



**Universidade Federal de Minas Gerais**

**Turma:** Ciência da Computação **Prof.:** Renato

**Nome:** Diane Fenzi Gonçalves  
Rubia Alice Moreira de Souza

## **Trabalho Prático 1 – Manipulação de sequências**

Belo Horizonte  
2025

# Sumário

<b>1. Problema Abordado</b>	<b>3</b>
<b>2. Passos de Implementação</b>	<b>3</b>
2.1. Implementação da Árvore Trie Compacta	3
2.2. Indexação	4
2.3. Busca	5
2.4. Endpoint API	6
2.5. Front-end	7
<b>3. Como Executar</b>	<b>10</b>

# 1. Problema Abordado

O problema abordado pelo trabalho é o desenvolvimento de uma máquina de busca de notícias da BBC News. Para isso foi necessária a implementação de vários módulos. Primeiro, um módulo para a estrutura, que cria índices reversos para cada notícia do corpus selecionadas para o trabalho. Depois, um módulo de Índice Reverso responsável por criar a indexação dos arquivos, um módulo de Recuperação de Informação e, por fim, uma interface gráfica para utilização do sistema.

## 2. Passos de Implementação

### 2.1. Implementação da Árvore Trie Compacta

O primeiro passo da implementação foi criar uma estrutura de Árvore Trie Compacta. Ela é constituída de três classes: WordInfo, TrieNode e CompressedTrie. A WordInfo é responsável por armazenar os dados relacionados a uma palavra de uma notícia. Ele armazena a própria palavra, a frequência dela em todo o corpus e os documentos em que ela ocorre. Um TrieNode armazena o prefixo da subárvore que ele pertence, os filhos e se ele é o final de uma palavra. Já a CompressedTrie possui um TrieNode que é a raiz de toda a árvore.

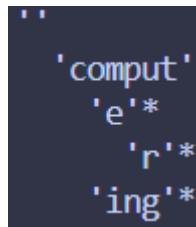
Uma CompressedTrie possui 7 métodos públicos. “*insert()*” responsável por inserir uma palavra na árvore trie. “*search()*” responsável por verificar se uma palavra está registrada na árvore trie. “*get\_word\_info()*” é responsável por retornar o WordInfo associado a uma palavra. “*print\_trie()*” e “*print\_all\_words()*” são métodos para debug que imprimem as informações armazenadas na árvore trie. O primeiro, imprime a sua estrutura de como a árvore está construída. Já o segundo, imprime todas as palavras armazenadas e a WordInfo delas. A Árvore Trie tem dois métodos para auxiliar a manipulação de arquivos. “*jsonify()*” constrói uma estrutura de lista de dicionários (*list[dict]*) para armazenar a Árvore Trie e seu conteúdo em um arquivo de texto. “*create\_from\_json()*” é o complemento ao “*jsonify()*”. Ele é utilizado para fazer a leitura do arquivo texto em que a Árvore Trie foi armazenada e reconstruí-la com os dados do arquivo. Além desses métodos públicos, ela possui um método privado “*\_\_common\_prefix\_length()*” responsável por encontrar o tamanho do maior prefixo em comum entre duas palavras.

Então, foi feito um arquivo de teste para a CompressedTrie. Nele construímos duas tries. A primeira é a Árvore Trie do arquivo “001.txt” da pasta business. Depois, imprimimos o “*jsonify()*” da árvore:

```
[{"word": "a", "frequency": 8, "documents": ["noticias/business/001.txt"]}, {"word": "ad", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "advert", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "advertising", "frequency": 2, "documents": ["noticias/business/001.txt"]}, {"word": "adjust", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "at", "frequency": 3, "documents": ["noticias/business/001.txt"]}, {"word": "an", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "and", "frequency": 10, "documents": ["noticias/business/001.txt"]}, {"word": "analysts", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "aol", "frequency": 3, "documents": ["noticias/business/001.txt"]}, {"word": "aol.", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "aol", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "aol's", "frequency": 2, "documents": ["noticias/business/001.txt"]}, {"word": "also", "frequency": 2, "documents": ["noticias/business/001.txt"]}, {"word": "alexander", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "all", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "already", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "around", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "accounts", "frequency": 2, "documents": ["noticias/business/001.txt"]}, {"word": "as", "frequency": 3, "documents": ["noticias/business/001.txt"]}, {"word": "aside", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "amount", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "sale", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "sales", "frequency": 3, "documents": ["noticias/business/001.txt"]}, {"word": "said", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "saw", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "search-engine", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "service", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "settle", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "set", "frequency": 2, "documents": ["noticias/business/001.txt"]}, {"word": "stronger", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "strong", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "stake", "frequency": 2, "documents": ["noticias/business/001.txt"]}, {"word": "stake.", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "sign", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "slightly", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "slump", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "sharp", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "boost", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "book", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "box-office", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "biggest", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "benefited", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "before", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "better", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "bertelsmann's", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "buoyed", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "but", "frequency": 2, "documents": ["noticias/business/001.txt"]}, {"word": "by", "frequency": 3, "documents": ["noticias/business/001.txt"]}, {"word": "back", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "bros.", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "broadband", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "time Warner", "frequency": 7, "documents": ["noticias/business/001.txt"]}, {"word": "to", "frequency": 18, "documents": ["noticias/business/001.txt"]}, {"word": "the", "frequency": 19, "documents": ["noticias/business/001.txt"]}, {"word": "three", "frequency": 2, "documents": ["noticias/business/001.txt"]}, {"word": "that", "frequency": 3, "documents": ["noticias/business/001.txt"]}, {"word": "than", "frequency": 2, "documents": ["noticias/business/001.txt"]}, {"word": "third", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "try", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "trilogy", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "warner", "frequency": 3, "documents": ["noticias/business/001.txt"]}, {"word": "warner's", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "was", "frequency": 2, "documents": ["noticias/business/001.txt"]}, {"word": "way", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "which", "frequency": 5, "documents": ["noticias/business/001.txt"]}, {"word": "while", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "when", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "were", "frequency": 3, "documents": ["noticias/business/001.txt"]}, {"word": "will", "frequency": 2, "documents": ["noticias/business/001.txt"]}, {"word": "wider", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "with", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "profit", "frequency": 5, "documents": ["noticias/business/001.txt"]}, {"word": "profits", "frequency": 5, "documents": ["noticias/business/001.txt"]}, {"word": "probe", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "projecting", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "preceding", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "previously", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "posted", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "performance", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "parsons", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "part", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "pay", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "publisher", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "purchase", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "quarter", "frequency": 3, "documents": ["noticias/business/001.txt"]}, {"word": "quarterly", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "quarters", "frequency": 1, "documents": ["noticias/business/001.txt"]}, {"word": "us", "frequency": 1, "documents": ["noticias/business/001.txt"]}]
```

Parte do resultado `jsonify()` arquivo 001.txt categoria business

Como segundo teste, realizamos a inserção das palavras “computer”, “compute” e “computing”. Isso gera a seguinte árvore trie como saída:



Árvore trie resultante das palavras computer, compute e computing

Depois, realizamos a busca das palavras “computer”, “compute” e “computing”, todas retornando verdadeiro, e depois “comp” que retorna falso.

## 2.2. Indexação

Depois de implementada a Árvore Trie, nós decidimos criar o código para gerar os índices invertidos para todas as notícias. Para isso, implementamos a função “`create_compressed_trie_from_raw_files()`” que percorre todas as notícias do corpus, armazenando cada palavra na Árvore Trie. Ao final, conforme solicitado pelo enunciado, armazenamos a árvore em um arquivo de texto chamado “full\_trie.txt”. Sempre que a aplicação é iniciada, ela verifica a existência desse arquivo. Caso ela não o encontre, ele é gerado novamente.

```

1 ['word': 'a', 'frequency': 18254, 'documents': ['noticias/business\\419.txt', 'noticias/politics\\169.txt', 'noticias/sport\\031.txt', 'noticias/tech\\171.txt', 'noticia
2 ['word': 'ad', 'frequency': 14, 'documents': ['noticias/tech\\289.txt', 'noticias/business\\001.txt', 'noticias/tech\\089.txt', 'noticias/politics\\393.txt', 'noticias/h
3 ['word': 'advert', 'frequency': 13, 'documents': ['noticias/business\\009.txt', 'noticias/tech\\195.txt', 'noticias/entertainment\\352.txt', 'noticias/tech\\063.txt', 'n
4 ['word': 'advertise', 'frequency': 12, 'documents': ['noticias/business\\090.txt', 'noticias/business\\279.txt', 'noticias/business\\035.txt', 'noticias/business\\440.
5 ['word': 'advertise', 'frequency': 7, 'documents': ['noticias/tech\\145.txt', 'noticias/tech\\196.txt', 'noticias/business\\442.txt', 'noticias/tech\\286.txt', 'noticia
6 ['word': 'advertisers', 'frequency': 8, 'documents': ['noticias/tech\\305.txt', 'noticias/tech\\206.txt', 'noticias/business\\014.txt', 'noticias/tech\\364.txt']]
7 ['word': 'advertiser's', 'frequency': 1, 'documents': ['noticias/tech\\286.txt']]
8 ['word': 'advertiser-based', 'frequency': 2, 'documents': ['noticias/tech\\286.txt', 'noticias/tech\\391.txt']]
9 ['word': 'advertisements', 'frequency': 5, 'documents': ['noticias/entertainment\\085.txt', 'noticias/politics\\332.txt', 'noticias/business\\226.txt', 'noticias/entert
10 ['word': 'advertisment', 'frequency': 3, 'documents': ['noticias/tech\\362.txt', 'noticias/politics\\315.txt']]
11 ['word': 'adverts', 'frequency': 29, 'documents': ['noticias/tech\\236.txt', 'noticias/tech\\195.txt', 'noticias/tech\\378.txt', 'noticias/business\\449.txt', 'noticias
12 ['word': 'adverse', 'frequency': 5, 'documents': ['noticias/business\\042.txt', 'noticias/politics\\160.txt', 'noticias/business\\258.txt', 'noticias/business\\275.txt', 'n
13 ['word': 'adversely', 'frequency': 2, 'documents': ['noticias/politics\\029.txt', 'noticias/politics\\302.txt']]
14 ['word': 'adversity', 'frequency': 2, 'documents': ['noticias/sport\\495.txt', 'noticias/sport\\048.txt']]
15 ['word': 'advent', 'frequency': 3, 'documents': ['noticias/tech\\289.txt', 'noticias/entertainment\\271.txt', 'noticias/tech\\098.txt']]
16 ['word': 'adventure', 'frequency': 23, 'documents': ['noticias/tech\\095.txt', 'noticias/business\\049.txt', 'noticias/tech\\244.txt', 'noticias/entertainment\\339.txt', 't
17 ['word': 'adventures', 'frequency': 3, 'documents': ['noticias/entertainment\\089.txt', 'noticias/entertainment\\077.txt', 'noticias/entertainment\\378.txt']]
18 ['word': 'adventurous', 'frequency': 2, 'documents': ['noticias/sport\\194.txt', 'noticias/tech\\378.txt']]
19 ['word': 'advertising', 'frequency': 7, 'documents': ['noticias/tech\\244.txt', 'noticias/tech\\378.txt']]
20 ['word': 'advice', 'frequency': 7, 'documents': ['noticias/entertainment\\194.txt', 'noticias/politics\\020.txt', 'noticias/business\\229.txt', 'noticias/sport\\043.txt
21 ['word': 'advisor', 'frequency': 10, 'documents': ['noticias/politics\\361.txt', 'noticias/business\\419.txt', 'noticias/business\\475.txt', 'noticias/business\\459.txt', 'n
22 ['word': 'advisors', 'frequency': 10, 'documents': ['noticias/business\\316.txt', 'noticias/business\\288.txt', 'noticias/sport\\118.txt', 'noticias/business\\122.txt', 'n
23 ['word': 'advisory', 'frequency': 12, 'documents': ['noticias/business\\599.txt', 'noticias/business\\220.txt', 'noticias/business\\280.txt', 'noticias/entertainment\\24
24 ['word': 'advise', 'frequency': 10, 'documents': ['noticias/politics\\003.txt', 'noticias/tech\\037.txt', 'noticias/sport\\386.txt', 'noticias/business\\188.txt', 'notic
25 ['word': 'advises', 'frequency': 4, 'documents': ['noticias/tech\\328.txt', 'noticias/business\\028.txt', 'noticias/tech\\073.txt', 'noticias/tech\\021.txt']]
26 ['word': 'advised', 'frequency': 19, 'documents': ['noticias/business\\316.txt', 'noticias/politics\\378.txt', 'noticias/politics\\353.txt', 'noticias/entertainment\\118
27 ['word': 'adviser', 'frequency': 21, 'documents': ['noticias/business\\229.txt', 'noticias/politics\\052.txt', 'noticias/tech\\017.txt', 'noticias/politics\\189.txt', 'n
28 ['word': 'advisers', 'frequency': 8, 'documents': ['noticias/business\\599.txt', 'noticias/politics\\197.txt', 'noticias/politics\\027.txt', 'noticias/business\\280.txt
29 ['word': 'advising', 'frequency': 3, 'documents': ['noticias/business\\209.txt', 'noticias/business\\229.txt', 'noticias/entertainment\\382.txt']]
30 ['word': 'advanta', 'frequency': 2, 'documents': ['noticias/sport\\467.txt', 'noticias/sport\\471.txt']]
31 ['word': 'advantage', 'frequency': 83, 'documents': ['noticias/tech\\388.txt', 'noticias/sport\\424.txt', 'noticias/politics\\290.txt', 'noticias/tech\\358.txt', 'noticia
32 ['word': 'advantages', 'frequency': 5, 'documents': ['noticias/tech\\286.txt', 'noticias/politics\\408.txt', 'noticias/sport\\160.txt', 'noticias/politics\\394.txt', 'n
33 ['word': 'advance', 'frequency': 43, 'documents': ['noticias/business\\214.txt', 'noticias/politics\\282.txt', 'noticias/business\\240.txt', 'noticias/tech\\276.txt', 'n
34 ['word': 'advanced', 'frequency': 28, 'documents': ['noticias/politics\\021.txt', 'noticias/tech\\163.txt', 'noticias/politics\\090.txt', 'noticias/business\\146.txt', 'n
35 ['word': 'advancement', 'frequency': 9, 'documents': ['noticias/politics\\080.txt', 'noticias/tech\\356.txt', 'noticias/entertainment\\193.txt', 'noticias/tech\\093.txt
36 ['word': 'advances', 'frequency': 9, 'documents': ['noticias/politics\\161.txt', 'noticias/entertainment\\352.txt', 'noticias/politics\\272.txt', 'noticias/tech\\135.txt
37 ['word': 'advancing', 'frequency': 3, 'documents': ['noticias/sport\\474.txt', 'noticias/sport\\146.txt', 'noticias/politics\\274.txt']]
38 ['word': 'advocate', 'frequency': 7, 'documents': ['noticias/politics\\289.txt', 'noticias/politics\\086.txt', 'noticias/politics\\161.txt', 'noticias/entertainment\\339
39 ['word': 'advocates', 'frequency': 16, 'documents': ['noticias/business\\296.txt', 'noticias/tech\\373.txt', 'noticias/politics\\142.txt', 'noticias/politics\\119.txt', 'n
40 ['word': 'advocated', 'frequency': 1, 'documents': ['noticias/sport\\389.txt']]

```

Entradas iniciais do arquivo full trie.txt

Escolhemos armazenar os dados em um arquivo de texto devido a legibilidade e simplicidade para debug do formato.

Também implementamos a função “*create\_compressed\_trie\_from\_reversed\_index()*”, que faz o caminho reverso da anterior. A função busca pelo arquivo “full\_trie.txt” na pasta “inverted\_index” e reconstrói a Árvore Trie com os dados presentes no arquivo.

## 2.3. Busca

A busca foi implementada em diferentes etapas. A primeira etapa consiste em limpar a expressão de entrada do usuário, convertendo-a para uma expressão válida. Para isso, implementamos o módulo Cleaner. Ele é responsável por fornecer as funções que normalizam os termos de uma expressão. Por exemplo, para a expressão: “.test, OR "patato" OR DuCk AND )(@RED! &%\*\*\$|\ OR or` | 10\$)(“, os caracteres especiais no início e no fim de cada palavra são removidos, resultando em:

test OR patato OR DuCk AND (@RED %\*\*\$| OR or | 10\$)

Depois os caracteres especiais que não são “(”, “)”, “{”, “-”, “.”, “,”, “.” são removidos das palavras. Um termo composto completamente por caracteres especiais é descartado neste ponto:

test OR patato OR DuCk AND (RED OR or | 10)

Então, as letras são transformadas em minúsculas e qualquer espaço extra é removido:

```
test OR patato OR duck AND (red OR or | 10)
```

Por fim, as palavras chaves “AND” e “OR” são convertidas em “&” e “|”, respectivamente:

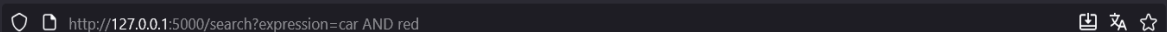
```
test | patato | duck & (red | or | 10)
```

Após limpar a expressão do usuário, nós carregamos em memória a Árvore Trie que foi armazenada em disco com a função “*create\_compressed\_trie\_from\_reversed\_index()*”. Nós utilizamos essa árvore mais os termos buscados para selecionar todos os arquivos que poderiam fazer parte da consulta do usuário, ou seja, todos os arquivos que tenham pelo menos um termo buscado. Com isso, passamos para a terceira etapa, em que filtramos a lista de arquivos anterior com base na expressão fornecida pelo usuário para selecionar os arquivos que efetivamente atendam aos requisitos da expressão booleana informada.

Por fim, com a lista de arquivos filtrada, nós construímos as respostas extraíndo a primeira linha de cada arquivo para ser o título. Depois utilizamos uma expressão regular para encontrar os termos buscados pelo usuário e envolvê-los com a tag “<span/>” para acrescentar o highlight ao snippet. Por último, calculamos o z-score de cada termo e pegamos a média deles.

## 2.4. Endpoint API

Então, implementamos um endpoint para fazer a busca de uma expressão. Ela recebe uma requisição GET e busca nos parâmetros de URL por um chamado “expression”. O valor dele é então passado para a função “*search\_in\_reversed\_index()*”. O resultado de uma requisição à API seria algo do tipo para a consulta “car AND red”:



http://127.0.0.1:5000/search?expression=car AND red

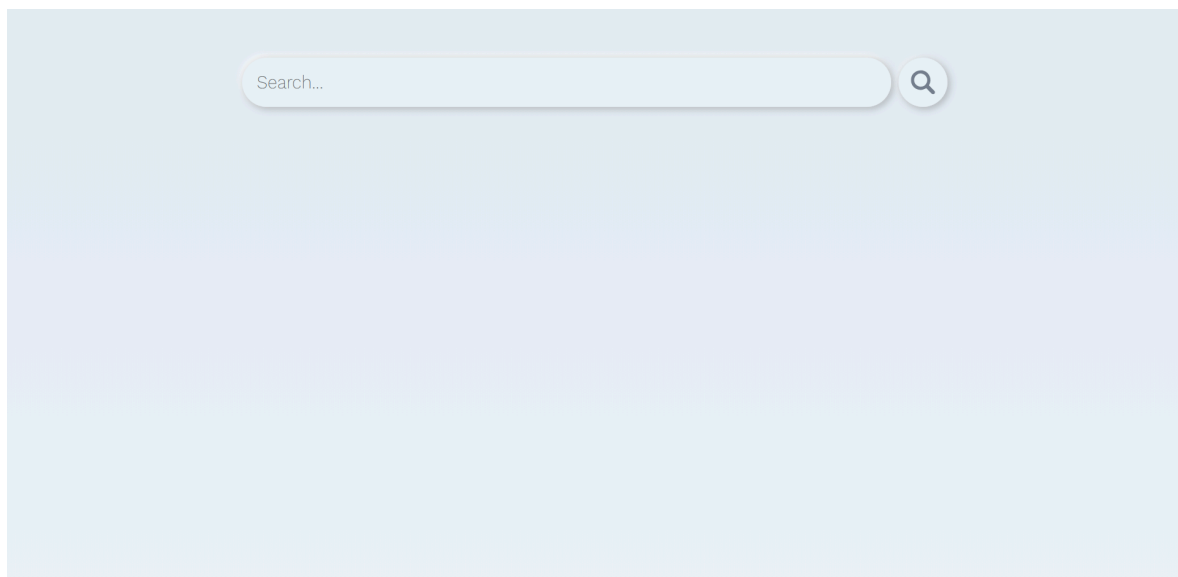
🔍 🌐 ⭐

```
JSON  Dados brutos  Cabeçalhos
Salvar Copiar Recolher tudo Expandir tudo Filtar JSON
▼ 0:
  snippet: 'Stars shine on Bafta <span class="highlight">red</span> carpet\nHollywood stars brought a touch of glamour to London on Saturday for th...'
  title: 'Stars shine on Bafta red carpet'
  z_score: 2.157829841415535
▼ 1:
  snippet: 'Saab to build Cadillacs in Sweden\nGeneral Motors, the world's largest <span class="highlight">car</span> maker, has confirmed that it will build a new medium-sized Cadillac BLS at its...'
  title: 'Saab to build Cadillacs in Sweden'
  z_score: 0.972683867753589
▼ 2:
  snippet: '...ective of the Fiat conglomerate has taken day-to-day control of its struggling <span class="highlight">car</span> business in an effort to turn it around.\nSergio Marchionne has replaced Herber...'
  title: 'Fiat chief takes steering wheel'
  z_score: 0.3582239553874542
▼ 3:
  snippet: '...cy. By the early seventies it was terrorism as well. Al Fatah, Black September, <span class="highlight">Red</span> Brigades, but most of all for us the IRA and the INLA. Thirty more years; 300 p...'
  title: 'Terror powers expose 'tyranny''
  z_score: 0.16471374315568355
▼ 4:
  snippet: '...val could be more than just a lot of hot air drifting up from the Valleys. The <span class="highlight">Red</span> Dragonhood subdued the <span class="highlight">Red</span> Rose Army in most areas of the field, but Henson...'
  title: 'Wales hails new superstar'
  z_score: 0.16471374315568355
▼ 5:
  snippet: '...the 77th Academy Awards, on Sunday.\nA host of stars are expected to grace the <span class="highlight">red</span> carpet outside Los Angeles' Kodak Theatre, including Johnny Depp, Cate Blanchet...'
  title: 'Hollywood ready for Oscars night'
  z_score: 0.16471374315568355
▼ 6:
  snippet: '...omic devastation for those who live there.\n\nThe International Federation of the <span class="highlight">Red</span> Cross and <span class="highlight">Red</span> Crescent Societies told the Reuters news agency that it was seeki...'
  title: 'Giant waves damage S Asia economy'
  z_score: 0.16471374315568355
▼ 7:
  snippet: '...sdually coloured pixels - ranging from blue at the lower end of this scale to <span class="highlight">red</span> at the upper end. Mr Wong says the application is still very much in its infancy...'
  title: 'Blind student 'hears in colour''
  z_score: 0.16471374315568355
▼ 8:
  snippet: '...aturday. An angry David Prutton pushed referee Alan Wiley after being shown the <span class="highlight">red</span> card, but his side still came back to draw 1-1. It was Saints' fourth stalemate...'
  title: 'Brentford v Southampton'
  z_score: 0.16471374315568355
▼ 9:
  snippet: '...pensioner, a patient or a youngster, there was something pulled from Mr Brown's <span class="highlight">red</span> box in an attempt to persuade you to stick with or switch to a New Labour govern...'
  title: 'Brown comes out shooting'
  z_score: 0.16471374315568355
```

Resultado da busca por “car AND red” na API

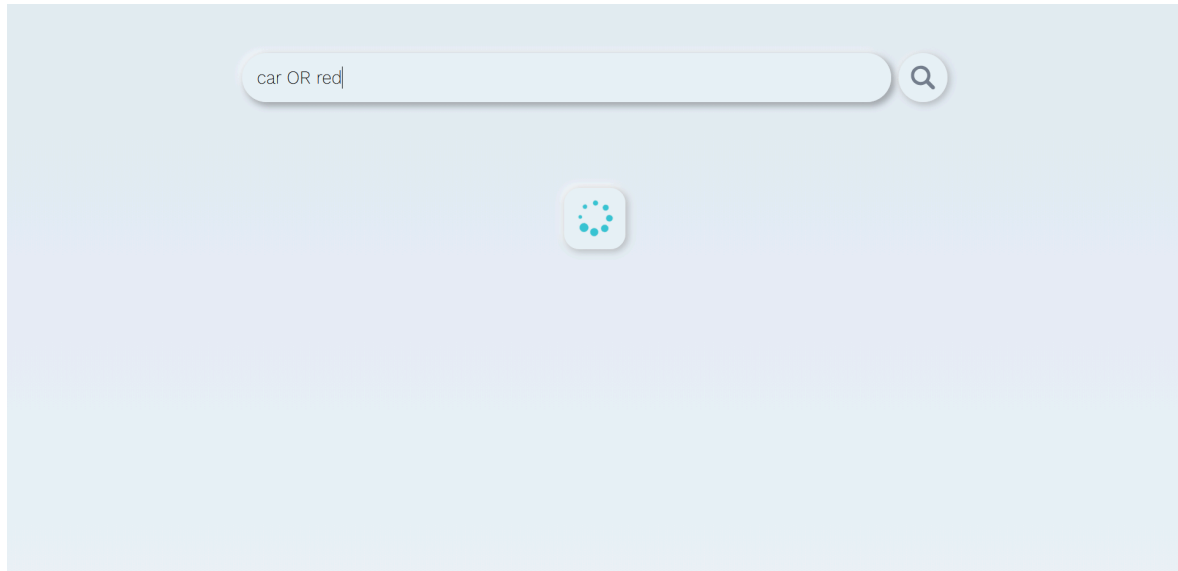
## 2.5. Front-end

Por fim, implementamos o front-end da aplicação. A página inicial segue um design simples e limpo. Ela consiste apenas na caixa de busca e um botão para efetivar a busca:



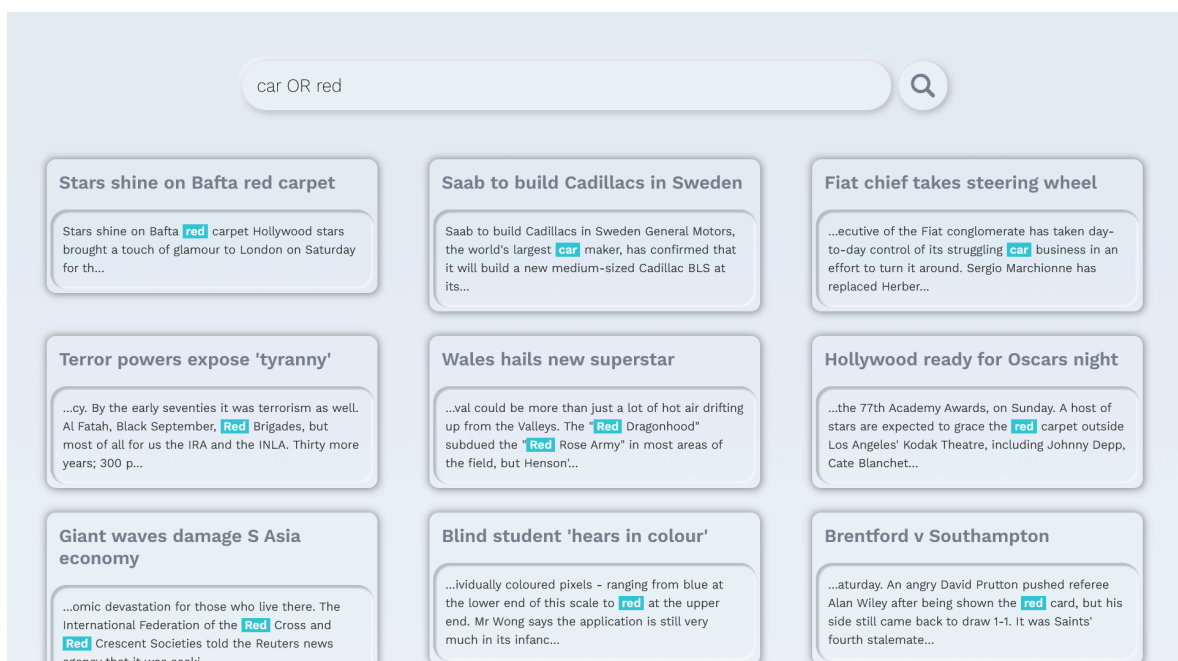
Tela inicial da aplicação

A busca pode ser efetivada tanto utilizando a tecla “ENTER” quanto o botão de busca ao lado direito. Enquanto a busca está sendo realizada, um spinner de carregamento é exibido para o usuário:



Tela de carregamento enquanto a busca está sendo feita

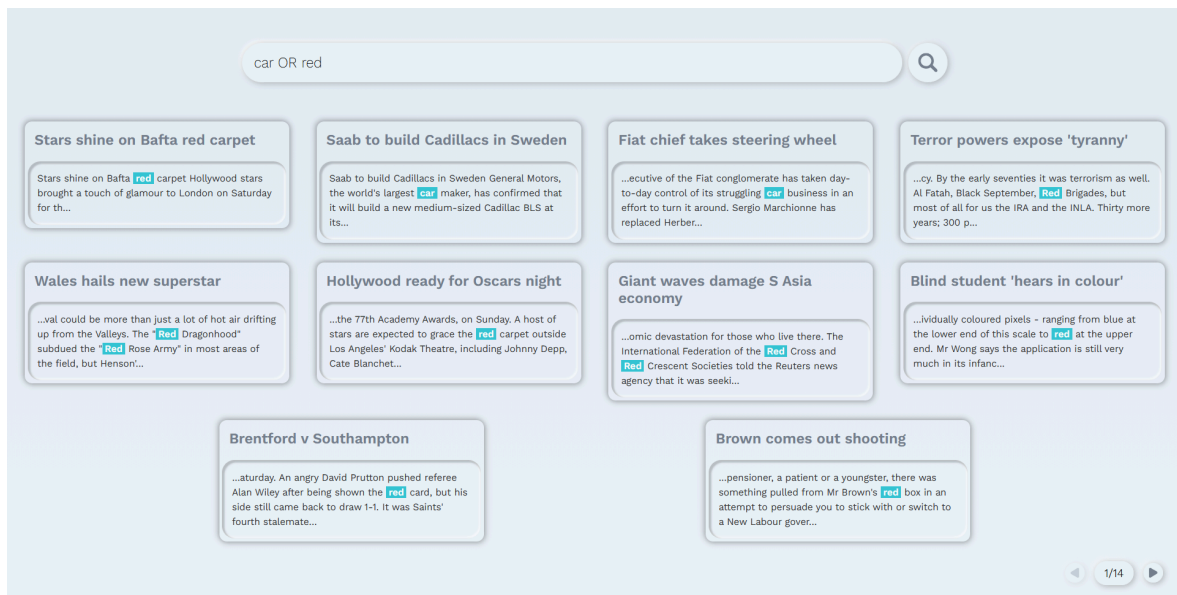
Após o processamento ser concluído, as notícias são exibidas em cards para o usuário com o título em cima e o snippet em baixo. Cada snippet tem marcado, em azul, o(s) termo(s) buscado(s) pelo usuário:



Resultados para a busca dos termos “car OR red”

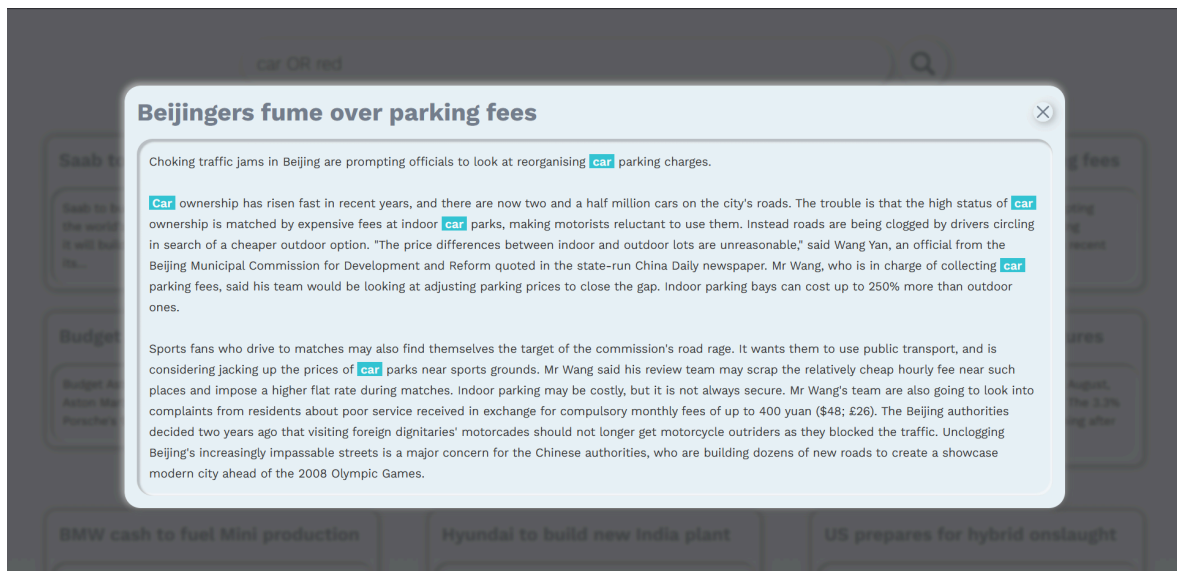


São exibidos um total de 10 cards por página. Ao final de cada página, temos a paginação que permite ver mais resultados:



Tela com zoom reduzido para exibir a primeira página de resultados

Além disso, ao clicar em um dos cards de notícias, um popup com a notícia completa é exibido ao usuário:



Modal com a notícia completa

### 3. Como Executar

- Crie um virtual enviroment: `python -m venv venv`.
- Ative o ambiente virtual:
  - Windows: `./venv/Scripts/activate`.
  - Linux/Mac: `source venv/bin/activate`.
- Instale o flask: `pip install Flask`.
- Rode comando de inicialização: `python -m flask --app ./app.py run`.