Priscilla Vanny Amelia (0206021810012)

PENJELASAN SOURCE CODE

Keterangan:

1. Source Code akan terlebih dulu dijelaskan secara umum, baru kemudian perbedaan crawler untuk web besar dan web kecil.
2. Web Crawler ini menggunakan module “scrapy”
3. Semua crawler diletakkan pada 1 file yang sama, sehingga setiap mau berpindah crawler, harus me-reset kernel

Penjelasan Crawler secara umum:

1. Import semua modul dan package yang diperlukan

```
import scrapy
from scrapy.crawler import CrawlerProcess
from scrapy.spiders import CrawlSpider, Rule
from scrapy.linkextractors import LinkExtractor
from scrapy.exceptions import CloseSpider
```

2. Buat kelas crawler yang merupakan turunan dari CrawlSpider:

```
class detik(CrawlSpider):
```

3. Deklarasi variable yang akan digunakan di parse:

```
class detik(CrawlSpider):
    name='detik'
    hitung = 0
    D = {0: "https://www.detik.com/"}
    E = []
    start_urls=["https://www.detik.com/"]
```

4. Atur property settings untuk Crawler:
    a. DEPTH_LIMIT: digunakan di web dengan jumlah halaman > 200, merupakan batas kedalaman crawler akan men-crawl web
    b. ROBOTSTXT_OBEY: menentukan apakah crawler akan menaati robots.txt domain atau tidak

c. CLOSESPIDER_TIMEOUT: crawler akan tertutup secara otomatis ketika waktu sudah mencapai detik yang ditentukan

```
start_urls=[ https://www.detik.com/ ]
custom_settings = {
    'DEPTH_LIMIT':'3',
    'ROBOTSTXT_OBEY':'True',
    'CLOSESPIDER_TIMEOUT':180

}
DEPTH_PRIORITY = 1
SCHEDULER_DISK_QUEUE = 'scrapy.squeues.PickleFifoDiskQueue'
SCHEDULER_MEMORY_QUEUE = 'scrapy.squeues.FifoMemoryQueue'
```

5. Menggunakan Package “Rule” dari “scrapy.spiders”, tentukan Batasan domain yang akan di crawl:

```
rules =[
  Rule(LinkExtractor(allow ="detik.com/"),
     callback = "parse_item",follow=True)
]
```

6. Buat function “parse_item” yang merupakan callback function. Berisi algoritma pengolahan response crawler: (Penjelasan callback function akan dijelaskan di segmen selanjutnya)

```
def parse_item(self, response):
    with open("detik/detikDE.txt",'w',encoding="utf-8") as f:
        f.write("\n")
          print(response.request.headers.get('Referer', None).decode('utf-8'), response.url)
        asal, tujuan = "New", "New"
        for k, v in self.D.items():
            if(v == response.request.headers.get('Referer', None).decode('utf-8')):
                asal = k
        if(asal=="New"):
            self.hitung += 1
            self.D[self.hitung] = response.request.headers.get('Referer', None).decode('utf-8')
            asal=self.hitung
        for k, v in self.D.items():
            if(v == response.url):
                tujuan = k
        if(tujuan=="New"):
            self.hitung += 1
            self.D[self.hitung] = response.url
            tujuan=self.hitung
          print("{}, {}".format(asal, tujuan))
        self.E.append("{}, {}".format(asal, tujuan))
        f.write("{}\n\n{}".format(self.D, self.E))
    with open("detik/"+str(self.hitung)+".html",'w',encoding="utf-8") as f:
        f.write(response.css("html").get())
```

7. Run Crawler:

a. Instansiasi class Crawler Process

b. Panggil crawl mana yang akan digunakan:

   process.crawl(<kelas crawler turunan dari CrawlSpider yang dibuat di no. 2>

c. Mulai proses crawling

```
  # set up a crawler
process = CrawlerProcess()
process.crawl(detik)
    # the script will block here until the crawling is finished
process.start()
```

Penjelasan "parse_item" Callback Function:

```
with open("detik/detikDE.txt",'w',encoding="utf-8") as f:
    f.write("\n")
      print(response.request.headers.get('Referer', None).decode('utf-8'), response.url)
    asal, tujuan = "New", "New"
    for k, v in self.D.items():
        if(v == response.request.headers.get('Referer', None).decode('utf-8')):
            asal = k
    if(asal=="New"):
        self.hitung += 1
        self.D[self.hitung] = response.request.headers.get('Referer', None).decode('utf-8')
        asal=self.hitung
    for k, v in self.D.items():
        if(v == response.url):
            tujuan = k
    if(tujuan=="New"):
        self.hitung += 1
        self.D[self.hitung] = response.url
        tujuan=self.hitung
      print("{}, {}".format(asal, tujuan))
    self.E.append("{}, {}".format(asal, tujuan))
    f.write("{}\n\n{}".format(self.D, self.E))
```

1. Untuk membuat file .txt penampung callback function:

   a. Deklarasi variable penampung index asal dan tujuan

   b. Menggunakan looping for, kita mencari indeks halaman referrer

   c. Apabila halaman referrer adalah baru/belum pernah tercatat di D, maka halaman tersebut akan dicatat

   d. Menggunakan looping for, kita mencari indeks halaman response

   e. Apabila halaman referer adalah baru/belum pernah tercatat di D, maka halaman tersebut akan ditambahkan di D

   f. Catat pasangan indeks halaman asal/referer dan tujuan/response di E

g. Tuliskan D dan E di file .txt yang disediakan menggunakan .write()

2. Simpan setiap html ke dalalm folder yang sudah dibuat menggunakan open() dan .write():

```
with open("harvard/"+str(self.hitung)+".html",'w',encoding="utf-8") as f:
    f.write(response.css("html").get())
```

Hal yang disimpan ke dalam file .html merupakan semua isi dalam tag <html> (didapatkan menggunakan response.css("html").get()

Perbedaan Crawler Web Besar dan Web Kecil: hanya terdapat di bagian 'custom_settings'. Web kecil tidak memakai parameter DEPTH_LIMIT dan CLOSESPIDER_TIMEOUT, karena untuk web kecil yg halamannya <200 akan di crawl semua

```
custom_settings = {
    'ROBOTSTXT_OBEY':'True'
}
```

custom_settings web kecil

```
custom_settings = {
    'DEPTH_LIMIT':'3',
    'ROBOTSTXT_OBEY':'True',
    'CLOSESPIDER_TIMEOUT':180
}
```

custom settings web besar

PENJELASAN HASIL D DAN E TIAP WEB

1. Hasil Web Indonesia Besar: https://www.detik.com/:

   Dengan pembatasan depth = 3 dan time limit 180 detik, didapatkan 930 halaman html

2. Hasil Web Indonesia Kecil: https://www.uc.ac.id/isb/:

   Didapatkan 75 halaman html

3. Hasil Web Bahasa Inggris Besar: https://pcwonderland.com/:

   Dengan pembatasan depth = 3 dan time limit 180 detik, didapatkan 766 halaman html

4. Hasil Web Bahasa Inggris Kecil: https://deepenglish.com/ :

   Didapatkan 110 halaman html