UNIVERSIDADE FEDERAL DO AMAZONAS
INSTITUTO DE COMPUTAÇÃO
BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO

Identificação de Eventos Culturais em Páginas Web Relacionados com Assuntos Estudados por Alunos do Ensino Fundamental e Médio

Pedro Henrique dos Santos Iunes

Manaus – AM
Março de 2023

Pedro Henrique dos Santos Iunes

Identificação de Eventos Culturais em Páginas Web Relacionados com Assuntos Estudados por Alunos do Ensino Fundamental e Médio

Monografia de Graduação apresentada à Coordenação de Engenharia da Computação, UFAM, da Universidade Federal do Amazonas, como parte dos requisitos necessários à obtenção do título de Engenheiro da Computação.

Orientador(a)

Raimundo da Silva Barreto, Dr.

Universidade Federal do Amazonas – UFAM

Faculdade de Tecnologia - FT

Manaus-AM

Março de 2023

Ficha Catalográfica

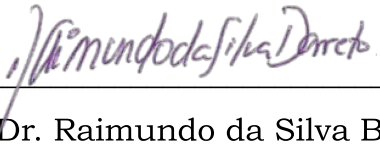
Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Iunes, Pedro Henrique dos Santos
I92i Identificação de eventos culturais em páginas web relacionados com assuntos estudados por alunos do ensino fundamental e médio / Pedro Henrique dos Santos Iunes . 2023
31 f.: il. color; 31 cm.

Orientador: Raimundo da Silva Barreto
TCC de Graduação (Engenharia da Computação) - Universidade Federal do Amazonas.

1. Web scraping. 2. Eventos. 3. Python. 4. Ensino. I. Barreto, Raimundo da Silva. II. Universidade Federal do Amazonas III. Título

Monografia de Graduação sob o título Identificação de Eventos Culturais em Páginas Web Relacionados com Assuntos Estudados por Alunos do Ensino Fundamental e Médio apresentada por Pedro Henrique dos Santos Iunes e aceita pelo corpo docente do curso de engenharia da computação da Universidade Federal do Amazonas, sendo aprovada por todos os membros da banca examinadora abaixo especificada:

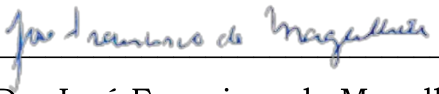


Prof. Dr. Raimundo da Silva Barreto

Orientador

IComp

UFAM



Prof. Dr. José Francisco de Magalhães Netto

IComp

UFAM

Prof. Dr. Ruiter Braga Caldas

IComp

UFAM

Manaus-AM, 03 de Março de 2023.

Dedico esse trabalho a minha amada esposa que não me deixou desistir e esteve sempre ao meu lado.

Agradecimentos

Ao longo deste anos de curso encontrei várias oportunidades que serviram para o meu desenvolvimento pessoal além de somente profissional, oportunidades estas que me foram concedidas primeiramente por Deus que me abençoou com a vida, a saúde e sabedoria necessária para chegar até o final e que foi o meu escape em momentos que me encontrei em angústia. Meus pais Antonio Jorge dos Meirelles Iunes e Cassia Cristina dos Santos Iunes me deram todo o amor, suporte e educação necessária para a formação do meu caráter que me deu a possibilidade de estar aqui e a minha esposa, Amanda Carolina Rocha Iunes que em vários momentos foi meu suporte nesse longo caminho.

Agradeço ao meu orientador, o prof. Dr. Raimundo Barreto, por me ajudar a melhorar a concepção da estrutura que seria utilizada ao desenvolver este trabalho e pelas conversas que me motivaram na continuidade do mesmo.

Agradeço a cada professor que tive ao cursar as disciplinas da graduação pelo fato de terem compartilhado de seus conhecimentos, isso foi algo de muita valia para mim. Agradeço também aos colegas que tive o prazer de conviver ao cursar esta graduação, cada experiência que passamos juntos contribuiu bastante para o meu crescimento. Durante o curso de Engenharia da Computação na UFAM, pude sentir um ambiente que proporcionou um acolhimento e respeito entre os alunos.

Identificação de Eventos Culturais em Páginas Web Relacionados com Assuntos Estudados por Alunos do Ensino Fundamental e Médio

Autor: Pedro Henrique dos Santos Iunes
Orientador: Raimundo da Silva Barreto, Dr.

Resumo

Com a virtualização do ensino, em especial nas bases fundamental e médio, que é uma das características do chamado "novo normal" pós pandêmico, a dificuldade dos professores em proporcionar novas experiências de aprendizagem além das aulas expositivas, e a dificuldade dos alunos em achar conteúdos em sites de eventos que sejam relevantes para as matérias que eles aprenderam em sala de aula. Este trabalho desenvolve uma plataforma que auxilia alunos e professores a encontrar eventos onlines ou presenciais que possuam alguma relação com as matérias expostas nas aulas através de um filtro. A plataforma consiste em uma aplicação frontend que realiza requisições a uma API Rest responsável pela obtenção das informações dos eventos nos sites *https://www.sympla.com.br* e *http://www.feirasdobrasil.com.br*. O diferencial no desenvolvimento deste sistema é a utilização de web scraping para a obtenção das informações dos eventos, ou seja, usasse a técnica de crawlers para percorrer as páginas e subpáginas dos sites propostos e coletar as informações necessárias de um evento. Desta forma os usuários têm sempre as informações atualizadas dos eventos não ocorrendo risco de redundância ou informações desatualizadas.

Palavras-chave: Web Scraping, Eventos, Python, Ensino.

Identification of Cultural Events on Web Pages Related to Subjects Studied by Students Elementary and high school

Autor: Pedro Henrique dos Santos Iunes
Orientador: Raimundo da Silva Barreto, Dr.

ABSTRACT

With the virtualization of teaching, especially in primary and secondary education, which is one of the characteristics of the so-called post-pandemic "new normal", the difficulty for teachers to provide new learning experiences beyond lectures, and the difficulty for students to find content on event websites that is relevant to the subjects they learned in the classroom. This work develops a platform that helps students and teachers to find online or face-to-face events that have some relation with the subjects exposed in the classes through a filter. The platform consists of a frontend application that makes requests to a Rest API responsible for obtaining event information on the websites https://www.sympla.com.br and http://www.feirasdobrasil.com.br. The difference in the development of this system is the use of web scraping to obtain event information, that is, using the crawlers technique to go through the pages and subpages of the proposed sites and collect the necessary information about an event. In this way, users always have up-to-date information on events, with no risk of redundancy or outdated information.

Keywords: Web Scraping, Events, Python, Teaching.

Lista de figuras

Lista de abreviaturas e siglas

UFAM – Universidade Federal do Amazonas

IComp – Instituto de Computação

Sumário

1 Introdução

1.1 Contexto

Em 2020, o mundo passou por uma pandemia, devido ao vírus SARS-CoV-2, popularmente conhecido como Coronavírus ou COVID19. Um vírus altamente contagioso e letal, que se espalhou pelo mundo inteiro causando milhões de mortes e sequelas em alguns dos sobreviventes. De início, pensava-se que essa doença não seria tão grave e que logo passaria, assim como outras que já existiram, mas não foi o caso da COVID 19, ela veio para ficar e mudar completamente as relações humanas da modernidade. Hoje já temos vacinas eficazes para neutralizar os efeitos graves da doença, mas o perigo ainda está longe de desaparecer. O distanciamento social, uso de máscaras e álcool 70% em gel, são um dos sinais a mostrar que nossa vida em sociedade mudou e que nada mais será como antes.

Aconteceu também que toda a interação social, teve que ser, em grande parte, virtualizada, em especial nos picos de casos com mortes no Brasil de 2020 e 2021. E essa virtualização foi em todos os lugares, encontro de família e amigos, trabalho e até mesmo as escolas, que ficaram por um momento sem aula alguma até iniciarem a adaptação para o sistema digital de ensino. As salas de aula foram substituídas por salas virtuais de bate-papo com vídeo. O que, claro foi um transtorno tanto para o professor, que foi obrigado a aprender a ser editor de vídeo e buscar diversas novas ferramentas de estímulo ao ensino, pensando nos diversos estilos de aprendizagens que existem, de forma online, quanto para os alunos que, muitos deles, em especial os de ensino público, às vezes nem mesmo acesso a internet ou dispositivos eletrônicos tinham.

1.2 Definição do problema

Com a virtualização do ensino, em especial nas bases fundamental e médio, que é uma das características do chamado "novo normal" pós

pandêmico, a dificuldade dos professores em proporcionar novas experiências de aprendizagem além das aulas expositivas, e a dificuldade dos alunos em achar conteúdos em sites de eventos que sejam relevantes para as matérias que eles aprenderam em sala de aula, surgiu uma incógnita: seria possível auxiliar os professores e alunos a encontrar eventos culturais online e, quando possível, presenciais, correlatos com os assuntos estudados por eles através de um sistema de filtro de eventos? Facilitando assim o encontro desses eventos e a fixação da matéria estudada?

Com essa possibilidade em mente, esse sistema será focado, a princípio, para eventos que estivessem registrados somente nas duas plataformas a seguir o https://www.sympla.com.br e o http://www.feirasdobrasil.com.br.

1.3 Justificativa

Segundo teoria desenvolvida por Fernald e Keller e Orton-Gillingham [5], há três estilos de aprendizado: visual, auditivo e cinestésico. Os alunos visuais são como "memória fotográfica" porque conseguem memorizar matérias, nomes e dados com mais facilidade quando a visão é estimulada. Os alunos auditivos precisam de silêncio e concentração para ouvir a explicação dos professores e, mais tarde, repassar o conteúdo em voz alta durante os estudos. Neste caso, os livros e documentários em áudio, além de podcasts, podem ajudar bastante. Já os alunos que adotam o modo de aprender cinestésico são mais no estilo "mão na massa" e que gostam de estímulos táteis. Para estes alunos, não basta apenas ler ou ouvir as explicações do professor, mas precisam entender o conteúdo na prática, metendo a mão na massa e experimentando.

E tendo em vista o momento atual, e até mesmo para o futuro, alcançar com facilidade eventos educativos na cidade como uma forma de auxiliar o ensino do professor, é necessário e que atinge um bem tanto para o educador como principalmente para o estudante. Essa ferramenta de filtragem de informações pretende ser de grande apoio no ensino no "novo normal".

1.4 Objetivo Geral

É fazer a identificação de eventos culturais em páginas *Web*, tais como: feiras, exposições, mostras, museus, peças teatrais, etc, que estejam acessíveis online, ou na cidade do aluno, e que possuam alguma relação com assuntos estudados por alunos de uma escola de ensino fundamental e médio.

1.5 Objetivos Específicos

O projeto consiste em atingir os seguintes objetivos:

a) Prover uma interface de fácil entendimento, para ambos os tipos de persona no caso estudante e professor;

b) Proporcionar um serviço cujo propósito é de ser o servidor para a interface que interage com os usuários, ou seja, este serviço deve receber requisições e passar a tarefa necessárias para os bots de web scraping;

c) Proporcionar bots capazes de pesquisar em ambos os sites definidos sem que falte informações de ambos;

d) Tornar possível que os sistemas desenvolvidos sejam acessíveis de forma online pelos usuários.

1.6 Organização do Trabalho

O documento está organizado como segue:

O Capítulo 1 apresenta os conceitos básicos que envolvem o objetivo deste trabalho e quais são os objetivos a serem alcançados. No Capítulo 2 é analisada as técnicas de ensino e as ferramentas de catalogação de informação web. Entre elas é aprofundado no *Web Scraping*. O Capítulo 3 descreve o desenvolvimento do *software*. O Capítulo 4 apresenta os resultados obtidos. E por fim, no Capítulo 5 são destacadas as considerações finais deste trabalho e sugestões para trabalhos futuros.

2 Fundamentos Teóricos

2.1 *Web Scraping*

Web scraping nada mais é do que a coleta de dados na internet, contudo sem a interação humana ou com uma API pública, como bem explica Mitchell no seu livro *Web Scraping* com *Python Web Scraping* [3]:

> "*web scraping" é a* prática de coletar dados por qualquer meio que não seja um programa interagindo com uma API (ou, obviamente, por um ser humano usando um navegador web). Isso é comumente feito escrevendo um programa automatizado que consulta um servidor web, requisita dados (em geral, na forma de HTML e de outros arquivos que compõem as páginas web) e então faz parse desses dados para extrair as informações necessárias."

Essa ferramenta é muito útil pois ela coleta dados além do que a busca no google nos apresenta, por exemplo, de maneira mais específica. Apesar de ser algo não tanto difundido no Brasil, por enquanto, diversos autores, estudantes e profissionais já estão se aproveitando dessa ferramenta para coleta de dados volumosos.

O *web scraping* não é algo novo, como explica Mitchell Web Scraping [3] esse tipo de coleta de dados é tão antigo quanto a própria internet, contudo ele tinha outros nomes:

> "Embora *web scraping* não seja um termo novo, no passado, a prática era mais conhecida como *screen scraping*, portanto esse é o termo que uso neste livro, embora também possa me referir aos programas que especificamente percorrem várias páginas como *web crawlers* ou aos próprios programas de *web scraping* como *bots*."

A utilização do *web scraping* se mostra necessária quando se pretende fazer uma pesquisa mais aprofundada sobre dados específicos disponíveis

na web, contudo, estes só poderiam ser achados através de meios limitados no que diz respeito ao número e tipos de análises a serem feitas. Nesse caso, por essa ferramenta ter esse alcance preciso e não limitado, ela se mostra a melhor maneira de se coletar dados de forma assertiva, específica e precisa.

2.2 Estilos de Aprendizado

Ensinar e facilitar o acesso ao conhecimento não é uma tarefa fácil, e nem tem um caminho apenas. Diferentes alunos, diferentes formas de ensinar e aprender. E a questão do aprendizado também não fica apenas preso às quatro paredes de uma sala de aula e tendo sua fonte em apenas uma pessoa de pé em frente de uma lousa. Mas em toda parte o estudante pode ter a oportunidade de aprender, especialmente no momento atual onde jamais foi tão fácil ter acesso a informação e conhecimento.

Na jornada do aprendizado existem vários modos de você reter o conhecimento, o psicólogo Carl Rogers acreditava e defendia que a facilitação da aprendizagem é o objetivo mor da educação. Ele propunha uma série de princípios de aprendizagem, baseado nos princípios de terapia centrada no cliente, como ele preferia chamar seus pacientes. Ele defendia, entre outras coisas, uma aprendizagem significante, ou seja, que gerasse um significado, uma relevância na vida do que aprende, para que de fato se torne algo retido de forma duradoura. De acordo com MOREIRA, Marco Antônio no seu livro Teorias de Aprendizagem [4]:

> "A aprendizagem significante é, para Rogers, mais do que uma acumulação de fatos. É uma aprendizagem que provoca uma modificação, quer seja no comportamento do indivíduo, na orientação da ação futura que escolhe, ou nas suas atitudes e na personalidade."

E além de a aprendizagem ter uma significância para o indivíduo, ela também tem estilos próprios de ser alcançada, como Fernald, Keller e Orton-Gillingham [5] explanam na sua teoria VAC (Visual, Auditivo e Cinestésico), em que eles pressupõe que a aprendizagem ocorre através dos sentidos, seja ele o visual, o auditivo ou o tátil. Trazendo uma lupa a este

assunto, cada um tem uma explicação específica para definir a sua classificação. Por exemplo, os indivíduos que tem o estilo de aprendizagem mais visual, são aqueles estudantes que assimilam estímulos recebidos visualmente de forma mais rápida, e conseguem fazer associações entre ideias e abstrair conceitos através de algo visual, seja isso uma ilustração, um mapa mental, um vídeo, um filme, etc. Por outro lado, os indivíduos do estilo auditivo de aprendizagem, são mais propensos a reter mais informações através da linguagem falada, através de, por exemplo, palestras, *podcasts*, músicas, etc. E por fim, o estudante cinestésico, precisa da questão sensorial para absorver o conhecimento, ou seja, ele aprende na prática. Pessoas que aprendem mais facilmente através desse estilo, não vão reter tanto algum ensinamento só ouvindo ou vendo, ela precisa fazer, construir, tocar, desenvolver, no linguajar mais popular "por a mão na massa".

Facilitar a obtenção de informação e a aprendizagem adequando aos diferentes estilos de cada pessoa, é algo que deve ser estimulado e desenvolvido atualmente. Quebrar os preconceitos de como é o "jeito certo" de aprender e passar o conhecimento é uma das propostas no presente projeto.

2.3 Python

Outra parte importante para o desenvolvimento da tecnologia de busca proposta neste trabalho, é o conhecimento de *Python*. Essa é uma linguagem de altíssimo nível orientada a objetos, de tipagem dinâmica e forte, interpretada e interativa [1]. Segundo Mark em seu livro Aprendendo *python* [2], esta "é comumente definida como uma linguagem *de script* orientada a objetos - uma definição que combina suporte para POO com orientação global voltada para funções de script." A Figura 1 mostra a logo da linguagem.

Figura 1 - Logo da linguagem Python.

Essa linguagem foi lançada por Guido van Rossum em 1991, mas atualmente possui um modelo de desenvolvimento comunitário, aberto e gerenciado pela organização sem fins lucrativos *Python Software Foundation*. Apesar de várias partes da linguagem possuírem padrões e especificações formais, a linguagem, como um todo, não é formalmente especificada. O padrão na prática é a implementação *CPython*.

Ela tem uma linguagem muito versátil de sintaxe simples, e que adota uma estratégia minimalista, pois há sempre uma maneira óbvia de se executar uma tarefa, embora ainda exista outras menos óbvias, mas por tendência, o programador vai na mais intuitiva, facilitando assim o trabalho e tornando o desenvolvimento nessa linguagem um tanto divertida.

Uma das coisas que mais chama atenção nessa linguagem é seu poder de tornar tudo mais simples, por mais complexo que seja.

"O *Python* permite que os programas sejam desenvolvidos muito mais rapidamente do que nas linguagens compiladas, como *C++*. Seu rápido ciclo de desenvolvimento promove um modo de programação exploratório e incremental que precisa ser experimentado para ser apreciado." Mark (2007, p.32).

A principal biblioteca Python que será usada no projeto é a Beautiful Soup. De acordo com a documentação a Beautiful Soup [8] “É uma biblioteca Python de extração de dados de arquivos HTML e XML. Ela funciona com o seu interpretador (parser) a fim de prover maneiras mais intuitivas de navegar, buscar e modificar uma árvore de análise (parse tree)”. Utilizaremos ela na versão 4.0.

2.4 Flask

Junto com as bibliotecas do python utiliza-se a *framework flask*. Ela é uma framework de aplicação Web escrita em Python. Foi desenvolvido por Armin Ronacher [10][11], que liderou uma equipe internacional de entusiastas do Python chamada *Poocco*. O *Flask* é baseado no *Werkzeg WSGI toolkit* e no *Jinja2 template engine*. Ambos são projetos da *Pocco*. A Figura 2 mostra a logo da *framework flask*.

Figura 2 - Logo da *framework* Flask.

Mais precisamente o flask é considerado um *microframework* porque não requer ferramentas ou bibliotecas particulares, mantendo um núcleo simples, porém, extensível. Isso quer dizer que o flask oferece suporte a extensões que podem adicionar recursos a aplicação como se fossem implementados no próprio Flask.

2.5 Docker

Docker é um conjunto de produtos de plataforma como serviço (PaaS) que usam virtualização de nível de sistema operacional para entregar software em pacotes chamados contêineres [12]. Assim, o Docker torna operações em uma infraestrutura como serviços web mais intercambiáveis, eficientes e flexíveis. Um contêiner é um pacote de virtualização de um sistema linux pronto para executar uma aplicação. Esses pacotes são leves, portáteis e autossuficientes. Os contêineres são isolados uns dos outros e agrupam seus próprios softwares, bibliotecas e arquivos de configuração. Eles podem se comunicar uns com os outros por meio de canais bem definidos. Todos os contêineres são executados por um único kernel do sistema operacional e, portanto, usam menos recursos do que as máquinas

virtuais. A Figura 3 mostra a logo dessa plataforma e a Figura 4 ilustra um exemplo de código.

Figura 3 - Logo da plataforma Docker.

```
Dockerfile > ...
1   FROM python:3.9.4
2   ENV FLASK_ENV = production
3   WORKDIR /app
4   RUN pip install --upgrade pip
5   COPY ./requirements.txt .
6   RUN pip install -r requirements.txt
7   ADD . /app
8   EXPOSE 8000
```

Figura 4 - Código exemplo de *dockerfile.*

2.6 Angular

Angular ou Angular 2 é uma framework para aplicações web voltada para o desenvolvimento front-end, O mesmo tem o código-fonte aberto e tem como linguagem base o TypeScript. A Figura 5 mostra a logo da *framework* Angular.

Figura 5 - Logo da *framework* Angular.

Segundo o site oficial dessa framework [14] ela possui uma estrutura baseada em componentes para criar aplicações Web escaláveis e uma coleção de bibliotecas bem integradas que cobrem uma ampla variedade de recursos como:

- Gerenciamento de rotas para as páginas,
- Gerenciamento de formulários,
- Comunicação cliente-servidor e etc.

Uma das estruturas base do angular e que permitem maior escalabilidade são os componentes. Eles são estruturas que encapsulam o código Typescript, o html e o css de algum objeto da página como um botão ou até mesmo a própria página inteira.

```
import { Component } from '@angular/core';

@Component({
  selector: 'hello-world',
  template: `
    <h2>Hello World</h2>
    <p>This is my first component!</p>
  `
})
export class HelloWorldComponent {
  // The code in this class drives the component's behavior.
}
```

Figura 6 - Exemplo de código angular.

E podem ser chamados por qualquer outro componente por meio do seu seletor. O seletor nesse exemplo seria uma tag html *<hello-world></hello-world>*

3 DESENVOLVIMENTO DO PROJETO

Neste capítulo será apresentado o escopo deste trabalho. Serão apresentados a metodologia utilizada para atender as especificações, o projeto do sistema final e dos subsistemas envolvidos e o desenvolvimento destas diversas etapas deste projeto.

3.1 Metodologia de Desenvolvimento

Para atender a lista de objetivos específicos propostos na Seção 1.4, o desenvolvimento do projeto se deu pelas seguintes etapas:

a) Elaboração de um diagrama de atividades onde são representados o fluxo de interação do backend com o frontend do sistema;
b) Definição da estratégia de web scraping para os sites selecionados;
c) Criação dos controles de web scraping para os sites

selecionados;

d) Elaboração de uma API REST e a rota para requisição do web scraping;
e) Desenvolvimento da aplicação frontend do sistema utilizando a framework Angular;
f) Mockup das telas de interface de usuário para o sistema web;
g) Comunicação entre o frontend desenvolvido e o endpoint criado na API REST;
h) Testes funcionais e de integração do sistema;
i) Configurar um dockerfile e docker-compose para a aplicações front e back;

3.2 Projeto do Sistema Geral

O projeto a ser desenvolvido pode ser dividido em três camadas, o *frontend* para a apresentação e coleta de informações para a pesquisa, o *backend* para o qual o *frontend* realiza requisições e o web scraping para a consulta das informações nos sites "https://www.sympla.com.br/" e "https://www.feirasdobrasil.com.br". As camadas funcionam da seguinte forma: o usuário através da interface de usuário do *frontend* realiza requisições http ao *backend* do sistema, que por sua vez repassa as informações pesquisadas para o controlador de *web scraping* que consulta os sites cadastrados atrás da informação pesquisada.

3.3 Diagrama de Atividades do Sistema

Em UML, para determinar e modelar partes do comportamento de um software são feitos diagramas de atividades, ele é essencialmente um gráfico de fluxo, mostrando o fluxo de controle de uma atividade para outra. Ilustrando graficamente como será o funcionamento do sistema, como se dará a execução de suas partes e como atuará no contexto em que está inserido.

Esta ferramenta UML neste trabalho foi aplicada da seguinte forma: foram definidas duas raias, a do frontend do sistema e o backend que responde às suas requisições, nesta representação, o backend é composto pela api rest e pelo scrapings que serão executados nos sites que foram definidos para este projeto, exibidos na Figura 1. Abaixo das definições de cada uma delas estão as atividades que realizam no sistema e que vão se sucedendo em um fluxo lógico à medida em que um usuário interage com o sistema. A Figura 2 exibe o diagrama de atividades deste projeto.

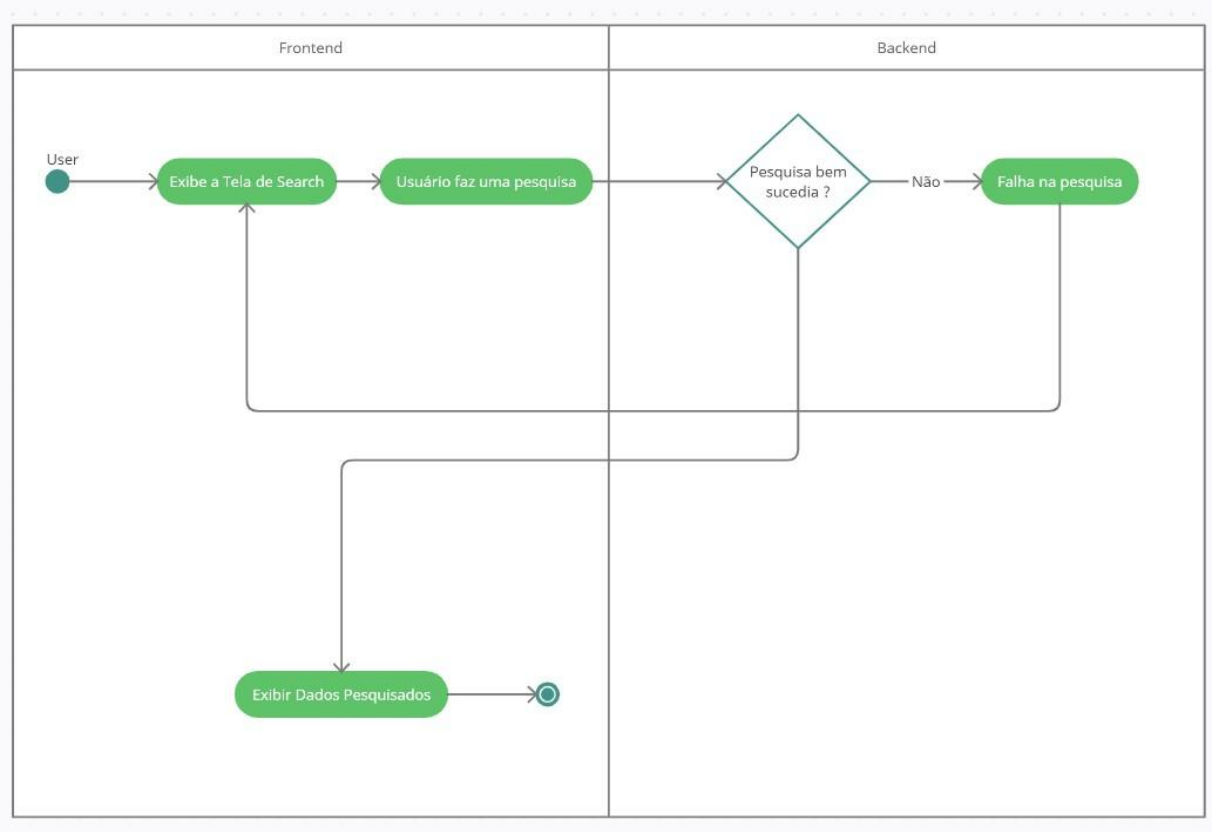


Figura 7 - Diagrama de atividades do sistema.

3.4 Projeto do Módulo de web scraping

O módulo de web scraping para esse projeto foi desenvolvido utilizando a linguagem python e a biblioteca *BeautifulSoup* [8]. Para diferença e organizar melhor as funções foi criado dois arquivos com as funções para fazer o coletar os dados dos eventos de dois sites diferentes sendo eles o Feiras do Brasil

(https://www.feirasdobrasil.com.br/feirasdasemana.asp) e Sympla (https://www.sympla.com.br/)

Nesses dois arquivos foram criadas funções para fazer a coleta correta de cada site proposto, pois cada site possui sua própria estrutura para apresentar as informações, então fazendo-se a necessidade de funções específicas para cada um.

3.5 Projeto da API Rest

Para o projeto da API Rest que recebe requisições do aplicação frontend, que foi desenvolvido na linguagem python com a framework Flask, uma estrutura de diretórios foi planejada a fim de organizar os arquivos dos códigos fontes do projeto de forma componentizada, como é mostrado na Figura 7.

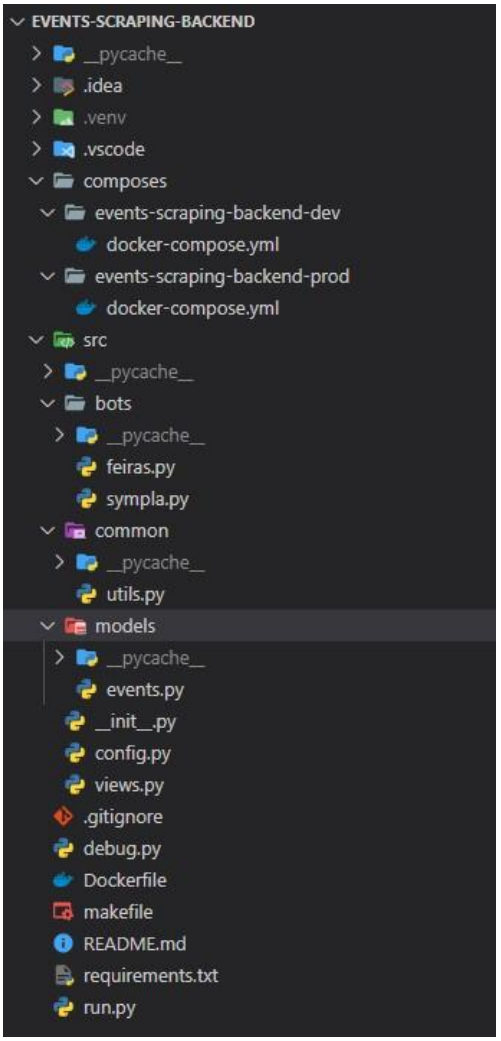


Figura 8 - Estrutura de diretórios do backend.

A seguir são explanadas as funções de cada diretório e dos arquivos pelos quais é composto.

- run.py: Arquivo utilizado para iniciar o sistema em ambiente de produção.
- debug.py: Arquivo utilizado para iniciar o sistema em ambiente de desenvolvimento.
- Dockerfile: Arquivo necessário para a configuração e criação da imagem do docker que será utilizada no container.
- composes: Neste diretório se encontram os arquivos composes responsáveis pela criação do container a partir da imagem feita pelo Dockerfile.
- bots: Neste diretório se encontram os arquivos dos *crawlers* responsáveis pelo scraping das páginas definidas anteriormente.
    - feiras.py: Aqui está o módulo de scraping para o site Feira do Brasil que está com as funções para garantir a coleta correta do site.
    - sympla.py: Aqui está o módulo de scraping para o site Sympla que está com as funções para garantir a coleta correta do site.
- common/utils.py: Arquivo que contém funções utilitárias que são chamadas ao longo do projeto.
- models: diretório que contém os modelos de dados utilizados pelo sistema.
    - events.py: Abstração das principais características de um evento como nome, local, data, tipo de evento e url.
- __init__.py: Arquivo utilizado para criar os objetos necessários para inicializar o flask.
- config.py: Arquivo que contém as constantes necessárias para configurar o sistema corretamente.
- views.py: Arquivo com as declarações das rotas de requisição necessárias para executar o sistema.

Após a definição da estrutura de diretórios e arquivos do backend, temos a rota principal do sistema que seria:

- ‘search’: Rota do tipo POST que tem como payload esperado um JSON com o atributo *name* e *categoria* para assim filtrar os eventos por meio desse.

3.6 Projeto do Frontend

Para o projeto do frontend que serve de interface do sistema para o usuário, foi planejado uma *single page application* ou spa utilizando Angular com a biblioteca de componentes Angular material.

Foram desenvolvidas duas telas: a principal ou homepage e a de resultados, utilizando como inspiração o modelo do Google. O fluxo consiste no usuário fazer a pesquisa na tela principal e ser redirecionado para a tela de resultados onde ele pode verificar os eventos que agradam o mesmo podendo executar uma nova pesquisa, voltar para tela inicial ou ser redirecionado para o site que contém o evento.

Uma vez definido o fluxo, foi feito um estudo de cores, fontes e ícones a serem dispostos na aplicação para assim haja uma melhor experiência de usuário. A Figura 9 mostra um resumo do que foi definido através de tal estudo.

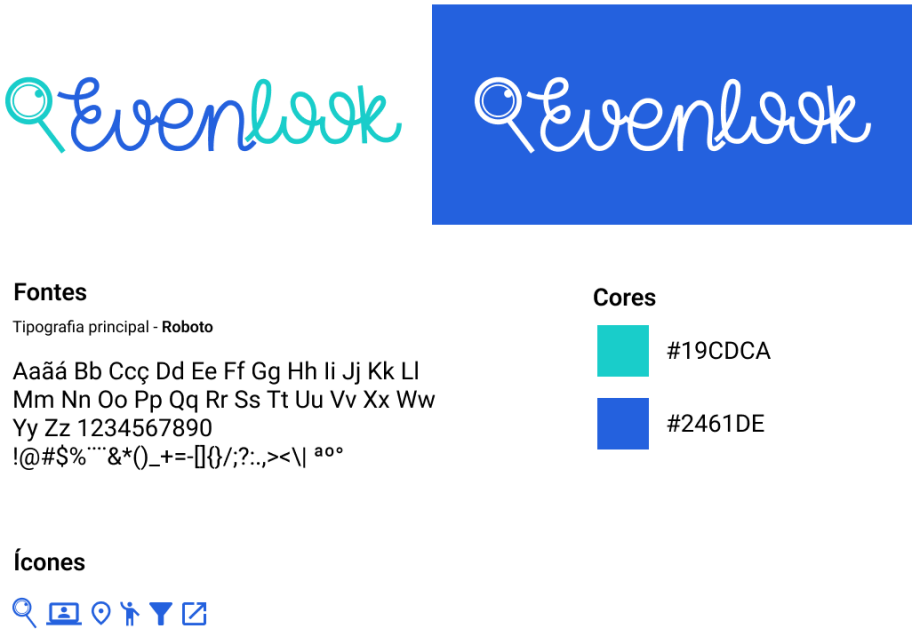


Figura 9 - Guia de estilização da aplicação frontend.

Sobre a identidade visual do site, o nome foi dado a partir da união de duas palavras em inglês: events (eventos) e look (olhar). O logotipo foi criado a partir de um lettering, pois a partir de pesquisas de similares e concorrentes, muitos deles usam logotipos que remetem ao manuscrito, com o intuito de trazer mais empatia para a marca, a ideia foi a mesma no caso do Evenlook. Um logotipo mais manuscrito, com formas arredondadas passa a sensação de fluidez, movimento e diversidade, contudo, ela também tem detalhes retos para demonstrar seriedade e compromisso, conceitos estabelecidos pela Gestalt . Pois o objetivo do site é ver eventos disponíveis, por isso o nome, Evenlook. Quanto a escolha de cores, foi pensado no azul e no verde pois, a partir da Psicologia das Cores [9] eles trazem uma sensação de confiança e leveza, transmitindo ao usuário a confiança que naquele site ele irá encontrar informações precisas, de forma descomplicada, sobre os eventos. Além disso, a cor azul é constantemente associada à tecnologia, e sendo este uma tecnologia de busca de eventos, a decisão pelo azul foi ainda mais acertada.

A tipografia escolhida foi a Roboto, pois ela já é bem conhecida no mercado, sendo uma fonte criada pelo Google para os meios digitais, com

uma boa leiturabilidade e legibilidade. E por fim, a escolha dos ícones foi a partir da biblioteca do Material Design, que já disponibiliza padrões iconográficos que o usuário já tem familiaridade, pois são os mesmos do Google e do Sistema Android, com exceção da lupa de pesquisa, que foi desenhada exclusivamente como símbolo da marca, e é usada em componentes de pesquisa dentro do site.

Na tela de homepage foram utilizados dois componentes principais a *Textfield* para fazer o campo de pesquisa e o botão personalizado ambos com as cores do sistema, como mostrado na figura seguinte:

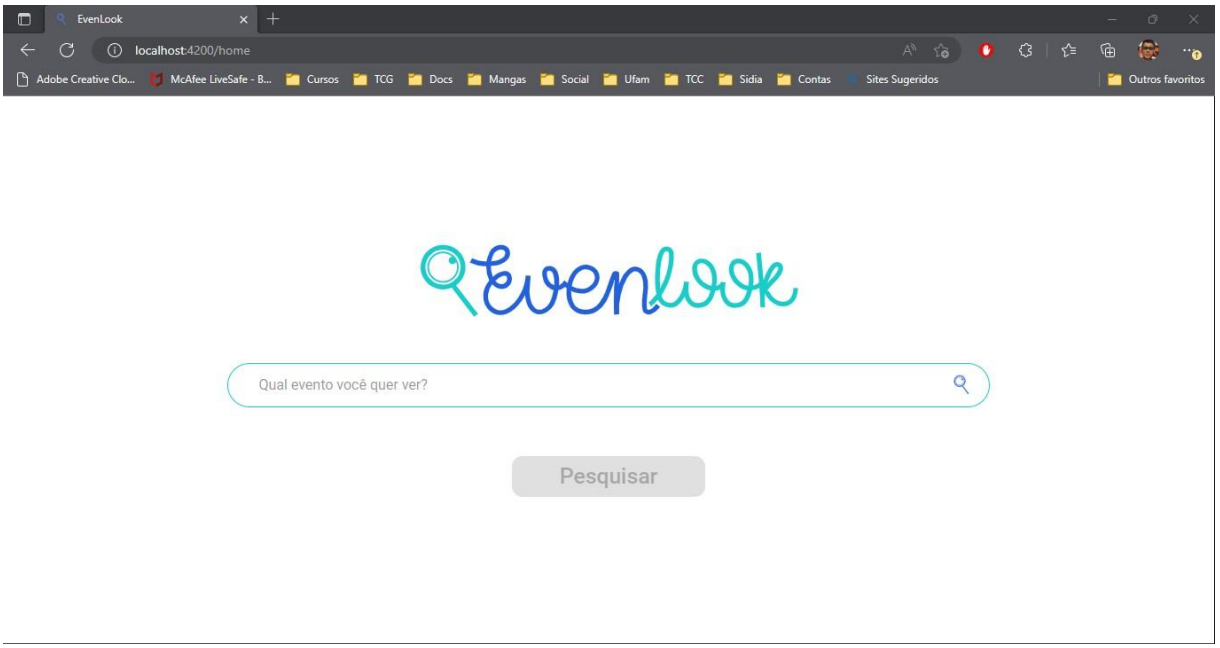


Figura 10 - Tela inicial da aplicação frontend.

E na tela de resultados foi utilizado um componente novo para a criação dos *cards* de eventos e a reutilização do componente de pesquisa mostrado na tela anterior, dentro do card de evento é possível verificar as informações do evento e também clicar no ícone que redireciona o usuário para a página do Sympla ou para a página do “Feiras do Brasil” dependendo da origem do evento, como mostrado na figura seguinte:

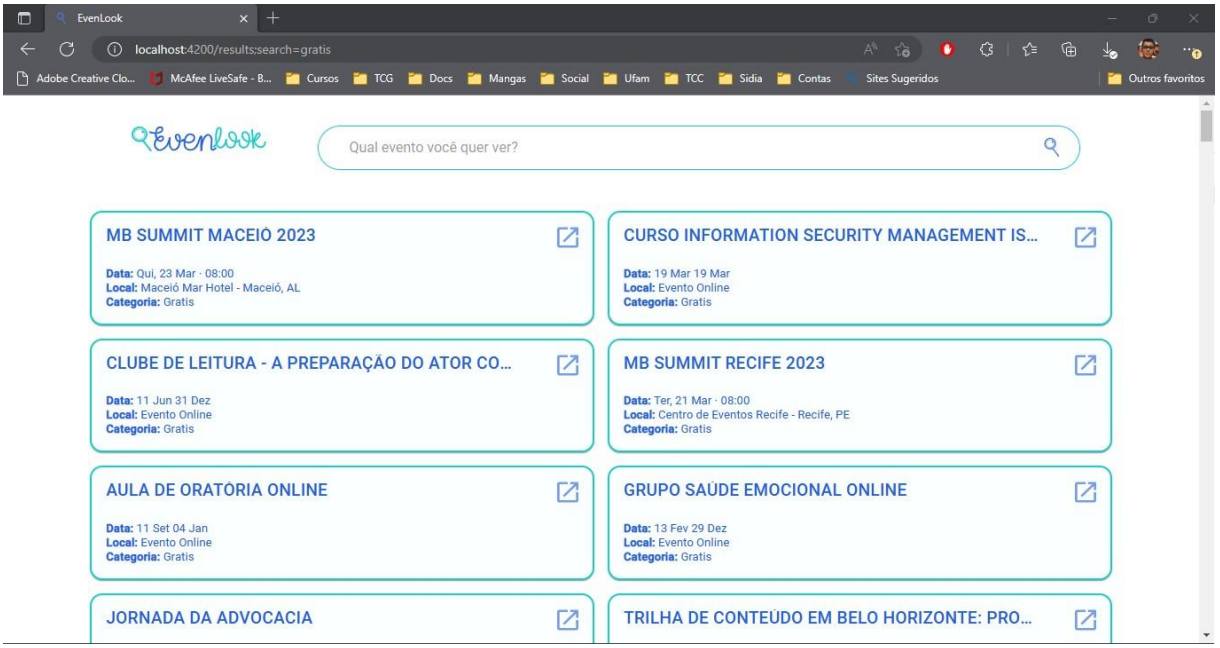


Figura 11 - Tela de resultados da aplicação frontend.

A arquitetura de diretórios do desenvolvimento do frontend foi feita da forma que é descrita pela Figura 12.

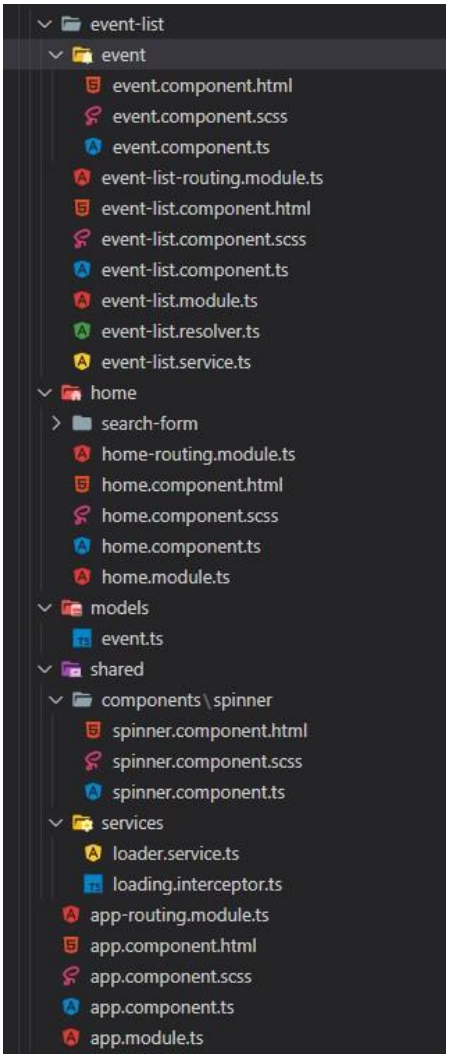


Figura 12 - Estrutura de diretórios do frontend.

Como a framework Angular criar vários arquivos de configurações por padrão explicarei mais os arquivos que eu alterei ou criei, sendo assim a seguir são explanadas as funções de cada diretório e dos arquivos críticos responsáveis pelo funcionamento do projeto:

- event-list: Neste diretório se encontram os componentes, módulos e serviços necessários para criar a página de resultados.
    - event: Neste diretório se encontra o componente de *card* para os eventos.
- home: Neste diretório se encontram os componentes e módulos necessários para criar a homepage do sistema.
    - search-form: Neste diretório se encontra o componente de *textfield* de pesquisa do sistema reutilizado também na

página de resultados.

- models: diretório que contém os modelos de dados utilizados pelo sistema.
    - event.ts: Abstração das principais características de um evento como nome, local, data, tipo de evento e url.
- shared: diretório que contém os componentes e serviços utilizados para iniciar o loading na página sempre que ocorre uma requisição para o backend.

4 Resultados

Neste capítulo será exibida a validação do desenvolvimento do projeto, que inclui a execução do serviço que foi projetado, os testes de resposta da rota implementada e a integração do serviço com a aplicação frontend.

4.1 Execução do Serviço Projetado

O serviço projetado neste sistema é a API Rest para executar o scraping nas páginas definidas na Seção 3.4. Esse serviço foi executado por meio de comandos docker dentro de um makefile. Para realizar este procedimento é necessário primeiramente instalar o docker na máquina que o serviço será executado, uma vez feito isso, é possível executar comandos via terminal, seja ele o do linux ou powershell do windows, para a criação da imagem e depois para a criação e execução do container dessa imagem. Após isso é possível verificar a integridade do sistema através de comandos docker, as Figuras 13, 14 e 15 mostram os resultados e comandos dos passos descritos neste tópico.

Figura 13 - Comando para criação da imagem do sistema

Figura 14 - Comando para criação e execução do container

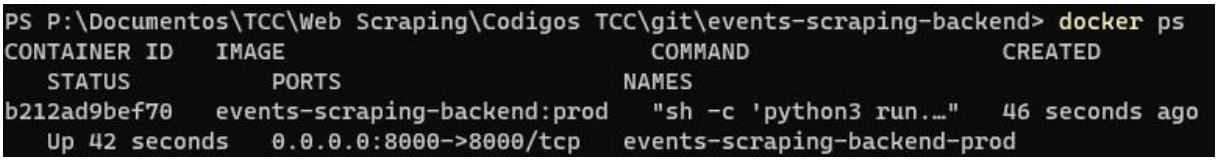


Figura 15 - Lista de container em execução na máquina

Desta forma este serviço está disponível para receber as requisições pelo nome de host “localhost:8000”.

4.2 Teste do Serviço Implementado

Para verificar as respostas do serviço a requisições de possíveis clientes, o serviço implementado foi testado utilizando o software Postman, que é um aplicativo desktop que atua como um cliente para API's Rest.

A rota que foi implementadas para o backend do sistema, que foi explicada na Seção 3.5, foi testada da seguinte forma:

- /search: foram feitos testes para o caso ideal onde é enviado tanto o parâmetro de nome quanto de categoria, para quando só é enviado categoria e quando só enviado nome e quando não é enviado ambos. As Figuras 16 a 19 ilustram as respostas dos testes.

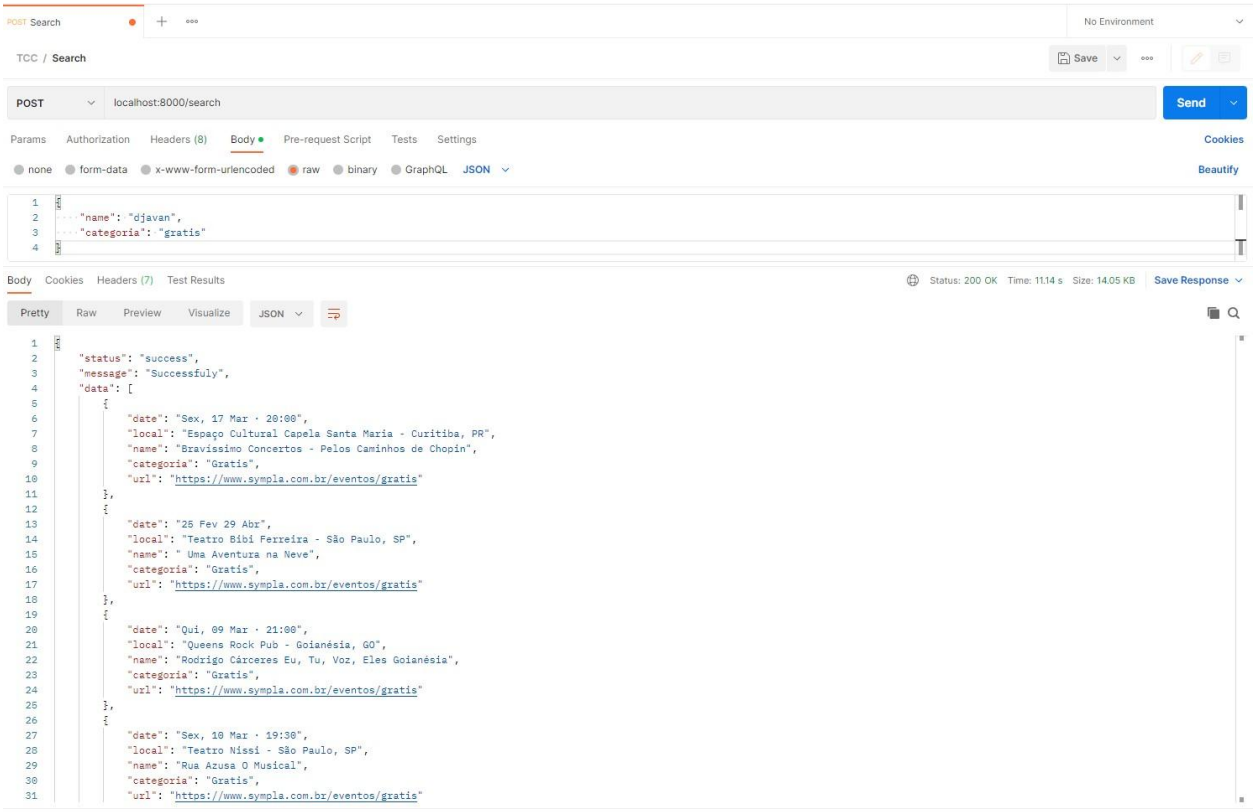


Figura 16 - Lista de eventos coletada e filtrada, pelo nome e categoria de maneira não excludente.

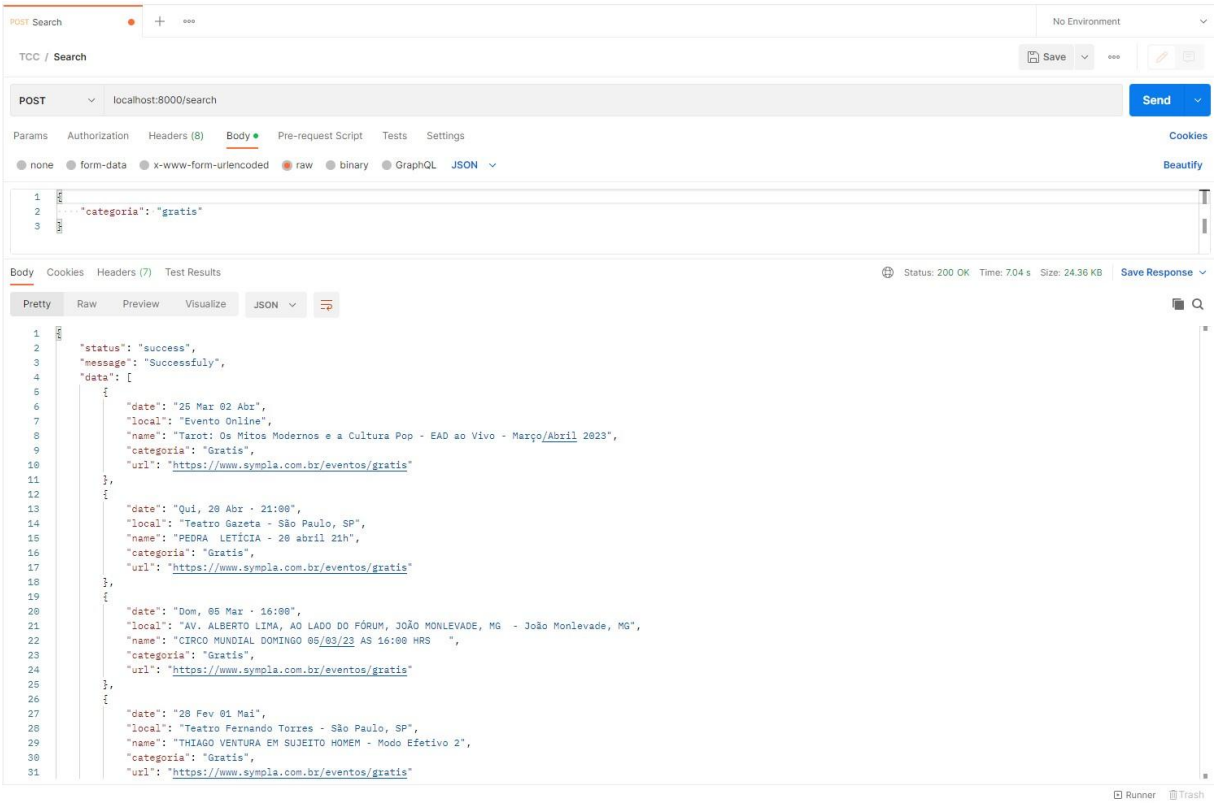


Figura 17 - Lista de eventos coletada e filtrada, pela categoria.

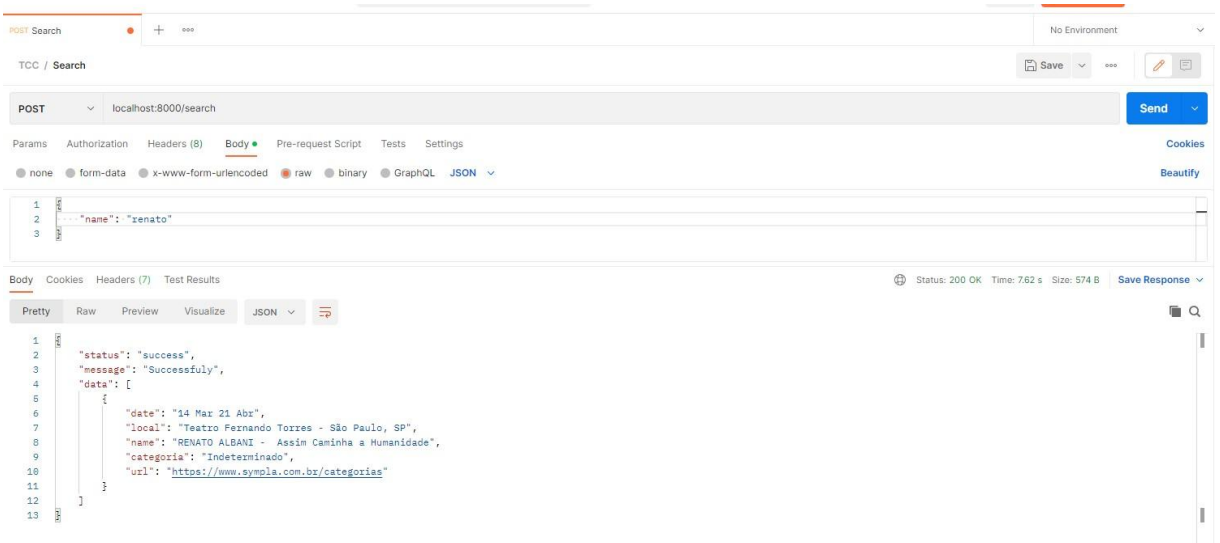


Figura 18 - Lista de eventos coletada e filtrada, pelo nome.

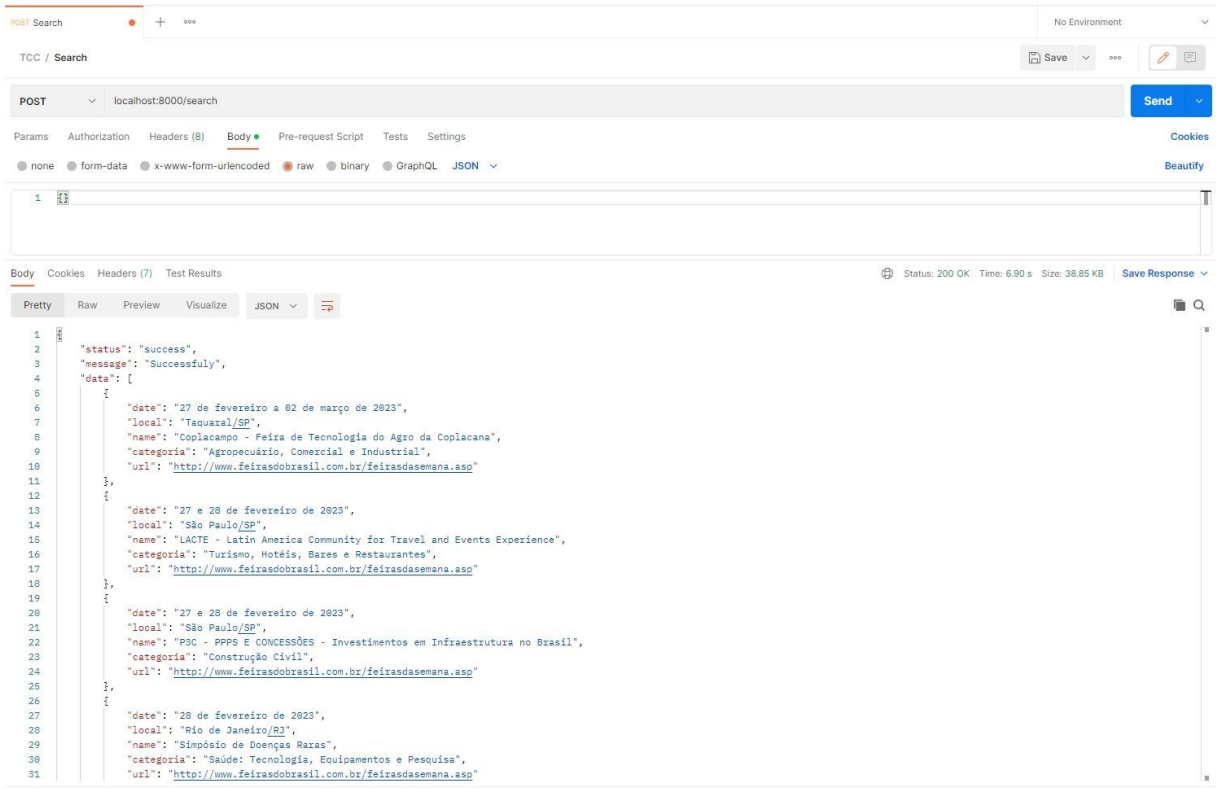


Figura 19 - Lista de todos os eventos de ambos os sites.

4.3 Execução da aplicação frontend

Para a execução da aplicação frontend também foi utilizado docker da mesma maneira que o serviço backend sendo as diferenças estão no conteúdo do Dockerfile, Docker compose e makefile pois precisam se adequar para configurar e inicializar uma aplicação frontend com Angular. O passo a passo para execução é o mesmo da Seção 4.1 e as Figuras 20, 21 e 22 mostram os resultados dos passos descritos.

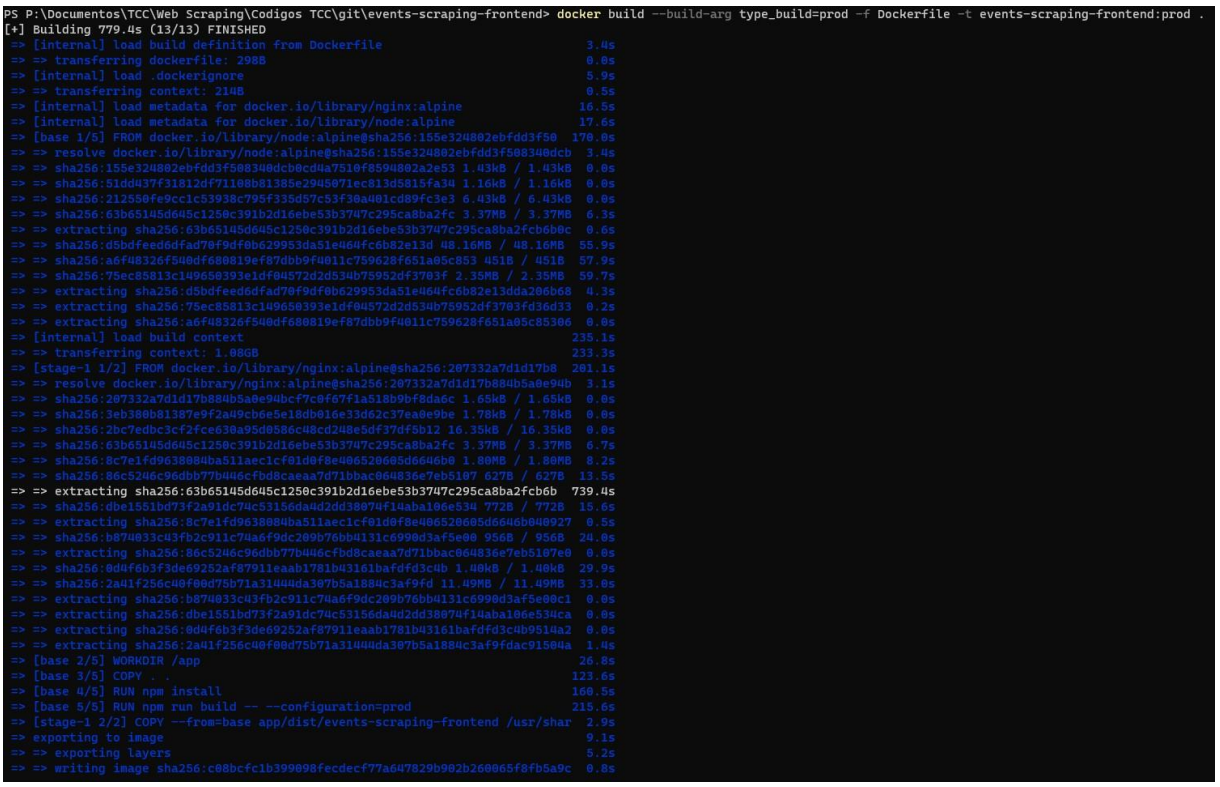


Figura 20 - Comando para criação da imagem da aplicação frontend

Figura 21 - Comando para criação e execução do container

Figura 22 - Lista de container em execução na máquina

4.4 Teste de integração do Sistema

Após validados os tipos de respostas esperados que o serviço de backend do sistema, utiliza-se a aplicação frontend desenvolvida para o sistema Evenlook atuando como interface do usuário para executar o scraping nas páginas proposta, o fluxo de telas do sistema foi exposto na

Seção 3.6, desta forma, aqui é dado ênfase no fluxo de execução natural primeiro sendo a pesquisa e logo em seguida os resultados.

Após a execução tanto do backend quanto do frontend o usuário poderá acessar a página mostrada na figura 23.

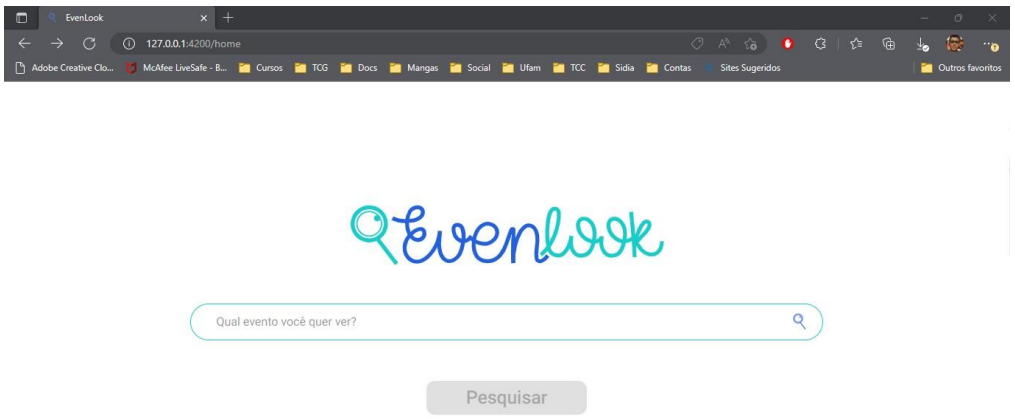


Figura 23 - Tela de Homepage do sistema

Nesta tela é possível preencher o campo de pesquisa e após o preenchimento de alguma letra no campo será habilitado o botão de pesquisa para assim submeter uma request para o backend. A figura 24 mostra o campo de pesquisa preenchido e o botão habilitado.

Figura 24 - Tela de Homepage com campo preenchido

Após o click no botão pesquisar o usuário será redirecionado para tela de *results* onde terá os resultados da pesquisa informada expostos em cards que podem redirecionar o usuário para o site de origem da informação através do ícone de redirecionamento. Na figura 25 é possível visualizar a tela com os cards preenchidos.

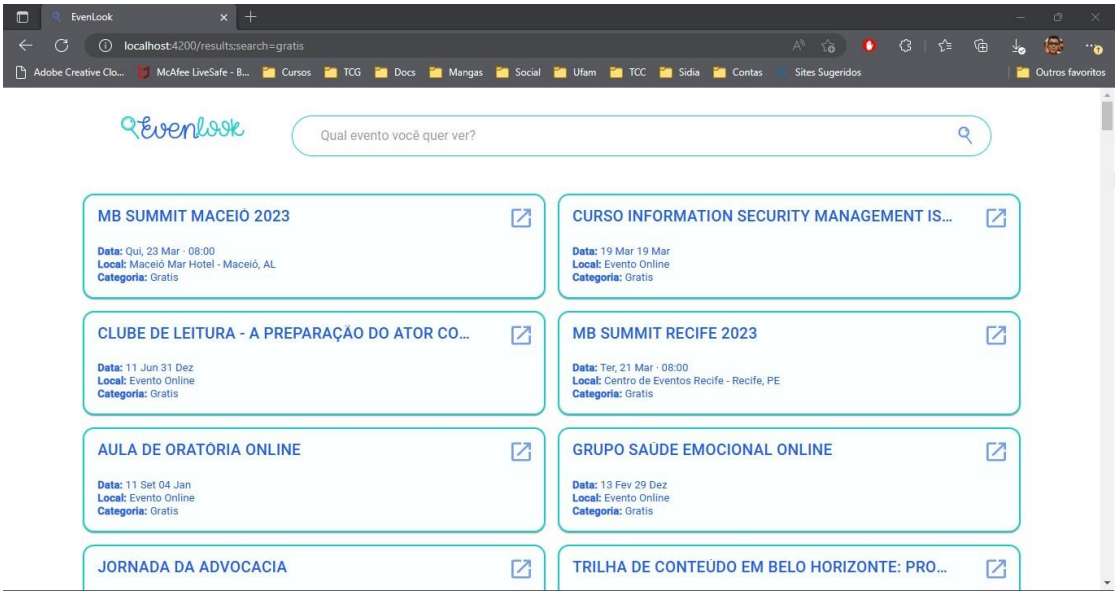


Figura 25 - Tela de Homepage com campo preenchido

Nessa tela também é possível executar uma nova busca por meio do campo de pesquisa ou se direcionar para tela de homepage por meio do ícone com a logo do sistema.

5 Conclusões

5.1 Considerações Finais

Com o desenvolvimento deste trabalho foi possível colocar em prática diversos temas e conceitos abordados durante o curso de Engenharia da Computação, como o de chamadas de procedimentos remotos que é em que se constitui de base para a implementação do modelo cliente-servidor utilizado neste projeto e o de estruturas de software tanto front quanto backend. A diagramação UML foi aplicada para uma concepção mais clara

da arquitetura desenvolvida no sistema e para que haja o proporcionamento de uma documentação inteligível que facilite possíveis manutenções a nível de código do projeto.

O conceito que agrega mais valor do ponto de vista tecnológico ao projeto é a utilização de web scraping para coleta de informações dos sites de maneira personalizada e direta para cada site proposto no trabalho.

Do ponto de vista social, neste trabalho, foi proposto um sistema que tem como premissa ser utilizado como um auxiliador tanto na aprendizagem dos alunos quanto para a abordagem das aulas dos professores.

5.2 Propostas para Trabalhos Futuros

Nesta seção, apresentamos algumas propostas para desenvolvimentos futuros relacionados aos métodos expostos neste trabalho. Estas propostas podem ser divididas em duas categorias. Na primeira, foram consideradas modificações estruturais no código e na segunda indicamos novas funcionalidades relevantes para o sistema.

Devido a diversidade de implementações, tecnologias e estruturas de cada site esse trabalho se concentrou em dois sites já expostos neste texto, então no que se refere à estrutura dos algoritmos, pode ser expandir o número de crawlers para sites diferentes dos propostos neste trabalho. Para isso também será adequado utilizar um orquestrado que ficará responsável por gerenciar os bots tirando a responsabilidade disso da view.

Com relação a novas funcionalidades um cadastro de usuário seria adequado, para assim poder favoritar e verificar o histórico de filtros do usuário.

Referências Bibliográficas

[1] BORGES, Luiz Eduardo. ***Python para desenvolvedores***. 1. ed. São Paulo: Novatec Editora Ltda, 2014.

[2] LUTZ, Mark; ASCHER, David. ***Aprendendo Python***. Tradução de João Tortello. 2.ed. Porto Alegre: Bookman, 2007.

[3] MITCHELL, Ryan. ***Web Scraping com Python***. 2ed. São Paulo: Novatec, 2019.

[4] MOREIRA, Marco Antônio. ***Teorias de Aprendizagem***. São Paulo: EPU, 1999.

[5] Mr. Sreenidhi S K , Ms. Tay Chinyi Helena. ***Styles of Learning Based on the Research of Fernald, Keller, Orton, Gillingham, Stillman , Montessori and Neil D Fleming*****. International Journal For Innovative Research In Multidisciplinary Field**. v. 3 n 4, p. 17-25 2017. Disponível em: <https://www.researchgate.net/profile/Sree-Nidhi-S-K/publication/317305325_Styles_of_Learning_VAK/links/59314ae0a6fdcc89e7937e2f/Styles-of-Learning-VAK.pdf>. Acesso em: 21 nov. 2022.

[6] **OMG® Unified Modeling Language®** (OMG UML®) Version 2.5. Disponível em: <https://www.omg.org/spec/UML/2.5.1/PDF>. Acesso em: 07 out. 2022.

[7] Booch, Grady. ***Uml - Guia do Usuário***. 2. ed. São Paulo: GEN LTC Editora Ltda, 2006.

[8] **Beautiful Soup Documentation — Beautiful Soup 4.9.0 documentation**. Disponível em: <https://www.crummy.com/software/BeautifulSoup/bs4/doc>. Acesso em: 07 out. 2022.

[9] HELLER, Eva. **A Psicologia das Cores: Como as cores afetam a emoção e a razão**. São Paulo: Editora Gustavo Gili, 2012.

[10] **What is Flask Python - Python Tutorial.** Disponível em: <https://pythonbasics.org/what-is-flask-python/>. Acesso em: 11 nov. 2022.

[11] **Flask Documentation**. Disponível em: <https://flask.palletsprojects.com/en/2.2.x/>. Acesso em: 13 nov. 2022

[12] **Docker: A 'Shipping Container' for Linux Code**. Disponível em: <https://www.linux.com/news/docker-shipping-container-linux-code/>. Acesso em: 13 dez. 2022

[13] **Docker overview**. Disponível em: <https://docs.docker.com/get-started/overview/>. Acesso em: 13 dez. 2022

[14] **Angular - What is Angular?**. Disponível em: <https://angular.io/guide/what-is-angular>. Acesso em: 20 dez. 2022