ĐẠI HỌC QUỐC GIA TP.HCM
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH

LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC

ỨNG DỤNG PYTHON VÀO LẬP TRÌNH WEBSITE ĐÁNH GIÁ SEO

GVHD: TS. NGUYỄN ĐỨC THÁI
GVPB: ThS. NGUYỄN HỒNG NAM

SVTH: NGUYỄN PHƯỚC THỊNH (1413785)

TP. HỒ CHÍ MINH, THÁNG 06/2019

Mục lục

Danh sách bảng

Danh sách hình vẽ

Chương I

Giới thiệu đề tài

1 Tính cấp thiết

SEO viết tắt của Search Engine Optimization có nghĩa là tối ưu hóa công cụ tìm kiếm, bao gồm tập hợp các phương pháp nhằn cải thiện thứ hạng website trang trang kết quả tìm kiếm, nổi trội nhất là trang Google.

Có rất nhiều tiêu chí để đánh giá một website có chuẩn SEO hay không như mức độ phân bổ từ khóa, số lượng liên kết bên ngoại, tốc độ tải trang,... Trong đó, cấu trúc mã nguồn website là một trong những yếu tố quan trọng giúp các trình thu thập dữ liệu của công cụ tìm kiếm phân tích website.

Hiện nay đã có nhiều website cung cấp dịch vụ đánh giá SEO, tuy nhiên hầu hết chúng đều thu phí người dùng và chưa hỗ trợ tiếng Việt. Do đó chúng tôi tạo ra ứng dụng này để đáp ứng nhu cầu sử dụng của người dùng Việt Nam và hoàn toàn miễn phí. Người dùng sẽ không cần phải am hiểu nhiều về lập trình mà vẫn có thể dễ dàng hiểu và sử dụng được ứng dụng của chúng tôi.

Chúng tôi sử dụng Python để tạo nên ứng dụng này, vì Python trong những năm gần đây được quan tâm và phát triển vượt bậc. Với hàng ngàn thư viện được chia sẻ miễn phí, việc sử dụng Python sẽ giúp ứng dụng của chúng tôi có thể mở rộng nhiều hơn trong tương lai.

2 Mục tiêu

Ứng dụng của chúng tôi sẽ đặt ra những mục tiêu để hoàn thành sau đây:

- Cung cấp dịch vụ đánh giá miễn phí cho người sử dụng.
- Tự động phân tích mã nguồn website bằng liên kết người dùng nhập vào.
- Chấm điểm website dựa trên phân tích cấu trúc SEO và mô tả kết quả.
- Cung cấp những tiêu chuẩn SEO đến người dùng thông qua các bài viết trên website.

3 Phương pháp thực hiện

Chúng tôi sử dụng Python để lập trình backend cho ứng dụng của mình. Python được biết đến là ngôn ngữ dành cho tính toán và phân tích nên thích hợp để xử lý các cú pháp cho website chúng tôi cần kiểm tra.

Bên cạnh đó, với kho thư viện đồ sộ được chia sẻ công khai và miễn phí trên `https://pypi.org` sẽ giúp chúng tôi triển khai nhanh chóng ứng dụng nhờ vào các framework mở được chia sẻ để sử dụng.

Những framework và thư viện chúng tôi sẽ sử dụng cho ứng dụng của mình:

- Django: Framework Python dùng để phát triển ứng dụng web.
- Requests, Lxml: Thư viện Python có nhiệm vụ phân tích cú pháp và lấy từng phần tử của website chúng tôi cần kiểm tra.
- reCAPTCHA: Ứng dụng do Google phát triển, nhằm hạn chế spam và BOT tác động lên website để giữ trang web trở nên an toàn.
- Bootstrap: Thư viện dùng để thiết kế giao diện cho trang web, nó hỗ trợ tốt cho việc hiển thị website đa nền tảng.
- Font Awesome: Thư viện cung cấp các icon cần thiết cho giao diện website.
- Heroku: Cloud platform miễn phí để chúng tôi triển khai ứng dụng Python của mình lên Internet.

4 Bố cục báo cáo

Tiếp theo, những phần sau chúng tôi sẽ trình bày cách chúng tôi sử dụng những framework và thư viện để xây dựng nên website cùng với những tiêu chí được áp dụng để đánh giá SEO cho một trang web. Phần kết luận đánh giá sẽ được giới thiệu vào mục cuối cùng kèm theo những tài liệu tham khảo mà chúng tôi sử dụng để hoàn thành báo cáo này.

Chương II

Nền tảng lý thuyết

Phần này chúng tôi sẽ giới thiệu cũng như trang bị những kiến thức cơ bản để sử dụng những công cụ được liệt kê ở phần trước.

1 Ngôn ngữ Python

Python hiện đang là một trong những ngôn ngữ lập trình phổ biến. Một phần nhờ vào khả năng dễ tiếp cận, cấu trúc rõ ràng và quan trọng hơn, nó có thể giải quyết tốt các bài toán kỹ thuật với thời gian thực thi nhanh và tiết kiệm dòng code. Python được tạo ra bởi Guido van Rossum và phát hành vào năm 1991[1].

Phiên bản sử dụng: 3.7.3

Kiến thức cơ bản

Biến (Variable): Không giống với các ngôn ngữ khác, Python không có câu lệnh riêng biệt để khai báo biến. Biến không cần phải khai báo kiểu giá trị nào và có thể thay đổi dựa vào giá trị mà nó được gán.

```
# x is of type int
x = 5
# x is now of type str
x = "Thinh"
```

Chuỗi (String): Chuỗi ký tự trong Python được chứa trong cặp dấu nháy đơn hoặc dấu nháy kép. Để hiển thị chuỗi ra màn hình, sử dụng lệnh `print()`.

```
a = "Hello, World!"
print(a)

>>>"Hello, World"
```

Toán tử (Operator):

- Số học:

$+$	Cộng
$-$	Trừ
$*$	Nhân
$/$	Chia
$\%$	Chia lấy phần dư
$**$	Lũy thừa
$//$	Chia lấy phần nguyên

Bảng II.1: Toán tử Số học

- So sánh:

$==$	Bằng
$!=$	Không bằng
$>$	Lớn hơn
$<$	Nhỏ hơn
$>=$	Lớn hơn hoặc bằng
$<=$	Nhỏ hơn hoặc bằng

Bảng II.2: Toán tử So sánh

- Logic:

and	Trả về `True` nếu 2 điều kiện đều đúng
or	Trả về `True` nếu 1 trong 2 điều kiện là đúng
not	Đảo ngược kết quả của điều kiện

Bảng II.3: Toán tử Logic

- Identity:

is	Trả về `True` nếu 2 biến cùng trỏ tới 1 đối tượng
is not	Trả về `True` nếu 2 biến không trỏ cùng đối tượng

Bảng II.4: Toán tử Identity

- Membership:

in	Trả về `True` nếu biến nằm trong tập hợp các biến
not in	Trả về `True` nếu biến không nằm trong tập hợp các biến

Bảng II.5: Toán tử Membership

Dictionary: Tập hợp không có thứ tự, có thể thay đổi và lập chỉ mục. Được biểu diễn bằng cặp dấu ngoặc nhọn, bên trong là khóa (key) và giá trị (value) tương ứng.

```
hoten = {
  "ho": "Nguyen Phuoc",
  "ten": "Thinh"
}
```

Câu điều kiện (If...Else): Dùng để thực thi một hành động sau khi thỏa điều kiện cho trước. Lưu ý trong Python, sử dụng thụt lề dòng để phân biệt các khối lệnh với nhau.

```
a = 1
b = 2
if a > b:
  print("a is greater than b")
elif a == b:
  print("a and b are equal")
else:
  print("b is greater than a")

>>>"b is greater than a"
```

Vòng lặp (For): Dùng để lặp qua một chuỗi (có thể là list, tuple, dictionary, set hoặc string).

```
hoten = ["Nguyen", "Phuoc", "Thinh"]
for x in hoten:
  print(x)

>>>"Nguyen"
>>>"Phuoc"
>>>"Thinh"
```

Hàm (Function): Gồm một khối code, được khởi chạy khi được gọi đến. Để truyền dữ liệu vào 1 hàm được gọi là tham số (parameter). Hàm trả về kết quả thông qua lệnh `return`.

```
# a function is defined using the def keyword
def add(n):
  return 1 + n
# calling a function
add(1)

>>>2
```

Lớp/Đối tượng (Class/Object): Python là ngôn ngữ lập trình hướng đối tượng. Hầu hết mọi thứ trong Python là một đối tượng (object), gồm thuộc tính và phương thức của nó. Sử dụng lớp (class) để khởi tạo 1 đối tượng mới.

```
# create a class
class Person:
  def __init__(self, name, age):
    self.name = name
```

```
    self.age = age
  # object method
  def func(self):
    print(f"My name is: {self.name}, {self.age} years old")
# create object
p = Person("Thinh", 20)
p.func()
# modify object property
p.age = 22
print(p.age)

>>>"My name is: Thinh, 20 years old"
>>>22
```

Module: Có thể xem module là 1 bộ thư viện mã code, được lưu bởi tệp hoặc thư mục tách biệt với project đang thực thi, được nhúng vào để tái sử dụng những bộ code chứa trong đó.

Để sử dụng các hàm hoặc lớp trong file `mymodule.py`, ta sử dụng lệnh sau:

```
import mymodule
# import only part from a module
from mymodule import myfunc
```

PIP: Là trình quản lý gói (package) hoặc module dành cho Python. Gói là nơi chứa tất cả các file cần thiết cho 1 module.

Để cài đặt 1 gói trong Python, sử dụng lệnh sau:

```
\>pip install Django
```

Xử lý ngoại lệ (Try...Except): Khi chương trình xảy ra lỗi hoặc ngoại lệ, Python sẽ dừng lại và đưa ra thông báo lỗi cho người dùng. Để tránh ứng dụng bị gián đoạn, sử dụng câu lệnh `try` để bắt và xử lý các ngoại lệ khi chương trình đang được thực thi.

```
try:
  print(x)
except NameError:
  print("Variable x is not defined")

>>>"Variable x is not defined"
```

2 Framework Django

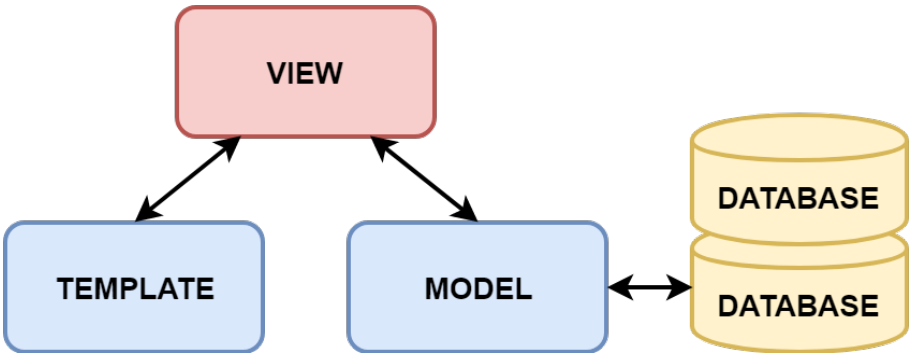
Django là một framework Python web cấp cao, thúc đẩy phát triển nhanh chóng, gọn gàng và tiện dụng. Được xây dựng bởi các nhà lập trình viên có kinh nghiệm, xử lý được các vấn đề rắc rối khi phát triển web, do đó người dùng chỉ cần quan tâm hoàn thiện các chức năng cho web mà không cần phải quá lo lắng về nền tảng phía sau. Và quan trọng

nó là mã nguồn mở và miễn phí[2].

Phiên bản sử dụng: 2.2

Mô hình MTV (Model - Template - View)

Trong quá trình phát triển dự án và ứng dụng. Django đưa ra mô hình cấu trúc chung cho hệ thống nhằm đảm bảo thiết kế nhất quán và hiệu quả[3].



Hình II.1: Mô hình MTV trong Django

Trong đó:

- Model (M): Xử lý biểu diễn dữ liệu của bạn, nó được thể hiện như một giao diện cho dữ liệu được lưu trữ trong cơ sở dữ liệu và cũng cho phép bạn tương tác với dữ liệu của mình mà không phải cần kiến thức về các lệnh cơ sở dữ liệu.
- Template (T): Đại diện cho những gì bạn thấy trên trình duyệt cho ứng dụng web.
- View (V): Cung cấp logic để xử lý luồng dữ liệu trong chế độ xem hoặc cập nhật dữ liệu của Model, nó sử dụng logic được lập trình để tìm ra những gì được chuyển từ cơ sở dữ liệu thông qua Model và chuyển đến Template. Ngoài ra, nó có thể nhận thông tin từ người dùng thông qua Template và thực hiện logic đã cho bằng cách thay đổi chế độ xem hoặc cập nhật dữ liệu qua Model.

Cấu trúc dự án và ứng dụng

Một project của Django (ví dụ, `lvtn`) sẽ có cấu trúc và chức năng như sau:

```
lvtn\
  lvtn\
    __init__.py
    settings.py
    urls.py
    wsgi.py
  manage.py
```

Trong đó:

- `manage.py`: Một CLI giúp tương tác với ứng dụng web.
- `lvtn\__init__.py`: File rỗng, để chỉ cho Python biết thư mục này nên được xem là một gói.
- `lvtn\settings.py`: Chứa các tùy chỉnh của project.
- `lvtn\urls.py`: Các khai báo URL cho trang web.
- `lvtn\wsgi.py`: Được sử dụng khi deploy project lên Internet.

Server phát triển: Dùng để khởi chạy ứng dụng website trên máy tính local. Khi server đang chạy, truy cập vào địa chỉ `http://localhost:8000` trên trình duyệt để thấy ứng dụng đang được chạy.

Tạo app mới: Mỗi ứng dụng được viết trong Django sẽ tuân thủ theo một quy ước nhất định. Django đi kèm với một tiện ích tự động tạo cấu trúc thư mục cơ bản của một ứng dụng, do đó người lập trình chỉ cần quan tâm đến việc phát triển code bên trong mà thôi.

Một app (ví dụ, `checkweb`) trong project của Django được tạo ra có cấu trúc như sau:

```
checkweb\
  migrations\
    __init__.py
  __init__.py
  admin.py
  apps.py
  models.py
  tests.py
  views.py
```

Trong đó:

- `migrations\`: Thư mục chứa các file được sinh ra khi có thay đổi về cấu trúc cơ sở dữ liệu.
- `admin.py`: Dùng để thiết đặt các thuộc tính được hiển trị trong trang quản trị admin mà Django cung cấp sẵn.
- `apps.py`: Khai báo app được sử dụng trong project, đảm bảo rằng các app không bị trùng lặp trong 1 dự án.
- `models.py`: Django hỗ trợ các phương thức để xử lý cơ sở dữ liệu mà không cần sử dụng đến các câu lệnh truy vấn SQL trực tiếp.
- `tests.py`: Được người dùng sử dụng để triển khai các kịch bản thử nghiệm và rà soát lỗi trước khi phát hành ứng dụng.
- `views.py`: Đóng vai trò xử lý trung tâm của ứng dụng, quản lý việc hiển thị, kết nối đến cơ sở dữ liệu và thực thi các hàm do lập trình viên thêm vào ứng dụng.

Sau khi tạo xong app, cần phải khai báo trong project bằng các thêm dòng sau vào file `lvtn\settings.py`:

```
INSTALLED_APPS = [
    "django.contrib.admin",
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",
    # add code below
    "checkweb.apps.CheckwebConfig",
]
```

Class-based view: Đây là chức năng được Django hỗ trợ, giúp lập trình viên ít phải viết code hơn để hiện thị một giao diện lên trình duyệt web. Nó hỗ trợ tốt trong việc truyền tham số, lấy giá trị từ model và có thể dễ dàng tùy chỉnh theo ý muốn.

Để sử dụng, cần phải thêm module vào file muốn dùng nó. Đoạn code sau có chức năng hiển thị file `about.html` ra đường dẫn `http://localhost:8000/about/`.

```
from django.urls import path
from django.views.generic import TemplateView

urlpatterns = [
  path("about/", TemplateView.as_view(template_name="about.html")),
]
```

Django template: Dựa trên file `.html` nhưng có chèn thêm các đoạn code riêng biệt để mỗi khi chạy chương trình, Django sẽ render ra giao diện lên trình duyệt tương ứng.

```
{% extends "base_generic.html" %}
{% block title %}{{ section.title }}{% endblock %}
{% block content %}
<h1>{{ section.title }}</h1>
{% for story in story_list %}
<h2>
  <a href="{{ story.get_absolute_url }}">
    {{ story.headline|upper }}
  </a>
</h2>
<p>{{ story.tease|truncatewords:100 }}</p>
{% endfor %}
{% endblock %}
```

3 Thư viện Python

3.1 Requests

Requests là một thư viện HTTP thanh lịch và đơn giản viết bằng Python, được xây dựng dành cho con người[4].

Phiên bản sử dụng: 2.21.0

Cách cài đặt:

```
\>pip install requests
```

Phương thức `get`: Dùng để lấy toàn bộ nội dung trang web dựa trên tham số url.

```
import requests
page = requests.get(url)
```

3.2 Lxml

Lxml là thư viện được sử dụng để phân tách các thành phần trong mã nguồn nhằm hỗ trợ cho cú pháp `xpath` lấy nội dung từ website[5].

Phiên bản sử dụng: 4.3.3

Cách cài đặt:

```
\>pip install lxml
```

Gói `lxml.html`: Dùng để phân tách chuỗi HTML.

```
from lxml import html
content = html.fromstring(page.content)
value = content.xpath("//title/text()")
```

Cú pháp phân tích sử dụng xpath

XPath sử dụng các biểu thức đường dẫn để chọn các nút hoặc tập hợp nút trong tài liệu XML. Các biểu thức đường dẫn này trông rất giống với các biểu thức đường dẫn bạn sử dụng với các hệ thống tệp máy tính truyền thống. Trong dự án, chúng tôi dùng để phân tích cú pháp HTML[6].

Cú pháp chọn nút

nodename	Chọn tất cả các nút có tên là "nodename"
/	Chọn từ nút gốc
//	Chọn các nút từ nút hiện tại khớp với lựa chọn bất kể chúng ở đâu
.	Chọn nút hiện tại
..	Chọn nút cha của nút hiện tại
@	Chọn thuộc tính trong thẻ

Bảng II.6: Cú pháp chọn nút trong xpath

4 Thư viện giao diện

4.1 Bootstrap

Bootstrap là một thư viện HTML, CSS và JavaScript phổ biến trên thế giới để xây dựng nên các website có giao diện đáp ứng trên nhiều kích cỡ thiết bị khác nhau[7].

Phiên bản sử dụng: 4.3.1

Cách sử sử dụng Bootstrap vào website:

- CSS: Sao chép và dán dòng code bên dưới vào trong thẻ `<head>` trước tất cả các định dạng khác để tải CSS của Bootstrap.

```
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/
4.3.1/css/bootstrap.min.css">
```

- JS: Đặt trong thẻ `<script>` ở gần cuối trang web, trước khi đóng thẻ `</body>` để kích hoạt chúng. jQuery phải được đặt trước, đến Popper.js và sau cùng là phần JavaScript.

```
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/
popper.min.js"></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/
bootstrap.min.js"></script>
```

4.2 Font Awesome

Font Awesome là thư viện miễn phí cung cấp các icon dạng vector và các logo xã hội lên website[8].

Phiên bản sử dụng: 5.8.2

Trước khi sử dụng được, cần phải chèn dòng code bên dưới vào thẻ `<head>` nằm ở đầu trang web.

```
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.8.2/css/
all.css">
```

Để chèn icon vào trang web, sử dụng dòng code tương tự cú pháp bên dưới:

```
<i class="fas fa-heart"></i>
```

5 reCAPTCHA

reCAPTCHA là một dịch vụ miễn phí bảo vệ website tránh khỏi spam và lạm dụng[9].

Phiên bản sử dụng: reCAPTCHA v2 Invisible

Cách đơn giản để sử dụng reCAPTCHA vào trang web bằng cách nhúng mã JavaScript và thẻ `g-recaptcha`. Thẻ `g-recaptcha` là 1 thẻ `button` với tên class là `"g-recaptcha"`, có thuộc tính `data-sitekey` chứa Site key được cấp khi đăng ký sử dụng reCAPTCHA.

```
<html>
    <head>
      <title>reCAPTCHA demo: Simple page</title>
      <script src="https://www.google.com/recaptcha/api.js" async
          defer></script>
      <script>
        function onSubmit(token) {
          document.getElementById("demo-form").submit();
        }
      </script>
    </head>
    <body>
      <form id="demo-form" action="?" method="POST">
        <button class="g-recaptcha" data-sitekey="your_site_key"
            data-callback="onSubmit">Submit</button>
        <br/>
      </form>
    </body>
</html>
```

Sau khi submit form sử dụng reCAPTCHA, cần phải gửi giá trị có tên là `g-recaptcha-response` bằng phương thức POST về máy chủ của Google để xác minh người dùng đã xác thực bằng reCAPTCHA tại địa chỉ `https://www.google.com/recaptcha/api/siteverify`.

- secret (bắt buộc): Khóa Secret key được cấp đồng thời với Site key khi đăng ký sử dụng.
- response (bắt buộc): Kết quả trả về của thuộc tính có tên là `g-recaptcha-response`.
- remoteip: Địa chỉ IP của người dùng cuối.

Tiếp theo, Google sẽ trả về kết quả kiểm tra có thỏa reCAPTCHA dưới dạng JSON bằng các giá trị sau:

```
{
  "success": true|false,
  "challenge_ts": timestamp, # timestamp of the challenge load
  "hostname": string, # the hostname of the site where the reCAPTCHA was solved
  "error-codes": [...] # optional
}
```

Dựa vào kết quả này, chúng tôi có thể biết được người dùng đã có giải được captcha hay chưa, sau đó tiến hành các yêu cầu từ người dùng.

6 Heroku platform

Heroku là một nền tảng đám mây cho phép xây dựng, phân phối, giám sát và mở rộng ứng dụng[10].

Đầu tiên và quan trọng nhất, các ứng dụng Heroku yêu cầu một file `Procfile` để cài đặt nền tảng sử dụng, được đặt tại thư mục gốc.

`Procfile`

```
web: gunicorn myproject.wsgi
```

File `Procfile` này yêu cầu `Gunicorn`, 1 máy chủ web được khuyến nghị dùng cho ứng dụng Django. Để cài đặt, sử dụng lệnh:

```
\>pip install gunicorn
```

Thay đổi trong file `settings.py` của ứng dụng Django. Khi sử dụng Heroku, các thông tin nhạy cảm sẽ được lưu trữ trong môi trường được gọi là config vars. Nó bao gồm các thông tin để kết nối đến cơ sở dữ liệu, trong khi bình thường sẽ được ghi trong file `settings.py` của Django.

Gói `django-heroku` sẽ tự động cấu hình ứng dụng Django để nó hoạt động trên Heroku. Nó tương thích với các ứng dụng Django 2.0. Để cài đặt, sử dụng lệnh:

```
\>pip install django-heroku
```

Sau khi cài đặt, cần phải `import` câu lệnh sau vào đầu file `settings.py`:

```
import django_heroku
```

Sau đó thêm phần sau vào cuối file `settings.py`:

```
# activate django-heroku.
django_heroku.settings(locals())
```

Chương III

Các tiêu chuẩn SEO

Để thu hút khách hàng truy cập và sử dụng một webstie, có rất nhiều tiêu chí như giao diện đẹp, tốc độ đáp ứng nhanh, thiết kế thuận tiện nhằm mang đến trải nghiệm tốt cho người dùng. Các vấn đề trên là một trong những tiêu chí mà công cụ tìm kiếm dùng để đánh giá website xem có phù hợp mà từ đó đề xuất lên trang kết quả khi có người dùng tìm kiếm dựa trên nội dung có liên quan.

Bên cạnh đó, việc xây dựng một website có cấu trúc phù hợp, sẽ giúp thể hiện rõ ràng nội dung khi các công cụ tìm kiếm phân tích website, qua đó nâng thứ hạng website trên trang kết quả tìm kiếm. Điều này sẽ khiến người dùng dễ dàng tiếp cận và truy cập vào website của mình. Ứng dụng của chúng tôi được tạo ra nhằm phân tích cú pháp SEO của một website, giúp người dùng tìm ra những thiếu sót về cú pháp trong SEO và sau đó cải thiện chúng.

Sau đây là các tiêu chuẩn cần thiết cho một website chuẩn SEO về mặt cú pháp, chúng tôi tham khảo bài viết từ trang moz.com là một trong những dịch vụ phân tích SEO uy tín và được thành lập từ năm 2004[11]:

1 Thẻ header

Thẻ tiêu đề là một yếu tố HTML được sử dụng để chỉ định các tiêu đề trên trang của bạn. Thẻ tiêu đề chính, được gọi là `h1`, thường được dành riêng cho tiêu đề của trang. Nó trông như thế này:

```
<h1>Tieu de trang</h1>
```

Ngoài ra còn có các tiêu đề phụ đi từ thẻ `h2` đến `h6`, mặc dù không cần sử dụng tất cả các tiêu đề này trên một trang. Hệ thống phân cấp của các thẻ tiêu đề đi từ `h1` đến `h6` theo thứ tự quan trọng giảm dần.

Mỗi trang nên có một `h1` duy nhất mô tả chủ đề chính của trang, điều này thường được tạo tự động từ tiêu đề của trang. Là tiêu đề mô tả chính của trang, `h1` nên chứa từ khóa hoặc cụm từ chính của trang đó.

Chủ đề chính của trang được giới thiệu trong tiêu đề `h1` và mỗi tiêu đề bổ sung được sử dụng để giới thiệu một chủ đề phụ mới. Ví dụ, `h2` cụ thể hơn `h1` và các thẻ `h3` cụ thể hơn so với `h2`.

2 Liên kết nội bộ

Một phần của trang web có khả năng thu thập dữ liệu nằm trong cấu trúc liên kết nội bộ của nó. Khi bạn liên kết đến các trang khác trên trang web của mình, bạn đảm bảo rằng trình thu thập thông tin của công cụ tìm kiếm có thể tìm thấy tất cả các trang của bạn và giúp khách truy cập điều hướng trang web của bạn.

3 Văn bản neo

Văn bản neo là văn bản mà bạn liên kết đến các trang. Dưới đây, bạn có thể thấy một ví dụ về những gì một liên kết không có văn bản neo và một liên kết với văn bản neo sẽ trông như thế nào trong HTML.

```
<a href="http://www.example.com"></a>
<a href="http://www.example.com" title="Keyword Text">Keyword Text</a>
```

Trên chế độ xem trực tiếp, nó sẽ trông như thế này:

```
http://www.example.com
Keyword Text
```

Văn bản neo gửi tín hiệu đến các công cụ tìm kiếm liên quan đến nội dung của trang đích. Ví dụ, nếu muốn liên kết đến một trang trên trang web bằng cách sử dụng văn bản neo "Học hỏi SEO", thì đó là một chỉ báo tốt cho các công cụ tìm kiếm rằng trang được nhắm mục tiêu là nơi mọi người có thể tìm hiểu về SEO.

4 Khối lượng liên kết

Trong Nguyên tắc quản trị trang web chung của Google, họ nói việc giới hạn số lượng liên kết trên một trang ở mức hợp lý (nhiều nhất là vài nghìn). Đây là một phần của hướng dẫn kỹ thuật của Google, thay vì phần hướng dẫn chất lượng, vì vậy có quá nhiều các liên kết nội bộ không phải là điều quá nghiêm trọng, nhưng nó ảnh hưởng đến cách Google tìm và đánh giá các trang của bạn.

Vì vậy, bạn chỉ nên liên kết khi bạn muốn nói điều đó. Quá nhiều liên kết không chỉ làm loãng nội dung của mỗi liên kết, mà chúng còn có thể không có ích và quá sức đối với việc Google hiểu trang của bạn.

Tập trung vào chất lượng và giúp người dùng điều hướng trang web của bạn sẽ có khả năng bạn đã giành chiến thắng mà không phải lo lắng về quá nhiều về số lượng liên kết.

5 Chuyển hướng

Xóa và đổi tên trang là một cách phổ biến, nhưng trong trường hợp bạn di chuyển một trang, hãy đảm bảo cập nhật các liên kết đến URL cũ đó. Bạn nên đảm bảo chuyển hướng URL đến vị trí mới của nó, nhưng nếu có thể, hãy cập nhật tất cả các liên kết nội bộ đến URL đó tại nguồn để người dùng và trình thu thập thông tin không phải chuyển hướng đến trang đích. Hãy cẩn thận để tránh các vòng chuyển hướng quá nhiều (theo Google, nên giữ số vòng chuyển hướng không quá 3 và ít hơn 5).

Thay vì:

```
example.com/location1 -> example.com/location2 -> example.com/location3
```

Nên dùng:

```
example.com/location1 -> example.com/location3
```

6 Tối ưu hình ảnh

Hình ảnh là thủ phạm lớn nhất của các trang web chậm. Cách tốt nhất để giải quyết vấn đề này là nén hình ảnh của bạn. Một cách khác để giúp tối ưu hóa hình ảnh của bạn (và cải thiện tốc độ trang) là chọn định dạng hình ảnh phù hợp.

Cách chọn định dạng hình ảnh sẽ sử dụng:

- Nếu hình ảnh của bạn là ảnh động, hãy sử dụng GIF.
- Nếu bạn không cần giữ độ phân giải hình ảnh cao, hãy sử dụng JPEG (và kiểm tra các cài đặt nén khác nhau).
- Nếu bạn cần duy trì độ phân giải hình ảnh cao, hãy sử dụng PNG.
  - Nếu hình ảnh của bạn có nhiều màu sắc, hãy sử dụng PNG-24.
  - Nếu hình ảnh của bạn không có nhiều màu sắc, hãy sử dụng PNG-8.

7 Thuộc tính alt

Văn bản thay thế trong hình ảnh là một nguyên tắc của khả năng truy cập web và được sử dụng để mô tả khi hình ảnh không thể hiển thị. Điều quan trọng là phải có mô tả văn bản thay thế để có thể hiểu những gì hình ảnh trên trang web của bạn mô tả vấn đề gì.

Các bot công cụ tìm kiếm cũng thu thập dữ liệu văn bản thay thế để hiểu rõ hơn về hình ảnh của bạn, điều này mang lại cho bạn lợi ích bổ sung trong việc cung cấp bối cảnh

hình ảnh tốt hơn cho các công cụ tìm kiếm. Chỉ cần đảm bảo rằng các mô tả alt của bạn đọc tự nhiên cho mọi người và tránh nhồi nhét từ khóa cho các công cụ tìm kiếm.

Thay vì:

```
<img src="grumpycat.gif" alt="grumpy cat, cat is grumpy, grumpy cat gif">
```

Nên dùng:

```
<img src="grumpycat.gif" alt="A black cat looking very grumpy at a big spotted
    dog">
```

8 Định dạng nội dung

Trang của bạn có thể chứa nội dung tốt nhất từng được viết về một chủ đề, nhưng nếu nó được định dạng không đúng, khán giả của bạn có thể không bao giờ đọc nó. Một số nguyên tắc có thể thúc đẩy khả năng đọc, bao gồm:

- Kích thước và màu sắc: tránh các phông chữ quá nhỏ. Google khuyến nghị sử dụng phông chữ 16px trở lên để tối ưu hóa trên điện thoại di động. Màu văn bản liên quan đến màu nền của trang cũng sẽ thúc đẩy khả năng đọc.
- Tiêu đề: chia nhỏ nội dung của bạn bằng các tiêu đề hữu ích có thể giúp người đọc điều hướng trang. Điều này đặc biệt hữu ích trên các trang dài, nơi người đọc có thể chỉ tìm kiếm thông tin từ một phần cụ thể.
- Ý chính: tuyệt vời cho danh sách, nội dung này có thể giúp người đọc đọc lướt và nhanh chóng tìm thấy thông tin họ cần.
- Phương tiện hỗ trợ: khi thích hợp, bao gồm hình ảnh, video và các tiện ích sẽ bổ sung cho nội dung của bạn.
- In đậm và in nghiêng: việc sử dụng hợp lý các tùy chọn định dạng này có thể gọi ra những điểm quan trọng bạn muốn giao tiếp.

9 Thẻ title

Thẻ tiêu đề của trang là một phần tử HTML mô tả, chỉ định tiêu đề của một trang web cụ thể. Chúng được lồng trong thẻ đầu của mỗi trang và trông như thế này:

```
<head>
  <title>Example Title</title>
</head>
```

Mỗi trang trên trang web của bạn nên có một thẻ tiêu đề mô tả, độc đáo. Những gì bạn nhập vào trường thẻ tiêu đề sẽ hiển thị ở đây trong kết quả tìm kiếm, mặc dù trong

một số trường hợp, Google có thể điều chỉnh cách thẻ tiêu đề của bạn xuất hiện trong kết quả tìm kiếm.

Thẻ tiêu đề của bạn có vai trò lớn đối với mọi người, ấn tượng đầu tiên về trang web của bạn và nó là một công cụ cực kỳ hiệu quả để thu hút người tìm kiếm đến trang của bạn hơn bất kỳ kết quả nào khác. Thẻ tiêu đề của bạn càng hấp dẫn, kết hợp với thứ hạng cao trong kết quả tìm kiếm, bạn càng thu hút được nhiều khách truy cập vào trang web của bạn. Điều này nhấn mạnh rằng SEO không chỉ là về công cụ tìm kiếm, mà là toàn bộ trải nghiệm người dùng.

Điều gì làm cho một thẻ tiêu đề hiệu quả?

- Từ khóa: có từ khóa mục tiêu của bạn trong tiêu đề có thể giúp cả người dùng và công cụ tìm kiếm hiểu trang của bạn nói về cái gì. Ngoài ra, là mặt tiền của trang web, từ khóa của bạn càng có nhiều khả năng người dùng sẽ đọc chúng (và hy vọng nhấp chuột) và chúng có thể hữu ích hơn để xếp hạng.

- Độ dài: trung bình, các công cụ tìm kiếm hiển thị 50 ký tự 60 đầu tiên ($\approx$ 512px) của thẻ tiêu đề trong kết quả tìm kiếm. Nếu thẻ tiêu đề của bạn vượt quá các ký tự được cho phép, dấu chấm lửng "..." sẽ xuất hiện ở nơi tiêu đề bị cắt.

- Thương hiệu: một đề cập đến tên thương hiệu vì nó thúc đẩy nhận thức về thương hiệu và tạo ra tỷ lệ nhấp cao hơn trong số những người quen thuộc với trang web. Đôi khi, nên đặt thương hiệu của bạn ở đầu thẻ tiêu đề, chẳng hạn như trên trang chủ của bạn, nhưng hãy chú ý đến những gì bạn đang cố xếp hạng và đặt những từ đó gần hơn với đầu thẻ tiêu đề.

10 Thẻ meta descriptions

Giống như thẻ tiêu đề, thẻ meta mô tả là các thành phần HTML mô tả nội dung của trang mà chúng xuất hiện. Chúng cũng được lồng trong thẻ head và trông như thế này:

```
<head>
  <meta name="description" content="Description of page here.">
</head>
```

Điều gì làm cho một mô tả meta hiệu quả?

- Mức độ liên quan: meta mô tả phải phù hợp với nội dung của trang của bạn, vì nó sẽ tóm tắt khái niệm chính của bạn dưới một số hình thức. Bạn nên cung cấp cho người tìm kiếm đủ thông tin để biết họ đã tìm thấy một trang đủ phù hợp để trả lời câu hỏi của họ, mà không cung cấp quá nhiều thông tin để loại bỏ nhu cầu nhấp qua trang web của bạn.

- Độ dài: công cụ tìm kiếm có xu hướng cắt ngắn mô tả meta thành khoảng 155 ký tự. Nó tốt nhất để viết meta mô tả giữa 150 đến 300 ký tự.

11 Định dạng URL

Các công cụ tìm kiếm yêu cầu các URL duy nhất cho mỗi trang trên trang web của bạn để họ có thể hiển thị các trang của bạn trong kết quả tìm kiếm, nhưng cấu trúc và đặt tên URL rõ ràng cũng hữu ích cho những người đang cố gắng hiểu URL cụ thể là gì.

Thay vì:

```
example.com/asdf/453?=recipe-23432-1123
```

Nên dùng:

```
example.com/desserts/chocolate-pie
```

Mặc dù không cần thiết phải có cấu trúc URL, nhiều nghiên cứu về tỷ lệ nhấp cho thấy rằng, khi được lựa chọn giữa một URL và một URL ngắn hơn, người tìm kiếm thường thích các URL ngắn hơn. Giống như thẻ tiêu đề và meta mô tả quá dài, URL quá dài cũng sẽ bị cắt bằng dấu chấm lửng. Giảm thiểu độ dài, cả bằng cách bao gồm ít từ hơn trong tên trang của bạn và xóa các thư mục con không cần thiết, giúp URL của bạn dễ sao chép và dán hơn, cũng như có thể nhấp nhiều hơn.

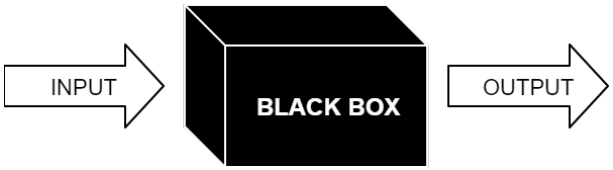
Ngoài ra, Google khuyến nghị rằng tất cả các trang web đều có giao thức bảo mật (phiên bản nâng cấp trong phiên bản https là viết tắt của cụm từ bảo mật). Để đảm bảo rằng các URL của bạn đang sử dụng giao thức `https://` thay vì `http://`, bạn phải có chứng chỉ SSL. Kể từ tháng 7 năm 2018, Google Chrome hiển thị "không bảo mật" cho tất cả các trang web HTTP, điều này có thể khiến các trang web này xuất hiện không đáng tin cậy đối với khách truy cập và dẫn đến việc họ rời khỏi trang web.

Chương IV

Thiết kế giải pháp

1 Phương pháp kiểm tra Black Box

Theo định nghĩa[12], phương pháp black box còn được biết đến là kiểm tra hành vi, là một phương pháp kiểm thử phần mềm trong đó cấu trúc/thiết kế/hiện thực của đối tượng đang được kiểm tra không được biết đến bởi người kiểm tra.



Hình IV.1: Phương pháp kiểm tra Black Box

Ứng dụng của chúng tôi sẽ áp dụng phương pháp kiểm tra này để đánh giá SEO cho website mà người dùng nhập vào. Cụ thể, ứng dụng sẽ tìm đến và tải về mã nguồn trang web, sau đó phân tích các thông tin nhận được, so sánh với các tiêu chí mà chúng tôi quy ước sẵn và trả về giao diện hiển thị kết quả cho người dùng.

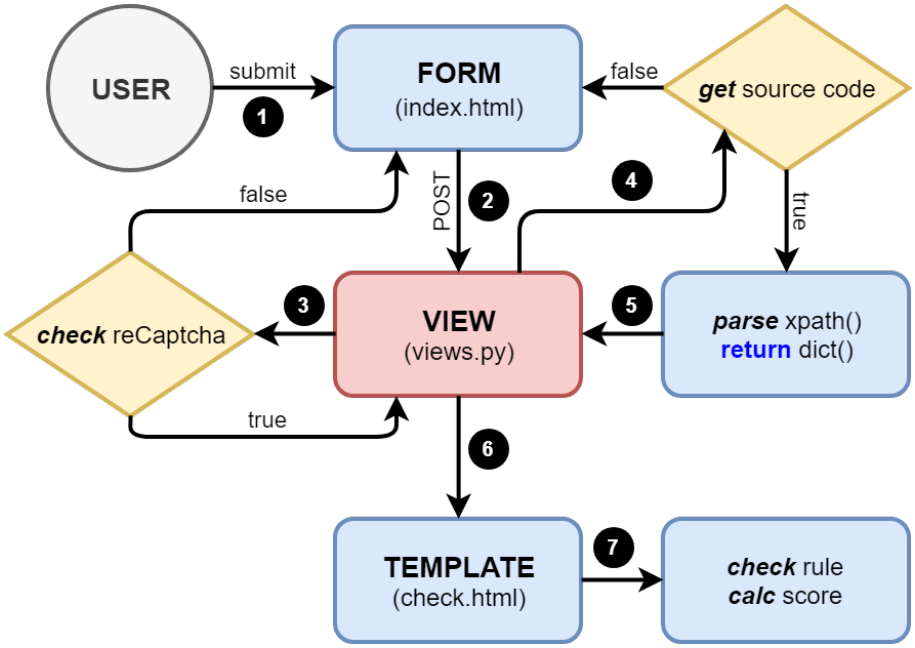
Với thiết kế này, người dùng sẽ không cần phải có kiến thức về lập trình. Ứng dụng của chúng tôi sẽ thay người dùng làm việc đó. Do đó mang lại trải nghiệm thuận tiện cho người dùng.

Tuy nhiên, với giải pháp này, ứng dụng của chúng tôi sẽ xuất hiện vài khuyết điểm. Do không biết được mã nguồn chính xác của trang web, nên đôi khi trình phân tích mã nguồn của chúng tôi sẽ không ổn định, dẫn đến việc đánh giá sẽ không được chính xác hoàn toàn. Đối với các trang web được sinh ra bằng JavaScript, hiện tại thư viện phân tích mã nguồn của chúng tôi không thể tải về được cú pháp, do đó không thể kiểm tra được những trang web thuộc loại này.

Nhìn chung, với việc áp dụng phương pháp kiểm tra trang web theo cơ chế black box sẽ mang đến trải nghiệm thuận tiện cho người dùng ứng dụng của chúng tôi.

2 Kịch bản người dùng

Chúng tôi xây dựng nên kịch bản của ứng dụng dành cho người dùng dựa trên mô hình MTV (Model - Template - View) của Django. Chúng tôi sử dụng sơ đồ bên dưới để trình bày về mô hình ứng dụng của chúng tôi.



Hình IV.2: Kịch bản người dùng

1. Người dùng truy cập vào trang chủ của ứng dụng được lưu ở file `index.html`, tại đây hiển thị thẻ `input` để người dùng nhập url website cần kiểm tra.

2. Khi người dùng bấm vào nút gửi, form sẽ gửi tín hiệu về `VIEW` với giao thức `POST`.

3. Tại đây, chúng tôi có hàm để kiểm tra xem người dùng đã xác thực reCaptcha chưa. Nếu đúng, sẽ đi tiếp đến bước kế tiếp. Ngược lại, chúng tôi sẽ chuyển hướng người dùng về lại trang chủ và thông báo lỗi xác thực reCaptcha.

4. Khi việc kiểm tra reCaptcha thành công, chúng tôi sẽ tiến hành truy vấn đến liên kết mà người dùng nhập vào, sau đó lấy nội dung source code bằng cách sữ dụng thư viện `requests`. Tuy nhiên, sẽ có trường hợp liên kết url của người dùng nhập vào bị sai, hoặc vì một lý do nào đó mà thư viện của chúng tôi không thể lấy source code được. Do đó, chúng tôi sử dụng cấu trúc `try/except` và trả về `false` nếu liên kết gặp lỗi.

5. Tại đây, sau khi đã lấy được nội dung source code của trang web, chúng tôi cần định dạng lại bằng thư viện `lxml` để có thể thực hiện các toán tử `xpath` phân tích các thành phần trang web từ source code. Chúng tôi khai báo biến `value` với kiểu dữ liệu là `dict()` để lưu trữ kết quả sau khi phân tích, với mỗi tiêu chí đánh giá tương ứng với mỗi khóa riêng biệt trong biến `value`. Sau hàm xử lý này, biến `value` được trả về để tiếp tục quá trình triển khai trong ứng dụng.

6. Sau khi nhận được dữ liệu về các yếu tố cần đánh giá, tiếp theo ứng dụng của chúng tôi sẽ truyền những giá trị này sang phía giao diện. Cụ thể, phần giao diện đảm nhiệm xử lý những dữ liệu này nằm ở file `check.html`.

7. Với từng giá trị nhận được, chúng tôi sử dụng các thẻ hỗ trợ của Django để tiến hành so sánh với những tiêu chí về đánh giá web. Sau cùng, dựa trên những kết quả so sánh, chúng tôi sẽ sử dụng JavaScript để tính toán điểm số cho trang web dựa trên trọng số và hiển thị lên giao diện người dùng.

3 Giao diện người dùng

Chúng tôi sử dụng tính kế thừa giao diện trong Django để thiết kế nên mô hình cho ứng dụng của mình. Ngoài ra, để tạo giao diện đa nền tảng, tối ưu với các thiết bị di động, chúng tôi sử dụng Bootstrap để quản lý tính responsive cho trang web. Bên cạnh đó, chúng tôi sử dụng Font Awesome để hiển thị icon trong trang web.

Trong dự án của chúng tôi, sẽ có hai phần ứng dụng con.

- Checkweb: đây là ứng dụng trọng tâm cho dự án của chúng tôi, nơi chúng tôi xây dựng các hàm kiểm tra và đánh giá SEO cho trang web người dùng cần kiểm tra.
- Tips: sẽ là nơi chúng tôi đăng những bài viết về các tiêu chuẩn SEO, ứng dụng này chủ yếu là nhận truy vấn của người dùng và hiển thị giao diện web.

base.html

Đây là bộ khung cho ứng dụng, được chia thành ba khối cơ bản là `title` chứa tiêu đề, `content` chứa nội dung, `script` chứa các mã JavaScript.

header.html, footer.html

Hai file giao diện này có nội dung là hiển thị thanh menu ở đầu trang web và thông tin về dự án ở cuối trang. Thay vì được viết trực tiếp trong file `base.html`, chúng tôi chia nhỏ từng phần ra để code của chúng tôi được cấu trúc rõ ràng hơn. Sau cùng, chúng được `include` lại vào file `base.html` tại vị trí tương ứng cần hiển thị.

index.html

File này đóng vai trò là trang chủ trong ứng dụng. Chúng tôi hiển thị một form để người dùng nhập vào URL cần kiểm tra.

Để bảo vệ website chúng tôi tránh liên tục spam truy vấn đến các trang web khác. Chúng tôi sử dụng phương thức POST cho form và sử dụng thêm reCAPTCHA nhằm xác thực người dùng.

about.html, contact.html

Hiển thị phần giới thiệu về ứng dụng của chúng tôi đến người dùng và thông tin liên hệ.

check.html

Trang này chúng tôi dùng để hiện thị thông tin được phân tích từ website của người dùng nhập vào. Kết quả sẽ được hiển thị bằng thẻ `table` gồm các cột:

- Tiêu chí: Hiển thị các mục mà chúng tôi xem xét trang web người dùng, như là tiêu đề, mô tả và các thẻ khác.
- Kết quả: Sử dụng icon từ Font Awesome để cho người dùng biết tiêu chí đó có đạt yêu cầu hay không. Nếu đạt yêu cầu thì sẽ hiển thị icon có dấu tích xanh lá, ngược lại trang web hiển thị dấu x đỏ.
- Chi tiết: Mục này chúng tôi dùng để liệt kê ra nội dung được lấy từ trang web của người dùng, ví dụ như nội dung của thẻ tiêu đề, mô tả, hình ảnh...

Ngoài ra, chúng tôi sử dụng đoạn JavaScript để tính toán điểm cho website dựa trên kết quả kiểm tra, màu sắc sẽ thay đổi theo từng thang điểm:

- $[80, 100]$: Màu xanh lá.
- $[50, 80)$: Màu vàng.
- $[0, 50)$: Màu đỏ.

Nút Trở lại được đặt ở cuối bảng cho phép người dùng quay lại trang chủ để có thể kiểm tra trang web khác.

4 Bảng đánh giá các tiêu chí SEO

Trọng số có ký hiệu '*' sẽ có giá trị dựa theo tỉ lệ, giá trị càng lớn nếu vi phạm càng nhiều và được quy về mức trọng số là 5.

Công thức tính điểm: $\frac{tongtrongso - trongsovipham}{tongtrongso} * 100$

Tiêu chí	**Trọng số**	**Chi tiết**
Tiêu đề	5	Độ dài lớn hơn 0 và nhỏ hơn 65
Mô tả	5	Độ dài lớn hơn 0 và nhỏ hơn 160
Favicon	5	Có ảnh favicon trong trang web
Robots	5	Có thuộc tính robots trong trang web
Thẻ h1	5	Có thẻ h1 trong trang web
Thẻ h2	4	Có thẻ h2 trong trang web
Robots.txt	5	Có liên kết robots.txt trong trang web
Sitemap	5	Có liên kết sitemap trong trang web
Lỗi liên kết	5*	Không có lỗi liên kết trong trang web
CSS nội tuyến	3	Không có thuộc tính CSS nội tuyến trong trang web
Thuộc tính alt	5*	Có thuộc tính alt trong hình ảnh

Bảng IV.1: Bảng đánh giá các tiêu chí SEO

Chương V

Hiện thực

Phần này chúng tôi sẽ trình bày quá trình chúng tôi xây dựng và phát triển để tạo thành sản phẩm hoàn chỉnh. Dựa trên nền tảng lý thuyết được giới thiệu ở phần trước, chúng tôi sẽ áp dụng chúng vào những hướng dẫn bên dưới. Ở phần cuối của chương này sẽ là hướng dẫn về cách triển khai ứng dụng lên Internet bằng việc sử dụng máy chủ do trang Heroku cung cấp.

Lưu ý, trong bài báo cáo này chúng tôi đang sử dụng môi trường lập trình là Windows nên có thể có những câu lệnh sẽ khác với MacOS, Linux hay những môi trường lập trình khác.

1 Khởi tạo project

1.1 Cài đặt Python

Để cài đặt Python vào máy tính, bạn truy cập vào trang chủ của Python là `https://www.python.org`, sau đó chọn mục `Downloads` và cài đặt theo môi trường hệ điều hành trên máy của bạn.

Với chúng tôi, chúng tôi chọn cách cài đặt Python gián tiếp bằng Miniconda. Theo cách này, chúng tôi có thể dễ dàng nâng cấp phiên bản Python mà không phải cài lại hoàn toàn. Tương tự như cách cài đặt trực tiếp, bạn truy cập vào trang `https://docs.conda.io/en/latest/miniconda.html`, chọn tải về file cài đặt phù hợp với hệ điều hành của mình và tiến hành cài đặt theo hướng dẫn.

Sau khi cài đặt hoàn tất, để kiểm tra xem Python đã cài thành công, bạn mở trình `command line` và nhập vào:

```
\>python --version
```

Kết quả trả về có thể là:

```
Python 3.7.3
```

1.2 Tạo thư mục và thiết lập môi trường

Đến đây bạn đã cài đặt thành công Python vào máy tính. Tiếp theo chúng tôi sẽ hướng dẫn bạn tạo môi trường cho từng dự án. Việc tạo môi trường ảo Python cho từng dự án sẽ đảm bảo các phiên bản thư viện trong các dự án khác nhau mà không làm ảnh hưởng đến những thư viện cài đặt gốc.

Chúng tôi quyết định đặt tên dự án của mình là `check-seo`, do đó chúng tôi sẽ tạo thư mục mới để lưu trữ code.

Để tạo mới thư mục, bạn có thể click chuột phải → chọn New → chọn Folder → đặt tên `check-seo`.

Ở đây, chúng tôi sẽ tạo thư mục bằng command line trong Windows PowerShell. Để mở PowerShell tại thư mục hiện tại, bấm giữ phím `Shift` → click chuột phải chọn Open PowerShell window here → nhập lệnh sau để tạo thư mục mới:

```
\>mkdir check-seo
```

Di chuyển khung làm việc vào thư mục project vừa tạo.

```
\>cd check-seo
```

Tại đây, chúng tôi sẽ tiến hành cài đặt môi trường để lập trình cho ứng dụng Python của mình. Trong khung cửa sổ của PowerShell, nhập lệnh sau để cài đặt môi trường:

```
\>python -m venv ./venv
```

Thư mục mới được tạo ra có tên là `venv` chứa các tập tin hệ thống giúp tạo môi trường ảo cho Python. Để kích hoạt môi trường ảo, sử dụng câu lệnh:

```
\>.\venv\Scripts\activate
```

Khi kích hoạt môi trường thành công, sẽ có phần thông tin `(venv)` hiển thị ở đầu mỗi dòng lệnh, giống như `(venv)\>`

1.3 Cài đặt các thư viện cần thiết

Sau khi tạo xong thư mục và kích hoạt xong môi trường ảo, tiếp theo chúng tôi sẽ tiến hành cài đặt các thư viện phục vụ cho dự án của chúng tôi.

Trong thư mục `check-seo`, tạo tệp mới có tên là `requirements.txt` sẽ chứa thông tin về thư viện và phiên bản sử dụng, để xem thông tin cụ thể của từng thư viện, bạn có thể tìm kiếm chúng trên kho lưu trữ công khai của Python là `https://pypi.org`. Nội dung của file `requirements.txt` như sau:

```
Django==2.2
lxml==4.3.3
requests==2.21.0
```

Để tiến hành cài đặt thư viện được liệt kê trong file `requirements.txt`, sử dụng câu lệnh:

```
(venv)\>pip install -r requirements.txt
```

Trình cài đặt thư viện Python sẽ tiến hành tải về và cài đặt trong môi trường ảo mà chúng tôi đã kích hoạt. Ngoài thư viện chính, trình cài đặt còn tải thêm những thư viện khác bổ trợ đi theo từng thư viện. Để kiểm tra các gói đã cài đặt, sử dụng lệnh:

```
(venv)\>pip freeze list
```

Kết quả trả về có thể như sau:

```
certifi==2019.3.9
chardet==3.0.4
Django==2.2
idna==2.8
lxml==4.3.3
pytz==2019.1
requests==2.21.0
sqlparse==0.3.0
urllib3==1.24.3
```

1.4 Tạo project và xây dựng app Django

Sau khi cài đặt xong những thư viện cần thiết, tiếp theo, chúng ta sẽ tiến hành tạo mới project Django có tên là `src` trong thư mục `check-seo` bằng câu lệnh:

```
(venv)\>django-admin startproject src .
```

Sau khi thực thi thành công câu lệnh trên thì sẽ tạo ra thư mục `src` chứa các file cài đặt cho project và file `manage.py` giúp quản lý các thao tác command line cho ứng dụng.

Theo thiết kế của ứng dụng, chúng tôi sẽ tạo thêm 2 app cho project là `checkweb` quản lý chính cho việc thu thập, đánh giá SEO cho website và `tips` sẽ đảm nhiệm hiển thị các bài đăng về thủ thuật SEO. Để tạo app, sử dụng lần lượt 2 câu lệnh sau đây:

```
(venv)\>python .\manage.py startapp checkweb
(venv)\>python .\manage.py startapp tips
```

Để quản lý các file giao diện `html`, chúng tôi tạo thêm thư mục `templates` tại thư mục chính của project. Tiếp theo, chúng tôi đi vào thư mục `src` sau đó tạo thêm thư mục có tên là `static_venv` đảm nhiệm việc lưu trữ các file CSS, JavaScript và các thư viện bên ngoài như Bootstrap, jQuery.

Sau khi tạo mới app, thư mục `templates` và `static_venv`, cần phải đăng ký vào cấu

hình để project hiểu được cấu trúc của chương trình tại file `settings.py` trong thư mục `src`.

Để khai báo app, tìm đến dòng `INSTALLED_APPS` và thêm đoạn code bên dưới vào hàng cuối cùng, kết quả sẽ tương tự như:

```
INSTALLED_APPS = [
    "django.contrib.admin",
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",

    "checkweb.apps.CheckwebConfig",
    "tips.apps.TipsConfig",
]
```

Cấu hình templates cho project tại khóa `DIRS` của mục `TEMPLATES`:

```
TEMPLATES = [
    {
        "BACKEND": "django.template.backends.django.DjangoTemplates",
        "DIRS": [os.path.join(BASE_DIR, "templates")],
        "APP_DIRS": True,
        "OPTIONS": {
            "context_processors": [
                "django.template.context_processors.debug",
                "django.template.context_processors.request",
                "django.contrib.auth.context_processors.auth",
                "django.contrib.messages.context_processors.messages",
            ],
        },
    },
]
```

Cuối cùng trong phần này, chúng tôi sẽ cấu hình phần `static` để hiển thị các file CSS, JavaScript,... tại mục `STATIC_URL`, chúng tôi sẽ thêm đoạn code vào để được kết quả như dưới đây:

```
STATIC_URL = "/static/"
STATIC_ROOT = os.path.join(BASE_DIR, "static")
STATICFILES_DIRS = [os.path.join(BASE_DIR, "src/static_venv")]
```

2 Cấu trúc giao diện Templates

2.1 Xử lý Frontend

Phần này, chúng tôi sẽ trình bày về cách chúng tôi phân chia các file giao diện `html` trong thư mục `templates` được tạo ở hướng dẫn bên trên.

Đầu tiên, chúng tôi tạo file `base.html` có chức năng là khung sườn cho toàn bộ giao diện với khả năng kết nạp các file khác để giúp chia nhỏ giao diện thành các phần có chức năng riêng biệt. Việc chia nhỏ giao diện thành các file thành phần giúp chúng tôi có thể quản lý code tốt hơn và tránh rối rắm khi nếu lưu quá nhiều dòng code trong một file duy nhất.

Tiếp theo, chúng tôi tạo thêm hai file nữa có tên là `header.html` và `footer.html`. Quay lại file `base.html`, ta có cấu trúc code đơn giản như sau:

```
<!DOCTYPE html>
<html lang="vi">
<head>
    <title>{% block title %}{% endblock %}</title>
</head>
<body>
    <!-- Header -->
    {% include "header.html" %}
    <main>
        {% block content %}{% endblock %}
    </main>
    <!-- Footer -->
    {% include "footer.html" %}
    {% block script %}{% endblock %}
</body>
</html>
```

Để có thể thay đổi nội dung theo từng giao diện, chúng tôi đã sử dụng ba block là `title`, `content` và `script`, do đó chúng tôi sẽ thay đổi nội dung ở hai block này tùy theo mục đích mà chúng tôi muốn hướng đến.

Ở file `header.html` và tương tự là file `footer.html` sẽ có nội dung cơ bản như sau:

```
<header>
    <nav>
        <a href="/"><h1>Danh Gia Web</h1></a>
    </nav>
</header>
```

```
<footer>
    <div>DGW &copy; 2018 - {% now "Y" %}</div>
</footer>
```

Sau khi cấu trúc xong bộ khung cho giao diện, tiếp theo chúng tôi sẽ xây dựng giao diện cho từng app dựa trên những gì đã thiết lập.

Tại thư mục `templates` tạo thêm hai thư mục mới có tên trùng với hai app đã tạo là `checkweb` và `tips`. Chúng tôi sẽ đi sâu vào việc tạo giao diện cụ thể cho phần app `checkweb` vì tại đây là trọng tâm chính của ứng dụng đánh giá website của chúng tôi. Phần giao diện `tips` có phần đơn giản hơn nhiều và bạn có thể thiết lập dựa theo hướng dẫn ở phần `checkweb`. Hơn nữa, chúng tôi sẽ cung cấp mã nguồn ở phần ***Kết luận***, do đó bạn có thể tự nghiên cứu dựa theo những đoạn code của chúng tôi.

Mở thư mục `checkweb` vừa tạo, dựa theo kiến trúc ở phần ***Thiết kế giải pháp***, chúng tôi tiến hành tạo thêm các file mới là `index.html`, `about.html`, `contact.html` và `check.html`.

`index.html`

```
{% extends "base.html" %}
{% block title %}Trang Chu{% endblock %}
{% block content %}
<div>Noi dung Trang chu</div>
{% endblock %}
```

Các file còn lại cũng có cấu trúc tương tự, với nội dung ở các block sẽ khác nhau tùy theo mỗi file. Ở đây chúng ta quan tâm đến hai file đó là `index.html` và `check.html` sẽ được nhắc đến ở những phần sau.

2.2 Xử lý Backend

Sau khi tạo xong các file `html`, tiếp theo chúng tôi sẽ tiến hành cấu hình để xử lý phần backend của ứng dụng.

Đầu tiên, chúng tôi sẽ quản lý các url để hiển thị file giao diện trong ứng dụng tại file `urls.py` trong thư mục `src`. File `urls.py` sẽ có nội dung như sau:

```
from django.urls import path, include

urlpatterns = [
    path("", include("checkweb.urls")),
    path("thu-thuat/", include("tips.urls")),
]
```

Tiếp theo, chúng tôi thực hiện việc kết nối giữa truy vấn url và giao diện tại file `views.py` trong thư mục app `checkweb` được tạo ra khi chạy lệnh `startapp` lúc đầu. Django hỗ trợ việc kết nối này đơn giản và tiết kiệm dòng code hơn nhiều bằng chế độ ***Class-based views***. Chúng tôi sẽ sử dụng để tạo giao diện cho trang chủ, giới thiệu, liên hệ và trang kiểm tra.

```
from django.views.generic import TemplateView

class IndexView(TemplateView):
```

```
    template_name = "checkweb/index.html"

class AboutView(TemplateView):
    template_name = "checkweb/about.html"

class ContactView(TemplateView):
    template_name = "checkweb/contact.html"

class CheckView(TemplateView):
    template_name = "checkweb/check.html"
```

Trong thư mục `checkweb` hiện tại, tạo file `urls.py` để quản lý các url trong ứng dụng được gọi từ hàm `include` ở file `urls.py` trong thư mục `src`.

```
from django.urls import path
from . import views

urlpatterns = [
    path("", views.IndexView.as_view(), name="index"),
    path("gioi-thieu/", views.AboutView.as_view(), name="about"),
    path("lien-he/", views.ContactView.as_view(), name="contact"),
    path("kiem-tra/", views.CheckView.as_view(), name="check"),
]
```

Với cách tương tự, chúng tôi sẽ tiến hành cấu hình đối với app `tips`. Chi tiết, bạn có thể xem trên mã nguồn của chúng tôi.

3 Hiện thực chức năng đánh giá website

Sau khi thiết đặt xong phần giao diện templates, tiếp theo chúng tôi sẽ đi sâu vào xây dựng các hàm thực hiện chức năng chính cho project của mình. Đó là công việc nhận vào url trang web mà người dùng muốn đánh giá, kiểm tra captcha và url, sau đó trả về kết quả cho người dùng.

Theo cách thông thường, các hàm này được viết trong file `views.py` ở thư mục app `checkweb`. Tuy nhiên, để dễ dàng quản lý hơn, chúng tôi sẽ tạo một file mới cùng trong thư mục app và đặt tên là `functions.py` sẽ chứa những đoạn code phục vụ cho việc phân tích và kiểm tra cho ứng dụng của chúng tôi. Do việc tách ra như vậy, tại file `views.py` chúng tôi sẽ nhúng file này bằng dòng code sau được thêm vào ở đầu file:

```
from . import functions
```

3.1 Tạo form nhập và xử lý xác thực reCaptcha

Tạo giao diện nhập url

Trước tiên, chúng tôi sẽ tạo giao diện form nhập để người dùng gửi url muốn kiểm tra vào ứng dụng của chúng tôi. Do đó, chúng tôi sẽ tiến hành thêm thẻ `form` vào file `index.html` trong thư mục `templates/checkweb`.

```
<form action="{% url 'check' %}" method="POST">
    {% csrf_token %}
    <input type="url" name="url" id="url" required>
    <button id="submit">Submit</button>
</form>
```

Form của chúng tôi sẽ sử dụng giao thức `POST`, được xử lý trong backend và xuất về đường dẫn được khai báo trong `urls.py` có tên là `check`, cụ thể ở đây sẽ là file `check.html`.

Sau khi đã tạo xong thẻ `form`, tiếp đến chúng tôi truy cập vào trang `https://www.google.com/recaptcha/admin/` để lấy thông tin về khóa API của reCaptcha. Khóa này được dùng để kích hoạt tính năng của reCaptcha. Theo hướng dẫn trên trang admin, chúng tôi sẽ chèn tiếp thẻ `div` bên trong thẻ `form` và bên dưới thẻ `button`.

```
<div class="g-recaptcha" data-sitekey="<SITE_KEY>" data-callback="onSubmit"
    data-badge="bottomleft" data-size="invisible"></div>
```

Ngoài ra, để reCaptcha có thể hoạt động được, chúng tôi sẽ chèn thêm đoạn code để xử lý JavaScript. Đoạn code này chúng tôi sẽ đặt trong khối `script` được kế thừa từ khung `base.html` mà chúng tôi đã khai báo trước đó.

Đoạn JavaScript bên dưới có nhiệm vụ bắt sự kiện click vào nút Summit và tạo ra đoạn mã xác thực reCaptcha mà chúng tôi sẽ xử lý tiếp ở phần tiếp theo:

```
{% block script %}
<script src="https://www.google.com/recaptcha/api.js" async defer></script>
<script>
var sm = document.getElementById("submit");
sm.onclick = validate;

function validate(event) {
    event.preventDefault();
    grecaptcha.execute()
};

function onSubmit(token) {
    sm.onclick = null;
    sm.click();
}
</script>
{% endblock %}
```

Hàm xác thực reCaptcha

Chúng tôi sẽ viết hàm này trong file `functions.py` có tên là `reCaptcha`.

- Đầu vào sẽ là mã được sinh ra từ việc xác thực của reCaptcha được lưu với tên là `g-recaptcha-response` từ truy vấn `POST` và IP của người dùng truy cập.
- Đầu ra là kết quả sau từ đánh giá của reCaptcha sau khi gửi các giá trị đầu vào lên máy chủ. Sẽ có hai giá trị là `true` và `false` tương ứng với kết quả xác thực người dùng là thành công hay thất bại.

Khi đăng ký reCaptcha, chúng ta còn nhận được ngoài khóa `SITE_KEY` thì còn một khóa nữa là `SECRET_KEY`. Để có thể dễ dàng kiểm soát các khóa trong ứng dụng, chúng tôi sẽ lưu khóa bí mật này trong file `settings.py` ở thư mục `src`.

```
# Google reCAPTCHA secret key
# https://developers.google.com/recaptcha/docs/verify/

GOOGLE_RECAPTCHA_SECRET_KEY = "<SECRET_KEY>"
```

Và để sử dụng biến `GOOGLE_RECAPTCHA_SECRET_KEY` thì Django có hỗ trợ bằng cách thêm dòng code sau vào đầu file `functions.py`:

```
from django.conf import settings
```

Bên cạnh đó, để gửi dữ liệu lên máy chủ của reCaptcha, chúng tôi cần thêm sự hỗ trợ của thư viện Python là `requests` mà chúng tôi đã cài đặt từ lúc mới thiết lập môi trường của ứng dụng.

```
import requests
```

Sau khi thêm các thư viện và ý tưởng xử lý thì hàm xử lý reCaptcha sẽ có nội dung như sau:

```
data = {
    "secret": settings.GOOGLE_RECAPTCHA_SECRET_KEY,
    "response": response,
    "remoteip": userIP,
}
verify = requests.post(
    "https://www.google.com/recaptcha/api/siteverify", data=data)
result = verify.json()
return result["success"]
```

Hàm `reCaptcha` được xây dựng xong thì tiếp theo, chúng tôi sẽ quay lại file `views.py` để gọi lại hàm này và xử lý nó.

Việc xử lý captcha nằm trong class `CheckView` mà chúng tôi đã tạo trước đó. Theo quy ước trong Django thì để xử lý truy vấn `POST`, chúng tôi viết thêm hàm `post` trong class `CheckView`.

```
def post(self, request):
    url = request.POST["url"]
    if reCaptcha(request.POST["g-recaptcha-response"],
        request.META["REMOTE_ADDR"]):
        # code check web here
        return render(request, "checkweb/check.html")
    return redirect("/")
```

Nếu kiểm tra captcha thành công thì ứng dụng sẽ trả về trang giao diện trong file `check.html`. Nếu xác thực thất bại thì sẽ chuyển hướng trở lại trang chủ. Chúng tôi có sử dụng hàm chuyển hướng trang `redirect`. Thêm vào đầu file `views.py` đoạn code sau để chèn hàm này:

```
from django.shortcuts import redirect
```

Đến đây, chúng tôi đã giải quyết xong vấn đề kiểm tra xác thực người dủng bằng reCaptcha.

3.2 Phân tích cấu trúc nội dung website

Lấy source code và phân tách trang web

Theo cách hoạt động ứng dụng của chúng tôi sẽ phân tích trang web bằng cách lấy source code tương đương với tính năng View source trên các trình duyệt web. Để xử lý chức năng này, chúng tôi xây dựng hàm `parsing` trong file `functions.py`.

- Đầu vào sẽ là url của trang web cần kiểm tra
- Đầu ra là biến có kiểu `dict()` chứa nội dung các yếu tố mà chúng tôi phân tách ra được từ source code.

Ngoài thư viện `requests`, chúng tôi cần sự trợ giúp thêm của thư viện `lxml`.

```
from lxml import html
```

Sau khi thêm vào thư viện cần thiết, bên dưới là đoạn code dùng để lấy source code dựa trên url trang web.

```
def parsing(url):
    try:
        page = requests.get(url, timeout=5)
        content = html.fromstring(page.content.decode("utf-8"))
    except BaseException:
        return False
```

Vì để tránh chương trình bị lỗi khi người dùng nhập vào url mà chúng tôi không thể kiểm tra được nên chúng tôi đã sử dụng cấu trúc `try/except` trong Python để giải quyết

vấn đề này. Ngoài ra, nếu truy vấn vượt quá năm giây cũng sẽ bị xem là lỗi không kiểm tra được. Cuối cùng, biến `content` được decode theo chuẩn mã `utf-8` để hiển thị font chữ không bị lỗi.

Tiếp theo, cũng trong hàm `parsing`, chúng tôi khai báo biến `value` có kiểu là `dict()` để lưu các yếu tố được phân tách. Bên cạnh đó là các biến dùng để phục vụ làm input cho các hàm về sau.

```
value = dict()
domain = url.split("/")[2]
urlDomain = url.split("/")[0] + "//" + domain
```

Việc phân tách cấu trúc trong source code, chúng tôi sử dụng phương pháp `xpath`. Để dễ dàng quản lý, chúng tôi chia ra hai loại là thành phần chỉ có một giá trị, ví dụ thẻ `title` sẽ được lưu cấu trúc `xpath` trong biến `elm`, những thành phần có hơn một giá trị, ví dụ thẻ `img` sẽ được lưu cấu trúc `xpath` trong biến `elms`. Cả hai biến `elm` và `elms` đều cò kiểu là `dict()`.

```
elm = {
    "title": "//title/text()",
    "description": "//meta[@name='description']/@content",
    "favicon": "//link[contains(@rel, 'icon')]/@href",
    "robotsMeta": "//meta[@name='robots']/@content",
}
elms = {
    "h1Tags": "//h1//text()",
    "h2Tags": "//h2//text()",
    "aTags": "//a/@href",
    "cssInlines": "//@style/..",
    "imgTags": "//img",
}
```

Sau khi đã có cấu trúc `xpath` của từng thuộc tính, tiếp theo chúng tôi sẽ sử dụng vòng lặp `for` để lấy dữ liệu dựa trên cấu trúc có sẵn. Do có hai cấu trúc khác khau, nên chúng tôi cũng sẽ dùng hai vòng lặp để xử lý.

```
for k, v in elm.items():
    try:
        value[k] = content.xpath(v)[0]
    except BaseException:
        value[k] = None

for k, v in elms.items():
    try:
        value[k] = content.xpath(v)
    except BaseException:
        value[k] = None
```

Sau khi chạy xong hai vòng lặp này thì biến `value` đã lưu được khá nhiều dữ liệu từ việc phân tách source code dựa trên `xpath`. Tuy nhiên, có một số yếu tố sẽ không đúng chuẩn do những website khác nhau dẫn đến có nhiều kết quả không như ý. chúng tôi sẽ

giải quyết vấn đề này bằng các hàm bổ sung và được chúng tôi đề cập trong những phần tiếp theo.

Cuối cùng, chúng tôi trả về kết quả của biến `value` và chuyển qua file `views.py` để xử lý dữ liệu trả về.

```
return value
```

Tiếp theo, chúng tôi sẽ xử lý dữ liệu của `value` ở file `views.py`. Đoạn code bên dưới bao gồm luôn phần xử lý xác thực reCaptcha được chúng tôi trình bày ở phần trước.

```
def post(self, request):
    url = request.POST["url"]
    if reCaptcha(request.POST["g-recaptcha-response"],
        request.META["REMOTE_ADDR"]):
        context = parsing(url)
        if context:
            context["url"] = url
            return render(request, "checkweb/check.html", context)
        return redirect("/")
    return redirect("/")
```

Biến `context` sẽ nhận kết quả trả về từ hàm **parsing**. Nếu không có giá trị, nghĩa là việc kiểm tra url gặp lỗi, do đó chúng tôi thay vì trả về truy vấn đến trang giao diện `check.html`, mà sẽ chuyển hướng trở về trang chủ.

3.3 Xử lý nâng cao các cấu trúc trong website

Favicon

Vấn đề xảy ra ở đây là do định dạng cấu trúc đường dẫn đến hình ảnh của các website khác nhau có phần khác biệt, và do `xpath` chỉ lấy nội dung bên trong thẻ nên chúng tôi cần phải định dạng lại cấu trúc đường dẫn này. Ví dụ, liên kết hình ảnh bắt đầu bằng '`/`', để hiển thị đúng thì cần phải gắn thêm phần tên miền vào trước liên kết. Sau đây là hàm xử lý trong file `functions.py`

```
def getLinkImg(elm, urlDomain):
    if elm and elm[:2] not in {"ht", "//"}:
        elm = urlDomain + "/" + elm.lstrip("/")
    return elm
```

Chúng ta sẽ gán lại khóa này trong biến `value`:

```
value["favicon"] = getLinkImg(value["favicon"], urlDomain)
```

Thẻ H1, H2

Trong quá trình phát triển, chúng tôi nhận thấy nội dung của các thẻ này thường xuyên bị thừa các ký tự khoảng trắng ở đầu hoặc cuối dòng. Đôi khi có vài thẻ có nội dung rỗng. Nguyên nhân là vì bên trong các thẻ này còn được lồng thêm các thẻ khác như thẻ `a`, `span`. Do đó chúng tôi cần viết thêm hàm để dọn sạch các khoảng trống bị dư thừa, cũng như các thẻ không chứa nội dung.

```
def cleanElms(elms):
    if elms:
        for idx, _ in enumerate(elms):
            elms[idx] = elms[idx].strip()
        elms = list(filter(None, elms))
    return elms
```

Sau đó, chúng tôi gán lại dữ liệu cho các thẻ này trong hàm `parsing`:

```
value["h1Tags"] = cleanElms(value["h1Tags"])
value["h2Tags"] = cleanElms(value["h2Tags"])
```

Robots.txt

Mặc định, nếu trang web sử dụng file này thì sẽ có cấu trúc đường dẫn là `https://ten-mien/robots.txt`. Do đó, chúng tôi sẽ sử dụng thư viện `requests` để truy vấn đến liên kết này và kiểm tra xem mã trạng thái trả về có phải là 200 không, cũng như có kiểu nội dung là `plain`.

```
def getlinkRobots(urlDomain):
    try:
        value = requests.get(urlDomain + "/robots.txt")
    except BaseException:
        return None
    if value.status_code != 200 or "plain" not in
        value.headers["Content-Type"]:
        return None
    return urlDomain + "/robots.txt"
```

Đây là yếu tố mới không có trong biến `value` khi xử lý `xpath`, do đó chúng tôi gán giá trị hàm này vào khóa mới:

```
value["robotsTxt"] = getlinkRobots(urlDomain)
```

Sitemap.xml

Thông thường, cũng tương tự như với `robots.txt`, đường dẫn của `sitemap.xml` sẽ là `https://ten-mien/sitemap.xml`. Tuy nhiên, ở một số các trang web, liên kết `sitemap.xml` không giống như mặc định và đường dẫn đó được khai báo trong file

`robots.txt` có nội dung tương tự như:

```
Sitemap: https://ten-mien/site-map/sitemap.xml
```

Do đó, để kiểm tra đường dẫn `sitemap.xml` của trang web, chúng tôi sẽ xem xét nội dung trong file `robots.txt` trước, nếu không tìm thấy, chúng tôi sẽ tiếp tục kiểm tra với liên kết mặc định. Đối với mỗi liên kết có được, chúng tôi sẽ kiểm tra mã trạng thái trả về có là 200 hay không, cũng như kiểu nội dung là `xml`.

```
def getlinkSitemap(urlDomain, robots):
    try:
        value = requests.get(urlDomain + "/sitemap.xml")
    except BaseException:
        return None
    if robots:
        txt = requests.get(robots).content.decode("utf-8")
        txt = txt.replace("\n", "")
        sitemap = re.findall(r"Sitemap:.*xml", txt)
        if sitemap:
            sitemap = sitemap[0].split("Sitemap: ")[1:]
            return sitemap
    if value.status_code != 200 or "xml" not in value.headers["Content-Type"]:
        return None
    sitemap = [urlDomain + "/sitemap.xml"]
    return sitemap
```

Cũng tương tự yếu tố `robots.txt`, chúng tôi tiếp tục thêm khóa mới vào biến `value`:

```
value["sitemaps"] = getlinkSitemap(urlDomain, value["robotsTxt"])
```

Lỗi liên kết

Đầu tiên, chúng tôi viết hàm dùng để định dạng lại thẻ liên kết lấy được từ `xpath`. Sau đó lọc ra các liên kết trùng nhau, liên kết `tel`, `mailto` và `javascript` cũng được chúng tôi loại ra trước khi kiểm tra liên kết đó có lỗi hay không.

```
def getlinkA(elms, urlDomain):
    idx = 0
    while idx < len(elms):
        elm = elms[idx]
        if elm in ("#", "/") or elm.split(":")[0] in ("tel", "mailto",
            "javascript"):
            elms.pop(idx)
            idx -= 1
        elif elm[:2] not in ("ht", "//"):
            elms[idx] = urlDomain + "/" + elm.lstrip("/")
        elif elm[:2] == "//":
            elms[idx] = "https:" + elm
        idx += 1
    return set(elms)
```

Tiếp theo, chúng tôi viết hàm để truy vấn đến liên kết bằng thư viện `requests` và để hạn chế thời gian phản hồi, chúng tôi đặt thêm tham số `timeout=5` để bỏ qua liên kết nếu nó phản hồi sau quá năm giây.

```
def checkBrokenLink(elm):
    try:
        value = requests.get(elm, timeout=5)
    except BaseException:
        return elm
    if value.status_code != 200:
        return elm
    return None
```

Hàm ở trên chỉ đang xét ở từng liên kết đơn lẻ. Tuy nhiên trong một website có thể có rất nhiều liên kết dẫn đến nếu xét tuần tự từng liên kết thì dẫn đến thời gian chờ sẽ rất lâu. Do đó, chúng tôi đã áp dụng xử lý đồng thời vào quá trình kiểm tra này. Để sử dụng gói hỗ trợ này trong Python, chèn dòng code sau vào đầu file `functions.py`:

```
from concurrent.futures import ThreadPoolExecutor
```

Sau khi chèn xong gói thư viện hỗ trợ, chúng tôi triển khai hàm đồng thời và trả về mảng các liên kết bị lỗi:

```
def getBrokenLink(elms):
    res = list()
    with ThreadPoolExecutor() as executor:
        for elm in executor.map(checkBrokenLink, elms):
            if elm:
                res.append(elm)
    return res
```

Cuối cùng, chúng tôi thêm các giá trị này vào khóa mới trong biến `value`

```
value["aTags"] = getlinkA(value["aTags"], urlDomain)
value["aBrokens"] = getBrokenLink(value["aTags"])
```

CSS nội tuyến

Chúng tôi cần định dạng lại kết quả trả về để nó ngắn gọn hơn nhưng vẫn đảm bảo đầy đủ nội dung mà chúng tôi muốn hiển thị. Bởi vì `xpath` sẽ trả về kết quả là toàn bộ nội dung của thẻ có CSS nội tuyến, chúng tôi sẽ viết hàm để chỉ trả về phần mở thẻ, nơi chứ nội dung của thuộc tính `style` và loại bỏ phần nội dung cũng như phần code đóng thẻ.

```
def getCSSInlines(elms):
    for idx, value in enumerate(elms):
        tmp = html.tostring(value, encoding="utf-8").decode("utf-8")
        elms[idx] = re.search(r"<.*?>", tmp).group()
    return elms
```

Sau đó, chúng tôi gán lại nội dung lại vào biến `value`:

```
value["cssInlines"] = getCSSInlines(value["cssInlines"])
```

Thuộc tính alt

Việc xử lý yếu tố này cũng gần tương đương với cách làm trên. Chúng tôi sẽ kiểm tra các thẻ `img` xem nếu bị thiếu thành phần thuộc tính `alt` thì sẽ trả về thẻ `img` đó.

```
def checkMissAlts(elms):
    res = list()
    for elm in elms:
        tmp = html.tostring(elm, encoding="utf-8").decode("utf-8")
        if not re.search(r"alt=(\'|\").+(\'|\")", tmp):
            res.append(re.search(r"<img.*?>", tmp).group())
    return res
```

Xếp hạng

Chúng tôi sử dụng dữ liệu về xếp hạng trang web bằng cách lấy API từng kho dữ liệu mở của `Open PageRank`. Sau khi đăng ký tài khoản thành công, bạn sẽ nhận được khóa API dùng để truy cập miễn phí lên kho dữ liệu này. Cũng tương tự như khóa khi đăng ký reCaptcha, chúng tôi sẽ lưu khóa này trong file `settings.py` của ứng dụng:

```
# Open PageRank key
# https://www.domcop.com/openpagerank/

OPEN_PAGERANK_KEY = "<API_KEY>"
```

Sau đó, chúng tôi quay trở lại file `functions.py` để hoàn thiện hàm để lấy thông tin xếp hạng.

```
def getPageRank(domain):
    url = "https://openpagerank.com/api/v1.0/getPageRank?domains[0]=" + domain
    headers = {
        "API-OPR": settings.OPEN_PAGERANK_KEY
    }
    data = requests.get(url, headers=headers)
    result = data.json()["response"][0]
    return result["rank"]
```

Cuối cùng là gán giá trị trả về này vào biến `value`:

```
value["pageRank"] = getPageRank(domain)
```

Đến đây chúng tôi đã hoàn thành xong giai đoạn lấy nội dung source code từ trang web người dùng muốn kiểm tra, phân tách các nội dung cần đánh giá bằng `xpath` và định

dạng lại những nội dung này bằng các hàm nâng cao bổ sung.

3.4 Hiển thị kết quả đánh giá ra giao diện

Đây là phần giao diện được xử lý tại file `check.html` mà chúng tôi đã tạo ở những phần trước. Lần này, chúng tôi sẽ đi sâu vào chi tiết ở file này.

Những yếu tố đánh giá chúng tôi sẽ sử dụng thẻ `table` để hiển thị thông tin. Nội dung này được đặt trong block `content`:

```
{% block content %}
<table id="tbCheck">
    <thead>
        <tr>
            <th scope="col" width="15%">Tieu chi</th>
            <th scope="col" width="15%">Ket qua</th>
            <th scope="col" width="70%">Chi tiet</th>
        </tr>
    </thead>
    <tbody>
        <!-- code here -->
    </tbody>
</table>
{% endblock %}
```

Bảng này chúng tôi sẽ chia thành ba cột lần lượt là:

- Tiêu chí: các yếu tố chúng tôi đánh giá trong website của bạn.
- Kết quả: hiển thị icon tương ứng từ `Font Awesome` với ba trạng thái là thành công, thất bại và thông tin.
- Chi tiết: sẽ là nội dung được lấy từ biến `value` mà chúng tôi xử lý ở phần trước, cùng với phần chú thích của chúng tôi để bạn có thể biết được rõ hơn tiêu chí đánh giá của chúng tôi về nội dung này.

Tiếp theo chúng tôi sẽ hiển thị thông tin đánh giá cho từng mục trong phần thân của thẻ `tbody`. Ví dụ, với mục tiêu đề, đoạn code của chúng tôi sẽ có nội dung như sau:

```
<tbody>
    <tr>
        <th scope="row">Tieu de</th>
        <td>
            {% if title and title|length < 65 %}
            <i class="fas fa-check-circle text-success"></i>
            {% else %}
            <i class="fas fa-times-circle text-danger"></i>
            {% endif %}
            <input type="hidden" class="point" value="5">
        </td>
```

```
        <td>
            {% if title %}
            <div>Do dai tieu de trang cua ban la <b>{{ title|length }}</b> ky
                tu. Hau hat cac cong cu tim kiem se cat bot tieu de trang thanh
                65 ky tu.</div>
            <small><i class="fas fa-angle-double-right"></i> {{ title }}</small>
            {% else %}
            <div>Khong tim thay tieu de tren trang cua ban.</div>
            <small><i class="fas fa-angle-double-right"></i><em>
                None</em></small>
            {% endif %}
        </td>
    </tr>
</tbody>
```

Đối với yếu tố đánh giá mà kết quả trả về là mảng danh sách ví dụ như là thẻ `h1` thì chúng tôi sẽ xử lý như sau:

```
<tbody>
    <!-- <tr>...</tr> -->
    <tr>
        <th scope="row">The h1</th>
        <td>
            {% if h1Tags %}
            <i class="fas fa-check-circle text-success"></i>
            {% else %}
            <i class="fas fa-times-circle text-danger"></i>
            {% endif %}
            <input type="hidden" class="point" value="5">
        </td>
        <td>
            {% if h1Tags %}
            <div>Tim thay <b>{{ h1Tags|length }}</b> the h1 tren trang cua
                ban.</div>
            <small>{% for h1Tag in h1Tags %}<i class="fas
                fa-angle-double-right"></i> {{ h1Tag }}<br>{% endfor %}</small>
            {% else %}
            <div>Khong tim thay the h1 tren trang cua ban.</div>
            <small><i class="fas fa-angle-double-right"></i><em>
                None</em></small>
            {% endif %}
        </td>
    </tr>
</tbody>
```

Chúng tôi đã giới thiệu hai hình thức đánh giá cơ bản để hiển thị ra giao diện cho người dùng. Những tiêu chí khác tương tự, bạn có thể tham khảo tại trang mã nguồn mở của chúng tôi.

Tiếp đến, chúng tôi sẽ trình bày về cách chúng tôi chấm điểm trang web của bạn dựa trên những kết quả này. Chúng tôi sử dụng thư viện jQuery để đọc các class hiển thị icon ở cột Kết quả. Với class `fa-check-circle` xuất hiện ở cột Kết quả có nghĩa là yếu tố

đó đã vượt qua bài kiểm tra của chúng tôi. Dựa vào vị trí đó, chúng tôi tìm tiếp đến thẻ `input` nơi chứa trọng số để đánh giá tiêu chí đó được thể hiện ở thuộc tính `value`.

Đoạn code này được chúng tôi đặt trong block `script` để có thể kế thừa giao diện từ bố cục chung trong file `base.html`:

```
{% block script %}
<script>
$(document).ready(function() {
    var total = 0;
    $("#tbCheck .point").each(function() {
        total += parseInt($(this)[0].value)
    });
    var score = 0;
    $("#tbCheck .fa-check-circle").each(function() {
        score += parseInt($(this).parent().children().last()[0].value)
    });
    score = Math.round(score / total * 100);
})
</script>
{% endblock %}
```

Sau cùng, tạo thẻ `div` có `id="score"` để chúng tôi hiển thị kết quả điểm đánh giá sau khi tính toán xong. Để trực quan hơn, chúng tôi sẽ hiển thị thêm màu tương ứng với điểm nhận được.

- Xanh lá nếu điểm lớn hơn hoặc bằng 80/100.
- Vàng nếu điểm lớn hơn hoặc bằng 50/100.
- Đỏ sẽ trong trường hợp còn lại.

```
<script>
$("#score").text("Diem: " + score);
if (score >= 80) {
    $("#score").addClass("btn-success")
} else if (score >= 50) {
    $("#score").addClass("btn-warning")
} else {
    $("#score").addClass("btn-danger")
}
</script>
```

Đến đây, chúng tôi đã trình bày xong tất cả các quá trình mà chúng tôi từ lúc khởi tạo, cài đặt và xây dựng nên ứng dụng của chúng tôi. Tuy nhiên, ứng dụng vẫn chỉ đang nằm ở `localhost`, dẫn đến người dùng chưa thể truy cập vào ứng dụng trên môi trường Internet. Để triển khai ứng dụng của mình, chúng tôi sẽ trình bày ở phần tiếp ngay sau.

3.5 Triển khai ứng dụng lên Heroku

Sau khi tạo tài khoản và cài đặt xong Heroku CLI, chúng tôi sẽ tiến hành các cấu hình để triển khai ứng dụng của mình.

Đầu tiên, chúng tôi cấu hình file `Procfile` để cài đặt nền tảng sử dụng, được đặt tại thư mục gốc `check-seo`.

```
web: gunicorn src.wsgi
```

Để chỉ định phiên bản hỗ trợ của Python, chúng tôi tạo thêm file `runtime.txt` có nội dung là:

```
python-3.7.3
```

Tiếp theo chúng tôi cài đặt thêm thư viện `gunicorn` là môi trường web server cho Django được hướng dẫn bởi Heroku.

```
(venv)\>pip install gunicorn
```

Heroku cũng khuyến khích cài thêm thư viện Python do họ phát triển dành riêng cho ứng dụng Django để giúp ứng dụng hoạt động tốt trên nền tảng của họ.

```
(venv)\>pip install django_heroku
```

Sau khi cài đặt, chúng tôi cần chèn thư viện này vào file `settings.py` trong thư mục `src`:

```
# Configure Django App for Heroku.
# https://devcenter.heroku.com/articles/django-app-configuration

import django_heroku
django_heroku.settings(locals())
```

Chúng tôi cũng không quên thêm thông tin của hai gói cài đặt này vào cuối file `requirements.txt`:

```
gunicorn==19.9.0
django-heroku==0.3.1
```

Đến đây, chúng tôi đã hoàn tất cấu hình để triển khai ứng dụng lên Heroku. Việc cần làm tiếp theo là gửi code này lên máy chủ của Heroku với tên app của chúng tôi trên tài khoản là `checkseo`.

Tạo `git` trong thư mục gốc:

```
(venv)\>git init
(venv)\>heroku git:remote -a checkseo
```

Xác nhận và tải code lên Heroku:

```
(venv)\>git add .
(venv)\>git commit -m "upload"
(venv)\>git push heroku master
```

Đợi khoảng vài giây để Heroku khởi động và xây dựng ứng dụng từ mã nguồn từ bạn gửi lên. Ứng dụng của chúng tôi sẽ có đường dẫn mặc định là `https://checkseo.herokuapp.com`.

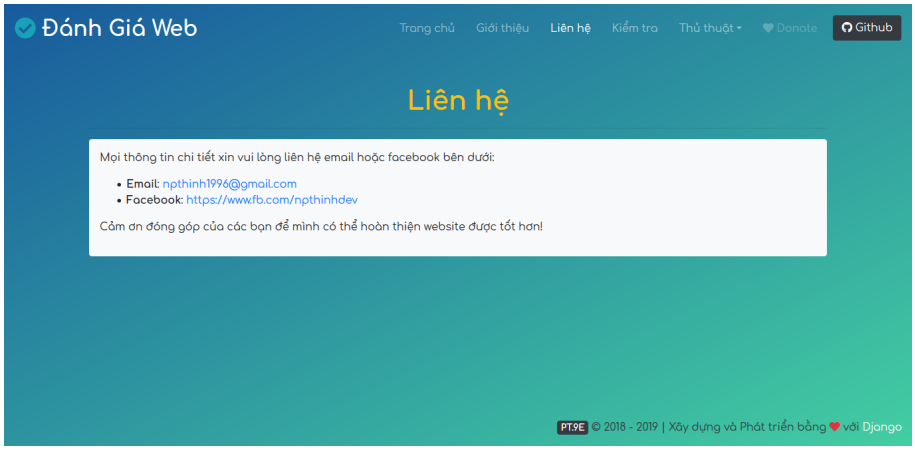
4 Sản phẩm hoàn chỉnh

Sau khi thực hiện xong các bước trong những hướng dẫn bên trên, chúng tôi đã triển khai thành công ứng dụng của mình. Tuy nhiên, để tránh sự phức tạp trong bài báo cáo này, chúng tôi đã lượt bỏ phần cấu hình các thẻ class cho thư viện Bootstrap, bạn có thể xem chi tiết chúng trong mã nguồn được chia sẻ của chúng tôi.

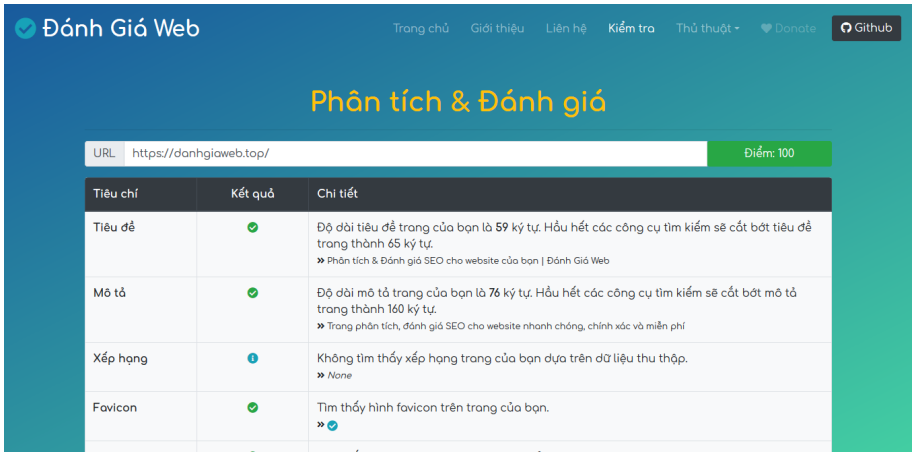
Bên dưới, chúng tôi sẽ đưa ra cho bạn thấy được khi ứng dụng của chúng tôi hoàn chỉnh về giao diện sẽ được hiển thị như thế nào.

Hình V.1: Trang chủ của ứng dụng

Hình V.2: Trang giới thiệu của ứng dụng

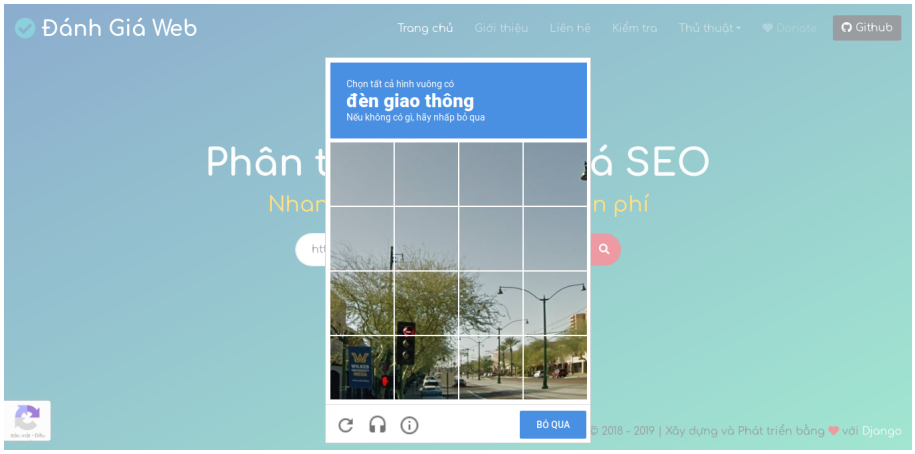


Hình V.3: Trang liên hệ của ứng dụng



Hình V.4: Trang phân tích của ứng dụng

Hình V.5: Trang thủ thuật của ứng dụng



Hình V.6: Trang kiểm tra reCaptcha của ứng dụng

Chương VI

Kết luận đánh giá

Trên đây, chúng tôi đã trình bày về dự án ***Ứng dụng Python vào lập trình website đánh giá SEO***. Website cho phép người dùng kiểm tra các thành phần cấu trúc SEO từ liên kết website người dùng nhập vào, sau đó phân tích và hiển thị kết quả. Việc sử dụng Python cùng với các thư viện hỗ trợ như `Django`, `Requests`, `Lxml`,... ứng dụng có tốc độ xử lý website khá nhanh và có thể dễ dàng mở rộng hơn sau này.

Trong quá trình phát triển, chúng tôi gặp khó khăn khi phân tích các website vì chúng có các kiểu cú pháp không giống nhau, do đó chúng tôi cần phải xây dựng thêm các hàm để lọc lại kết quả rồi mới hiển thị ra giao diện. Bên cạnh đó, ứng dụng của chúng tôi cũng gặp vấn đề với các thành phần được tạo ra bằng JavaScript, hiện tại thư viện chúng tôi đang sử dụng không thể lấy được mã nguồn từ phần nội dung này.

Hướng phát triển tiếp theo, chúng tôi sẽ thử các thư viện khác nhau nhằm khắc phục nhược điểm đối với những website chưa thể lấy được mã nguồn hoàn toàn mà vẫn đảm bảo tốc độ xử lý nhanh chóng. Xây dựng công cụ giúp người dùng có thể đánh giá mật độ từ khóa trong nội dung bài viết. Ngoài ra, chúng tôi sẽ mở rộng thêm phạm vi đánh giá SEO về các mặt như nội dung website, backlink, từ khóa, tốc độ tải trang,...

Hiện tại, ứng dụng của chúng tôi đã được xây dựng thành công trên nền tảng Heroku, người dùng có thể truy cập và thử dùng các tính năng mà chúng tôi đã phát triển:

- Website: `https://danhgiaweb.top`
- Mã nguồn: `https://github.com/pythinh/check-seo`

Tài liệu tham khảo

[1] Python tutorial. https://www.w3schools.com/python/. (truy cập 26/05/2019).

[2] Django documentation. https://docs.djangoproject.com/en/2.2. (truy cập 25/12/2018).

[3] Understanding the mvc pattern in django. https://dev.to/kolokodess/understanding-the-mvc-pattern-in-django. (truy cập 31/05/2019).

[4] Requests 2.21.0. https://pypi.org/project/requests. (truy cập 25/12/2018).

[5] lxml.html. https://pypi.org/project/lxml. (truy cập 25/12/2018).

[6] Xpath syntax. https://www.w3schools.com/xml/xpath_syntax.asp. (truy cập 31/05/2019).

[7] Introduction. https://getbootstrap.com/docs/4.1. (truy cập 25/12/2018).

[8] Icons. https://fontawesome.com/icons. (truy cập 26/12/2018).

[9] recaptcha v2. https://developers.google.com/recaptcha/docs/invisible. (truy cập 25/12/2018).

[10] Django. https://devcenter.heroku.com/categories/working-with-django. (truy cập 26/12/2018).

[11] On-site optimization. https://moz.com/beginners-guide-to-seo/on-page-seo. (truy cập 31/05/2019).

[12] Black box testing. http://softwaretestingfundamentals.com/black-box-testing. (truy cập 30/05/2019).