تکلیف شماره ۴

شماره گروه: ۸

اعضای گروه: آرمین افتخاری (۹۶۲۲۷۶۲۰۳۳)

محمدرضا پوررضا (۹۶۱۲۷۶۲۵۹۲)

رضا برزگر طرقبه (۹۶۲۲۷۶۲۳۸۴)

سبحان مرادیان دقیق (۹۶۲۲۷۶۲۰۶۶)

محمد سلیمان بهزاد (۹۶۲۲۷۶۲۴۵۳)

لینک "github": **HTTPS://GITHUB.COM/IRGROUP8/NEWSCRAWLER.GIT**

کلاس NewscrawlerItem

```
class NewscrawlerItem(scrapy.Item):
    # define the fields for your item here like:
    titles = scrapy.Field()
    authors = scrapy.Field()
    dates = scrapy.Field()
    page_Url = scrapy.Field()
    tags = scrapy.Field()
    content = scrapy.Field()
    cats = scrapy.Field()
```

این کلاس در فایل items.py که بصورت پیش فرض توسط خود scrapy ایجاد می شود وجود دارد که یک آیتم شخصی سازی شده متناسب با کاربرد در پروژه را مشخص می کند.

```
import scrapy
from ..items import NewscrawlerItem
```

در ابتدا لایبری scrapy را ایمپورت کرده و کلاس newsCrawlerIterm را از فایل items.py اضافه میکنیم.

```
class QuoteSpider(scrapy.Spider):
    name = 'newsCrawler'
    allowed_domains = ['truthorfiction.com']
    start_urls = ['https://www.truthorfiction.com/page/1/', ]
```

در اینجا کلاس QuoteSpider را داریم که از scrapy.spider ارث بری می کند.

در داخل آن سه متغیرname و allowed_domains و start_urls را داریم که دقیقا باید به همین نام ها تعریف شوند که name مربوط به اسم خزنده مورد استفاده است، allowed_domains دامنه مجاز برای خزش را معین می کند و start_urls آدرس شروع خزش را مشخص می کند.

عمل خزش با کمک دو تابع parse و parse_news انجام می گیرد.

تابع parse

```
page_number = 2
def parse(self, response):
    page_Urls = response.css('.entry-title
a').xpath("@href").extract()
    for i in range(len(page_Urls)):
        yield response.follow(page_Urls[i], callback=self.parse_news)

    next_page = 'https://www.truthorfiction.com/page/' +
str(self.page_number) + '/'
    if self.page_number < 12:
        self.page_number += 1
        yield response.follow(next_page, callback=self.parse)
```

در ابتدای این تابع با استفاده از response.css لیستی از تمام لینک های مربوط به محتوا را استخراج می کنیم و سپس برای هر خبر از response.follow استفاده می کنیم و با بهره بردن از آدرس مطلب و همینطور تابع parse_news عمل خزش را روی هر مطلب انجام می دهیم سپس با استفاده از متغیر page_number که در ابتدا مقدار ۲ را دارد و آدرس صفحه بعدی را برای ما تولید می کند. در ادامه با استفاده از تعیین یک آستانه، تعداد صفحات مورد نیاز برای خزش را مشخص و پیمایش می کنیم.(بالا بودن آستانه موجب کندی عمل خزش می شود).

تابع parse_news

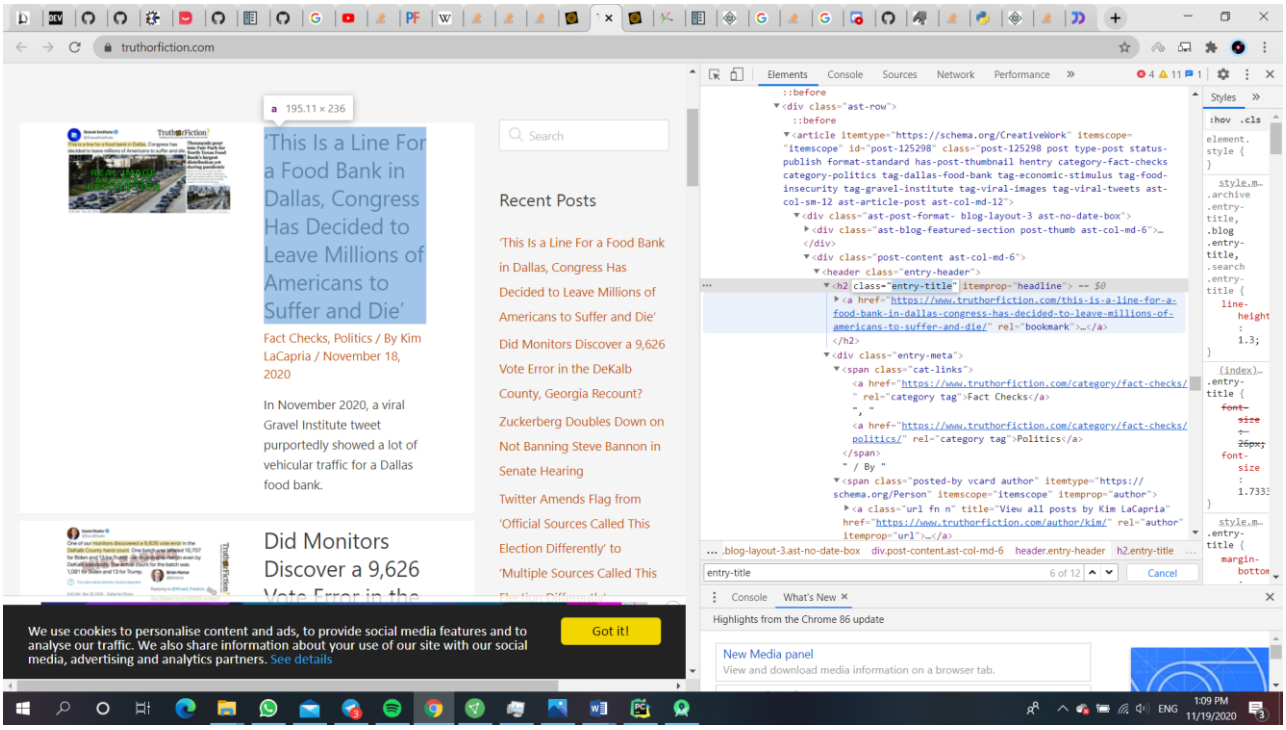
```
def parse_news(self, response):
    items = NewscrawlerItem()
    titles = response.css('.entry-title::text').extract()
    authors = response.css('.author-name::text').extract()
    dates = response.css('.published::text').extract()
    url = response.request.url
    tags = response.css(".rating-description span::text").extract()
    content = response.css(".entry-content p::text").extract()
    cats = response.css('.cat-links a::text').extract()

    # adding to item
    items['titles'] = titles
    items['authors'] = authors
    items['dates'] = dates
    items['page_Url'] = url
    items['tags'] = tags
    items['content'] = content
    items['cats'] = cats

    yield items
```

شی items از جنس کلاس newscrawlerItem را داریم که فیلد های مختلف را درون آن مقدار دهی کنیم

در این تابع تک تک فیلدهایی که برای این پروژه مورد نظر است را استخراج می کنیم. برای اینکار از insepct element در صفحه سایت مورد نظر استفاده می کنیم. به عنوان مثال برای یافتن فیلد عنوان مطلب، در فایل HTML سایت آیتم مربوط به عنوان را پیدا میکنیم که در اینجا مربوط به کلاس entry-title است. حال کافیست تا در فایل پروژه، به کمک response.css آیتم های مربوط به کلاس مورد نظر را استخراج کنیم و سپس تمامی این آیتم ها را درفیلدهای متناظر در شی items قرار میدهیم.



اجرای عمل خزش

در داخل ترمینال ، با استفاده از command زیر عمل خزش آغاز می شود.

```
scrapy crawl newsCrawler -o items.json
```

خروجی خزش

خروجی عمل خزش علاوه بر نمایش در ترمینال، در داخل فایل items.json نیز ذخیره می گردد.