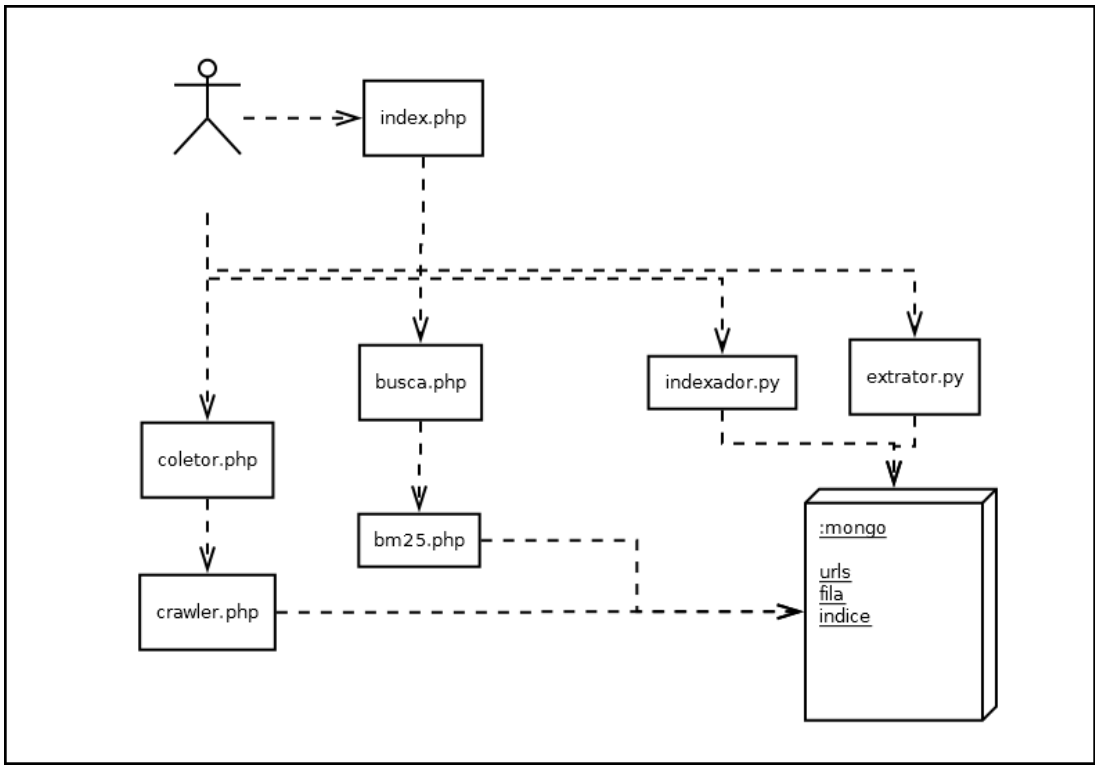
Daniel Reis e Lucas Viana

1- Visão Geral



Arquivos:

- **index.php**: Interface principal do motor de busca
- **busca.php**: Faz o intercâmbio entre a interface e o bm25, processando a consulta do usuário e retornando os resultados
- **bm25.php**: Implementação do modelo BM25
- **coletor.php**: Interface do componente coletor
- **crawler.php**: Componente coletor
- **indexador.py**: Componente indexador
- **extrator.py**: Componente extrator

Pastas:

- **arquivos**: Contém os documentos html coletados
- **css**: Contém as regras de estilo
- **imagens**: Imagens utilizadas na interface do usuário

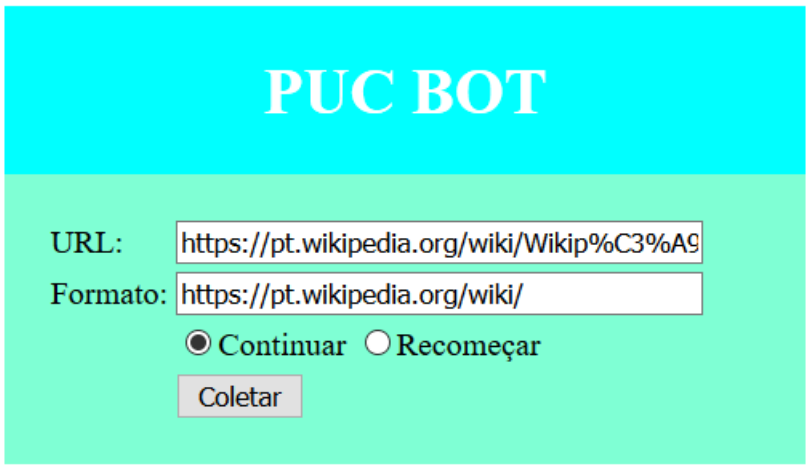
Banco de Dados (mongoDB):

- **fila**: Links encontrados mas ainda não analisados
- **urls**: Páginas coletadas, juntamente com a informação relevante
- **indice**: Ìndice invertido

2- Coletor

O componente coletor foi desenvolvido em PHP e foi dividido em dois arquivos: coletor.php e crawler.php.

O arquivo coletor.php contém uma interface na qual o usuário pode definir os parâmetros da coleta:



No campo “URL” deve-se inserir a url de onde será iniciada a coleta. No campo “Formato” pode-se definir um filtro, restringindo o domínio de onde serão extraídos os documentos. Marcando “Continuar” a coleta partirá dos links já coletados, enquanto que marcando “Recomeçar”, a coleta partirá do link inserido no campo “URL”.

O arquivo crawler.php contém o coletor em si. O algoritmo funciona com base em uma busca em largura. Os links extraídos (por meio de expressões regulares) são inseridos na coleção “fila”. As páginas são baixadas inteiramente e armazenadas na pasta “arquivos”. Finalmente, a url e o lugar onde o documento foi armazenado são inseridos na coleção “url”.

Foram coletados 53.821 documentos com um tempo médio de 1,21 segundos por url.

3- Extrator

O componente extrator foi desenvolvido em Python e está inteiramente contido no arquivo extrator.py. Por meio de expressões regulares, os seguintes atributos são extraídos do documento html:

- **titulo**: título da página
- **categorias**: categorias que a entidade pertence
- **descricao**: pequeno excerto do documento, utilizado na interface do usuário, como um preview do documento
- **texto**: conteúdo relevante da página, processado pelo extrator e indexador para ser utilizado para o ranqueamento
- **length**: quantidade de palavras no texto, também utilizado para o ranqueamento

O processamento do texto implica nas seguintes operações:

- Conversão dos caracteres para caixa baixa
- Remoção de stop words

- Remoção de acentos, caracteres especiais e números
- Stemming

Para a remoção de stop words e stemming foi utilizada a biblioteca Natural Language Toolkit (nltk).

Exemplo:


```
{ "_id" : ObjectId("574aff0752c67328280046ce"), "url" : "https://pt.wikipedia.or
g/wiki/Tuba_uterina", "dir" : "arquivos", "doc" : "574aff0752c67328280046ce.html
", "lastModified" : ISODate("2016-05-29T14:39:20.046Z"), "categorias" : [ "siste
ma reprodutor" ], "texto" : "tub uterin conhec tromp falopi dois tub contrat cm
aproxim estend angul super lateral uter lad pelv tub uterin transport ovul romp
superfic ovari cavidad uter pass direca opost espermatozoid onde habitual ocorr
fecundacaoporca lateral tub uterin possu form assemelh part larg funil orientaca
 inferior poster medial movel fac lateral envolv peritoniositu infundibul istmo
localiz form interpost emergenc ligament redond", "length" : 418, "descricao" :
"As tubas uterinas, também conhecidas por trompas de Falópio, são dois tubos con
tráteis, com 10 cm aproximadamente, que se estendem do ângulo súpero-lateral do
útero para os lados da pelve. As tubas ut...", "titulo" : "Trompas de Falópio -
Wikipédia, a enciclopédia livre" }
```


O extrator conseguiu processar cerca de 48 documentos por segundo.

4- Indexador

O indexador foi desenvolvido em Python e está inteiramente contido no arquivo indexador.py. Ele tem como entrada o atributo “texto” da coleção “urls” e cria o índice invertido na coleção “indice”.

O índice contém os seguintes atributos:

- _id: id do termo
- termo: o termo em si, já processado pelo extrator
- freq: quantidade de documentos que contém o termo (n_i)
- ocorrencias: informações sobre as ocorrencias do termo nos documentos
    1. _id: id do documento cujo termo ocorre
    2. freq: quantas vezes o termo ocorre no documento
    3. pos: posições em que o termo ocorre


```
> db.indice.find().skip(500).limit(1)
{ "_id" : ObjectId("5724cf50152fbc0858381847"), "termo" : "abraa", "ocorrencias"
 : [ { "pos" : [ 26 ], "doc" : ObjectId("56ecc27e52c673581f005b04"), "freq" : 1
}, { "pos" : [ 2144, 2161, 2165, 2184, 2188, 2210, 2227, 2237, 2272, 2291 ], "do
c" : ObjectId("56ed501952c673840f00f896"), "freq" : 10 }, { "pos" : [ 5162 ], "d
oc" : ObjectId("56ed502e52c673840f010883"), "freq" : 1 }, { "pos" : [ 849 ], "do
c" : ObjectId("56ed532052c673840f01b6d9"), "freq" : 1 }, { "pos" : [ 1009 ], "do
c" : ObjectId("56ed533652c673840f01bd3f"), "freq" : 1 }, { "pos" : [ 1044 ], "do
c" : ObjectId("56ed533b52c673840f01bfdd"), "freq" : 1 }, { "pos" : [ 333 ], "doc
" : ObjectId("56ed534352c673840f01c5cc"), "freq" : 1 }, { "pos" : [ 349 ], "doc"
 : ObjectId("56ed567952c673840f01d258"), "freq" : 1 }, { "pos" : [ 112 ], "doc"
: ObjectId("56ed569752c673840f01de27"), "freq" : 1 }, { "pos" : [ 66, 1153, 3118
 ], "doc" : ObjectId("56ed56b352c673840f01f26f"), "freq" : 3 }, { "pos" : [ 300
], "doc" : ObjectId("56ed56b952c673840f01f79f"), "freq" : 1 }, { "pos" : [ 218 ]
, "doc" : ObjectId("56ed62cc52c673401f0032c0"), "freq" : 1 }, { "pos" : [ 7 ], "
doc" : ObjectId("56ed65be52c673840f059950"), "freq" : 1 }, { "pos" : [ 7 ], "doc
" : ObjectId("56ed65be52c673483c000138"), "freq" : 1 }, { "pos" : [ 7 ], "doc" :
 ObjectId("56ed65be52c673401f00cc4a"), "freq" : 1 }, { "pos" : [ 42 ], "doc" : O
```

Sendo indexados 26.850 documentos, o índice invertido continha 202.006 registros e ocupava 384,23MB. O tempo de processamento variou entre 1-4 segundos por documento, sendo em média 2,83 segundos.

5- Ranking

O modelo de recuperação de informação implementado pelo motor de busca é o BM25 puro, contido no arquivo bm25.php. O arquivo busca.php faz o intermédio entre a interface e o modelo, processando a consulta do usuário (stemming, caixa baixa e remoção de caracteres especiais) e retornando o resultado, em formato JSON.

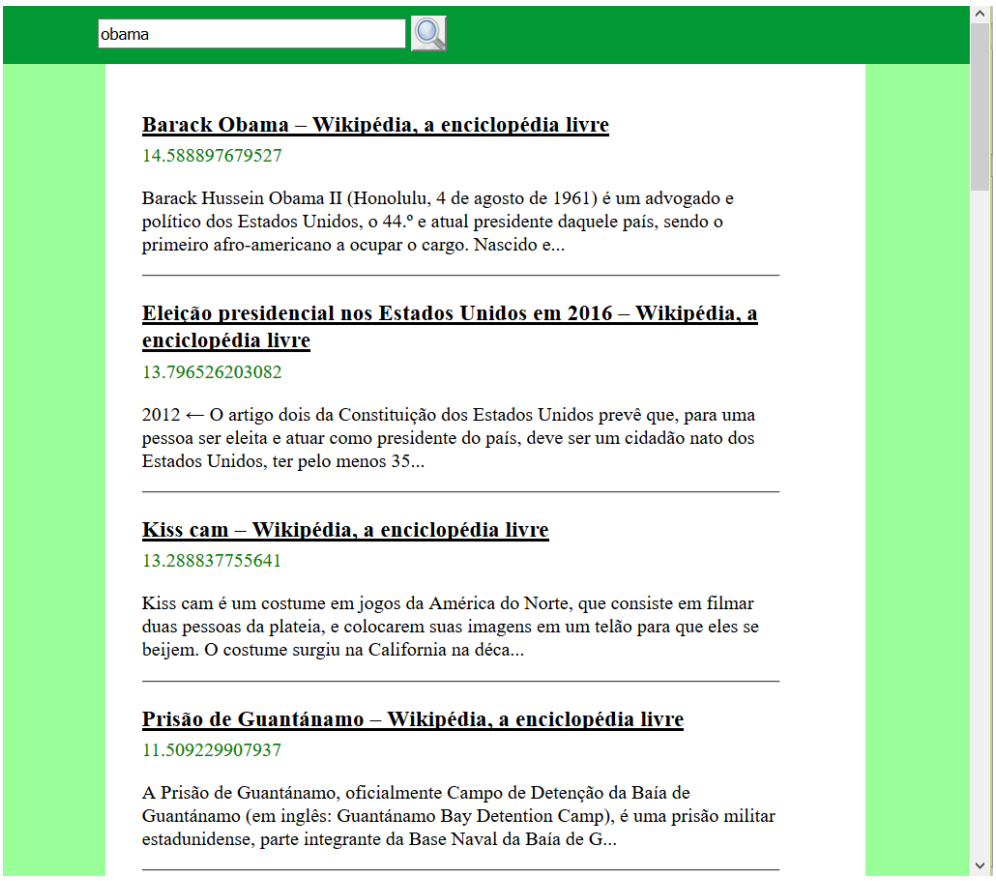
O modelo utiliza como parâmetros k=1 e b=0.75.

O arquivo busca.php recebe a consulta por meio de uma requisição GET, contendo a variável “busca”. O resultado é repassado para a interface no formato JSON contendo os seguintes atributos:

- url: link da página
- titulo: título da página
- descricao: pequeno excerto do texto
- ranking: valor da similaridade calculada para o documento

6- Interface

A interface está contida no arquivo index.php. A consulta do usuário é inserida na barra que fica no topo da página. Após o usuário aperta o botão de pesquisa, uma função do JQuery realiza uma requisição GET para o arquivo busca.php. Em seguida, o resultado é processado pela função mostrarResultados() e os 20 documentos mais relevantes são exibidos na interface:



7- Outras Informações

O site pode ser acessado pelo endereço http://192.99.58.107