`, para permitir comparações com as datas dos trabalhos. Em seguida, a data de publicação de cada trabalho,

armazenada na variável `publishedAt`, foi convertida para uma lista de duas strings (usando a função `split`), para extrair apenas a data, já que a variável continha tanto a data quanto a hora. A data de publicação foi então convertida para um objeto `datetime` e os trabalhos foram filtrados para selecionar apenas aqueles cujas datas de publicação estavam entre as datas inicial e final fornecidas. Por fim, foi feita uma seleção para incluir apenas os trabalhos que continham todas as skills inseridas.

```

"Title": "Data Analyst",
"Skills": "Oracle to Java",
"Description": "We are looking for a Data Analyst with strong SQL and DEVELOP TECHNOLOGY skills. You will be responsible for analyzing data, creating reports, and supporting the development team. This role requires a strong understanding of data structures and the ability to work with large datasets. You will be working closely with the development team to ensure data integrity and accuracy. The ideal candidate will have a strong background in data analysis and a passion for technology. You will be responsible for analyzing data, creating reports, and supporting the development team. This role requires a strong understanding of data structures and the ability to work with large datasets. You will be working closely with the development team to ensure data integrity and accuracy. The ideal candidate will have a strong background in data analysis and a passion for technology."
},
{
"Title": "Software Engineer",
"Skills": "Python, Java, JavaScript, React, Node.js, AWS, Docker, Kubernetes",
"Description": "We are looking for a Software Engineer with strong Python, Java, JavaScript, React, Node.js, AWS, Docker, and Kubernetes skills. You will be responsible for developing and maintaining our software applications. This role requires a strong understanding of software development principles and the ability to work with large teams. You will be working closely with the product team to ensure our software meets the needs of our customers. The ideal candidate will have a strong background in software development and a passion for technology. You will be responsible for developing and maintaining our software applications. This role requires a strong understanding of software development principles and the ability to work with large teams. You will be working closely with the product team to ensure our software meets the needs of our customers. The ideal candidate will have a strong background in software development and a passion for technology."
},
{
"Title": "Product Manager",
"Skills": "Product Management, Marketing, Sales, Customer Support",
"Description": "We are looking for a Product Manager with strong Product Management, Marketing, Sales, and Customer Support skills. You will be responsible for managing our product line and ensuring our products meet the needs of our customers. This role requires a strong understanding of product management principles and the ability to work with large teams. You will be working closely with the development team to ensure our products are of high quality. The ideal candidate will have a strong background in product management and a passion for technology. You will be responsible for managing our product line and ensuring our products meet the needs of our customers. This role requires a strong understanding of product management principles and the ability to work with large teams. You will be working closely with the development team to ensure our products are of high quality. The ideal candidate will have a strong background in product management and a passion for technology."
},
{
"Title": "Sales Representative",
"Skills": "Sales, Marketing, Customer Support",
"Description": "We are looking for a Sales Representative with strong Sales, Marketing, and Customer Support skills. You will be responsible for selling our products and services to our customers. This role requires a strong understanding of sales principles and the ability to work with large teams. You will be working closely with the product team to ensure our products meet the needs of our customers. The ideal candidate will have a strong background in sales and a passion for technology. You will be responsible for selling our products and services to our customers. This role requires a strong understanding of sales principles and the ability to work with large teams. You will be working closely with the product team to ensure our products meet the needs of our customers. The ideal candidate will have a strong background in sales and a passion for technology."
},
{
"Title": "Customer Support Representative",
"Skills": "Customer Support, Sales, Marketing",
"Description": "We are looking for a Customer Support Representative with strong Customer Support, Sales, and Marketing skills. You will be responsible for providing support to our customers and ensuring they are satisfied with our products and services. This role requires a strong understanding of customer support principles and the ability to work with large teams. You will be working closely with the product team to ensure our products meet the needs of our customers. The ideal candidate will have a strong background in customer support and a passion for technology. You will be responsible for providing support to our customers and ensuring they are satisfied with our products and services. This role requires a strong understanding of customer support principles and the ability to work with large teams. You will be working closely with the product team to ensure our products meet the needs of our customers. The ideal candidate will have a strong background in customer support and a passion for technology."
}
]

```

Figure 8: Exemplo de output do comando `skills`

Caso o utilizador opte por exportar os dados para CSV, os resultados serão organizados e exportados contendo as respetivas informações dos trabalhos encontrados, como o título do trabalho, a empresa, a data de publicação e as skills requeridas, entre outras.

[illegible]

Figure 9: Exemplo da estrutura de um ficheiro **CSV** com os dados obtidos

3.2.5 Obter Informações Detalhadas de um jobId

A funcionalidade `get_job_details` é responsável por obter informações detalhadas de uma vaga específica, identificada pelo `jobID`. Começa por iterar cada página da API, verificando se o `jobID` fornecido corresponde ao identificador de alguma vaga. Quando encontrado, os principais detalhes da vaga, como título, empresa, descrição,

data de publicação, localização, salário e descrição da empresa, são extraídos e armazenados.

Além disso, se o nome da empresa estiver disponível, a funcionalidade chama a função `fetch_ambitionbox_data` para adicionar detalhes da vaga com informações obtidas do site *AmbitionBox*. Nesta função, o conteúdo HTML da página da empresa é analisado com a biblioteca `BeautifulSoup`. Para localizar elementos específicos na página, utilizamos comandos como:

- `select_one('div[data-testid="reviewRating"] span')` (localiza o primeiro elemento `span` dentro de um `div` com atributo `data-testid="reviewRating"`).
- `find('div', class_='css-175oi2r flex flex-col flex-1')` (localiza o primeiro elemento `div` com a classe especificada).
- `find_all('h4')` (retorna uma lista com todos os elementos `h4` encontrados dentro do `body`).

Após localizar os elementos desejados, utilizamos o atributo `.text` para extrair apenas o texto contido em cada elemento.

Os resultados obtidos incluem a avaliação geral da empresa e os principais benefícios de trabalhar na mesma.

Por fim, os resultados são apresentados em formato **JSON**, e, opcionalmente, podem ser exportados para um arquivo **CSV**.

[illegible]

Figure 10: Exemplo de output do comando `get_job_details`

Título	Empresa	Descrição da vaga	Data de publicação	Localização	Salário	Descrição da empresa	Rating da empresa (0 a 5)	Principais benefícios de trabalhar na empresa
Data Engineer	(Snowflake & DBT), SDG Group	"<p>do you want to be part of a company that specializes 100% in its own #Data & #Analytics projects? If so, keep reading!</p><p>Even though we are relatively						

Figure 11: Exemplo da estrutura de um ficheiro **CSV** com os dados obtidos.

3.2.6 Contagem de vagas

A funcionalidade `statistics` tem como objetivo criar um relatório sobre o número de vagas de emprego disponíveis, agrupadas por título, zona e tipo de trabalho. O resultado pode ser exportado para um ficheiro CSV.

A funcionalidade começa por criar um `defaultdict(int)` para contar as vagas por combinação de título (trabalho), zona (localização) e tipo de trabalho (full-time ou part-time). Em seguida, faz requisições à API `itjobs.pt`, página por página, para obter os trabalhos. Para cada trabalho, são extraídos o título, a zona (localização) e o tipo de trabalho, e o contador no dicionário é incrementado para cada combinação desses dados.

Após processar todas as páginas, os dados são organizados numa lista e ordenados por título. Se o utilizador optar por exportar, a funcionalidade chama a função `exportar_csv`, criando um ficheiro CSV com os resultados.

```
PS C:\Users\Vila Verde\OneDrive\2ºano\Ambientes e Linguagens de Programação para
Ciência de dados\ALPCD_6> python jobscli.py statistics --export-csv
Dados exportados para jobs.csv
```

Figure 12: Mensagem de verificação no terminal

```
Título,Zona,Tipo de Trabalho,Nº de Vagas
.net & nodejs backend developer,Porto,Full-time,1
.net developer,Lisboa,Full-time,4
.net developer,Porto,Full-time,4
.net developer & react.js,Lisboa,Full-time,1
.net developer (french speaker),Porto,Full-time,1
.net software developer,Porto,Full-time,2
.net specialist,Lisboa,Full-time,1
accipiens developer,Porto,Full-time,1
administrador de base de dados,Lisboa,Full-time,1
administrador de cloud (azure),Lisboa,Full-time,1
administrador de sistemas de ti,Lisboa,Full-time,1
administrador de sistemas it,Coimbra,Full-time,1
administrador de sistemas linux,Porto,Full-time,1
administrador de sistemas linux mid,Lisboa,Full-time,1
"administrador sistemas microsoft, networking e cibersegurança",Lisboa,Full-time,1
administrador/a de sistemas windows,Lisboa,Full-time,1
ai architect,Lisboa,Full-time,1
ai architect,Porto,Full-time,1
analista de cibersegurança (suporte),Lisboa,Full-time,1
analista funcional,Lisboa,Full-time,3
analista funcional,Porto,Full-time,1
analista funcional júnior,Lisboa,Full-time,1
analista funcional júnior,Porto,Full-time,1
analista programador .net,Lisboa,Full-time,1
analista programador mainframe,Porto,Full-time,1
analyst developer,Lisboa,Full-time,1
android developer,Lisboa,Full-time,4
android developer,Porto,Full-time,1
android developer (kotlin),Porto,Full-time,1
android engineer,Aveiro,Full-time,1
android engineer,Porto,Full-time,1
android native,Leiria,Full-time,1
android native,Lisboa,Full-time,1
android native,Porto,Full-time,1
angular developer,Lisboa,Full-time,4
angular developer,Porto,Full-time,3
angular developer,Leiria,Full-time,2
```

Figure 13: Exemplo da estrutura de um ficheiro CSV

3.2.7 Principais skills pedidas num determinado trabalho.

A funcionalidade `list_skills` tem como objetivo imprimir uma lista das skills solicitadas em ofertas de trabalho, mostrando o número de vezes que cada uma aparece, para um determinado emprego. A funcionalidade começa por chamar a função `get_job_urls(job_title)`, que procura as URLs das vagas relacionadas com o título de trabalho fornecido. Essa função faz uma requisição ao site *AmbitionBox* e retorna uma lista de URLs de ofertas de emprego. De seguida, para cada URL retornado pela função `get_job_urls`, a função `get_skills_from_job(url)` é chamada. Esta função realiza uma requisição HTTP à página de cada vaga, extrai as skills solicitadas e retorna uma lista de skills específicas para o cargo.

Com todas as skills extraídas, a função conta a frequência de cada skill nas diferentes vagas, utilizando o `Counter` da biblioteca `collections`. A lista das skills mais solicitadas é organizada e impressa em formato JSON. Caso o argumento `export_csv` seja `True`, as skills são exportadas para um arquivo CSV utilizando a função `exportar_csv(skills)`.

```
PS C:\Users\Vila Verde\OneDrive\2ºano\Ambientes e Linguagens de Programação para Ciências de dados\ALPCD_6º python jobscli.py "data scientist" --export-csv
[
  {
    "skill": "python",
    "count": 10
  },
  {
    "skill": "machine learning",
    "count": 8
  },
  {
    "skill": "data science",
    "count": 8
  },
  {
    "skill": "sql",
    "count": 6
  },
  {
    "skill": "aws",
    "count": 5
  },
  {
    "skill": "data analysis",
    "count": 3
  },
  {
    "skill": "artificial intelligence",
    "count": 3
  },
  {
    "skill": "data entry",
    "count": 3
  },
  {
    "skill": "data analytics",
    "count": 3
  },
  {
    "skill": "clinical data management",
    "count": 2
  }
]
```

Figure 14: Exemplo do output da função `list_skills`

```
skill,count
python,10
machine learning,8
data science,8
sql,6
aws,5
data analysis,3
artificial intelligence,3
data entry,3
data analytics,3
clinical data management,2
```

Figure 15: Exemplo da estrutura de um ficheiro CSV

3.2.8 Alternativas de Sites: Indeed e SimplyHired

Uma alternativa para obter mais informações acerca das empresas foi a implementação da funcionalidade `get_job_details`, adaptada para os sites *Indeed* e *SimplyHired*. Estes aparecem como argumentos opcionais na funcionalidade, sendo que cada um utiliza a sua respetiva função (`fetch_indeed_data` e `fetch_hired_data`) para obter os dados. Assim, a funcionalidade procura informações específicas disponíveis no *Indeed*, como a classificação da empresa e o setor de atuação, ou no *SimplyHired*, como a avaliação da empresa (0 a 5), o setor de atuação e os principais benefícios de trabalhar na empresa.

A tentativa de obtenção de dados é feita por web scraping, contudo, tanto o Indeed como o SimplyHired apresentaram o erro 403 (proibido - o servidor recusou a solicitação, indicando que compreendeu o pedido, mas não permite acesso), o que indica que apresentam proteções avançadas contra web scraping, o que impediu a obtenção de dados diretamente dos sites.

3.2.9 Extrair Contacto de uma Vaga de Emprego

A função `contacto` extrai o número de telefone de uma vaga de emprego específica, identificada pelo `job_id`. Começa por procurar o número nas várias páginas da API, onde cada página contém uma lista de vagas. Percorre as páginas uma a uma até encontrar a vaga correta. Se a página não contiver a vaga desejada, a função passa para a página seguinte. Quando encontra a vaga com o `job_id` desejado, a função verifica primeiro o campo `company` para ver se o telefone está diretamente disponível, através de `company.get('phone')`. Caso o número de telefone não esteja diretamente preenchido, a função tenta extraí-lo da descrição da vaga (`body`) ou da descrição da empresa, utilizando uma expressão regular.

```
if job:
    company = job.get('company', {})
    phones = company.get('phone')
    if not phones:
        body = job.get("body", None)
        phones = re.search(r"\b((\+351)?(9|2)\d{2}\s?\d{3}\s?\d{3})\b", body)
    if not phones:
        description = company.get('description', None)
        phones = re.search(r"\b((\+351)?(9|2)\d{2}\s?\d{3}\s?\d{3})\b", description)
```

Figure 16: Expressão regular utilizada

Esta expressão procura números de telefone portugueses, tanto com o indicativo +351 como sem ele. O padrão de número considerado começa com 9 ou 2, seguido por nove dígitos. O símbolo `_` no início e no final da expressão regular garante esta procure apenas números completos e isolados, evitando apanhar partes de outros números. A função `re.findall` é utilizada para encontrar todos os números correspondentes na descrição, o que procurar mais do que um número, caso existam.

Se existir algum número de telefone, o mesmo é exibido ao utilizador. Caso contrário, a função imprime uma mensagem indicando que o número de telefone não foi especificado.

```
PS C:\Users\adria> & C:/Python312/python.exe "c:/Users/adria/OneDrive/Documents/uni/2º ano/1º semestre/ambientes/Nova pasta/ALPCD_6/jobscli.py" contacto 492263
Telefones disponíveis: +351 213 174 164
PS C:\Users\adria> & C:/Python312/python.exe "c:/Users/adria/OneDrive/Documents/uni/2º ano/1º semestre/ambientes/Nova pasta/ALPCD_6/jobscli.py" email 492263
Nenhum email especificado.
```

Figure 17: Exemplo de output do comando `contacto`.

3.2.10 Extrair Email de uma Vaga de Emprego

A função `email` tem como objetivo extrair o endereço de e-mail de uma vaga de emprego específica, identificada pelo `job.id`. A função funciona exatamente da mesma forma que a função `contacto`, divergindo apenas na expressão regular utilizada.

```
if job:
    company = job.get('company', {})
    emails = company.get('email')
    if not emails:
        body = job.get("body", None)
        emails = re.search(r"\b[A-Za-z0-9._-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}\b", body)
    if not emails:
        description = company.get('description', None)
        emails = re.search(r"\b[A-Za-z0-9._-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}\b", description)
```

Figure 18: Expressão regular utilizada

A expressão foi projetada para procurar e-mails padrão, isto é, começando com letras, números ou alguns caracteres especiais, seguido pelo símbolo `@`, o domínio e a extensão do e-mail. A função usa `re.findall` para encontrar todos os e-mails na descrição do trabalho.

Caso a função encontre um e-mail, o mesmo é exibido ao utilizador. Caso contrário, a função imprime a mensagem "E-mail não especificado".

```
PS C:\Users\adria> & C:/Python312/python.exe "c:/Users/adria/OneDrive/Documents/uni/2º ano/1º semestre/ambientes/Nova pasta/ALPCD_6/jobscli.py" contacto 492282
Nenhum número de telefone especificado.
PS C:\Users\adria> & C:/Python312/python.exe "c:/Users/adria/OneDrive/Documents/uni/2º ano/1º semestre/ambientes/Nova pasta/ALPCD_6/jobscli.py" email 492282
Emails disponíveis: lisbon@altar.io
```

Figure 19: Exemplo de output do comando `email`.

4 Conclusão

Este projeto permitiu desenvolver uma ferramenta útil para explorar e processar ofertas de emprego da API `itjobs.pt`, tornando mais acessível a consulta de informações sobre o mercado de trabalho. Ao longo do processo, aplicámos diversas técnicas de programação em Python, desde requisições à API, filtragem de dados com expressões regulares, até à exportação de ficheiros CSV e enriquecimento de dados com Web Scraping.

5 Referências

- **SimplyHired:** Simplyhired.pt. (2024). Motor de pesquisa de emprego — SimplyHired. [online] Disponível em: <https://www.simplyhired.pt/> [Acessado a 30 Dec. 2024].
- **Indeed:** pt.indeed.com. (n.d.). Pesquisa grátis de ofertas de emprego — Indeed. [online] Disponível em: <https://pt.indeed.com/> [Acessado a 30 Dec. 2024].
- **AmbitionBox:** www.ambitionbox.com (n.d.). Company Reviews, Salaries by Company, Interview Questions, Salary Calculator. [online] AmbitionBox. Disponível em: <https://www.ambitionbox.com/> [Acessado a 30 Dec. 2024].
- **Typer:** typer.tiangolo.com. (n.d.). Typer. [online] Disponível em: <https://typer.tiangolo.com/> [Acessado a 30 Dec. 2024].
- **Requests:** requests.readthedocs.io. (n.d.). Requests: HTTP for Humans™ — Requests 2.31.0 documentation. [online] Disponível em: <https://requests.readthedocs.io/> [Acessado a 30 Dec. 2024].
- **Beautiful Soup:** Readthedocs.io. (2020). Beautiful Soup Documentation — Beautiful Soup 4.4.0 documentation. [online] Disponível em: <https://beautiful-soup-4.readthedocs.io/> [Acessado a 30 Dec. 2024].
- **API ITJobs:** Itjobs.pt. (2024). API - Documentation / Intro - ITJobs. [online] Disponível em: <https://www.itjobs.pt/api/docs> [Acessado a 30 Dec. 2024].
- **Biblioteca Collections:** docs.python.org. (n.d.). collections — Container datatypes — Python 3.8.3 documentation. [online] Disponível em: <https://docs.python.org/3/library/collections.html> [Acessado a 30 Dec. 2024].