Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«ВОЛГОГРАДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ВолгГТУ)

Факультет электроники и вычислительной техники

Кафедра «САПР и ПК»

Контрольная работа

по дисциплине «Компьютерная лингвистика и анализ текста»

Выполнил:

Ст. гр. САПР – 1.2

Звездин Р.Е

Проверил:

к.т.н. Коробкин Д. М.

Волгоград, 2020 г.

Содержание

[1. Постановка задачи 3](#_Toc43438289)

[2. Реализация 4](#_Toc43438290)

[3. Руководство пользователя 5](#_Toc43438291)

[Приложение А – Код парсера 6](#_Toc43438292)

[Приложение Б – Код работы с базой данных 8](#_Toc43438293)

[Приложение В – Код web-сервиса 10](#_Toc43438294)

# Постановка задачи

В рамках данной работы разрабатывается вариант для интернет-ресурса: https://bloknot-volgograd.ru/. Необходимо произвести парсинг сайта и вывести в web-интерфейсе данные.

Краулер должен считывать новостную ленту с первой страницы сайта. Периодичность повторения устанавливается пользователем. Данные заполняются в БД MongoDB.

Обязательные поля для текста новости:

* название новости;
* дата новости;
* ссылка на новость;
* текст новости;
* ссылка на видео (если есть);
* количество просмотров новости (если есть);
* количество комментариев новости (если есть);

1. Реализация

Парсер реализован на языке Python. Для парсинга сайта использована библиотека BeautifulSoup [[1](https://www.crummy.com/software/BeautifulSoup/bs4/doc/)] (Приложение А).

Для хранения данных используется MongoDB, для работы с ней использована библиотека MongoClient [[2](https://api.mongodb.com/python/current/api/pymongo/mongo_client.html)] (Приложение Б).

Пример коллекции новостей представлен на Рисунке 1.

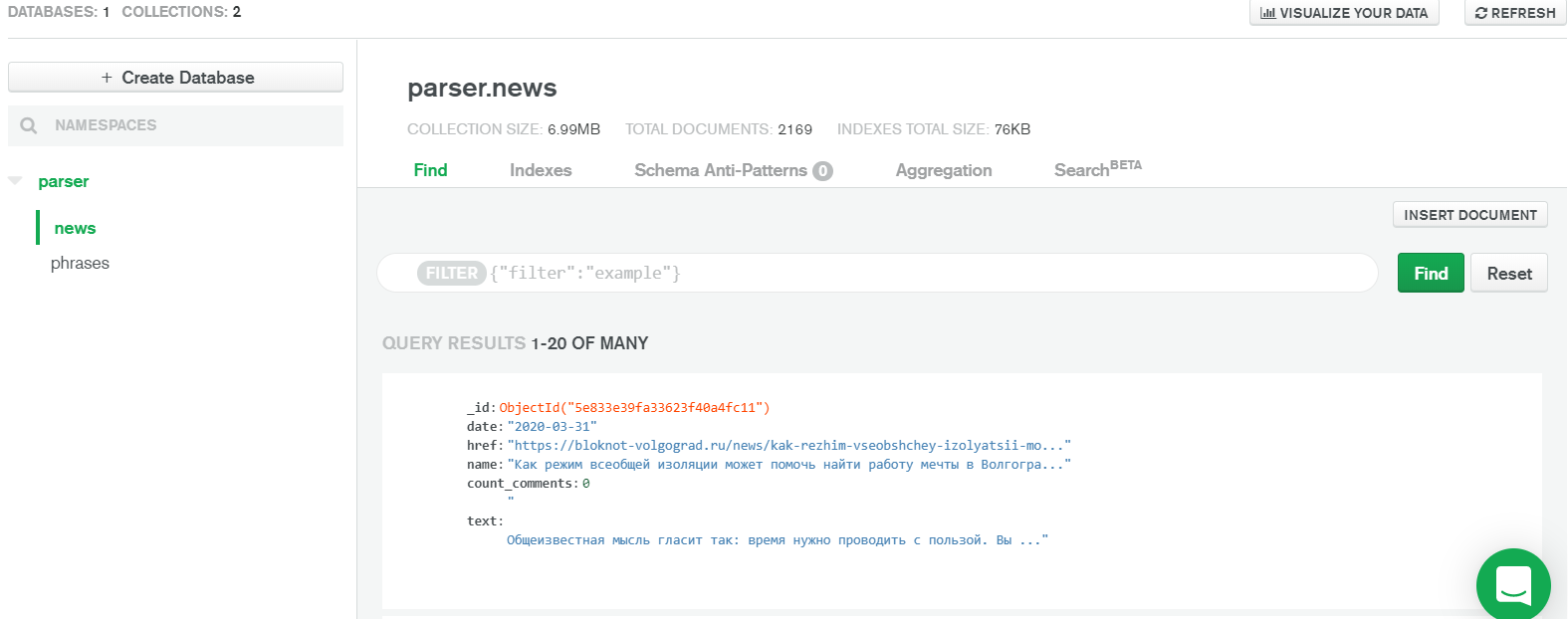


Рисунок 1 – Пример коллекции новостей

Пример коллекции фраз представлен на Рисунке 2.

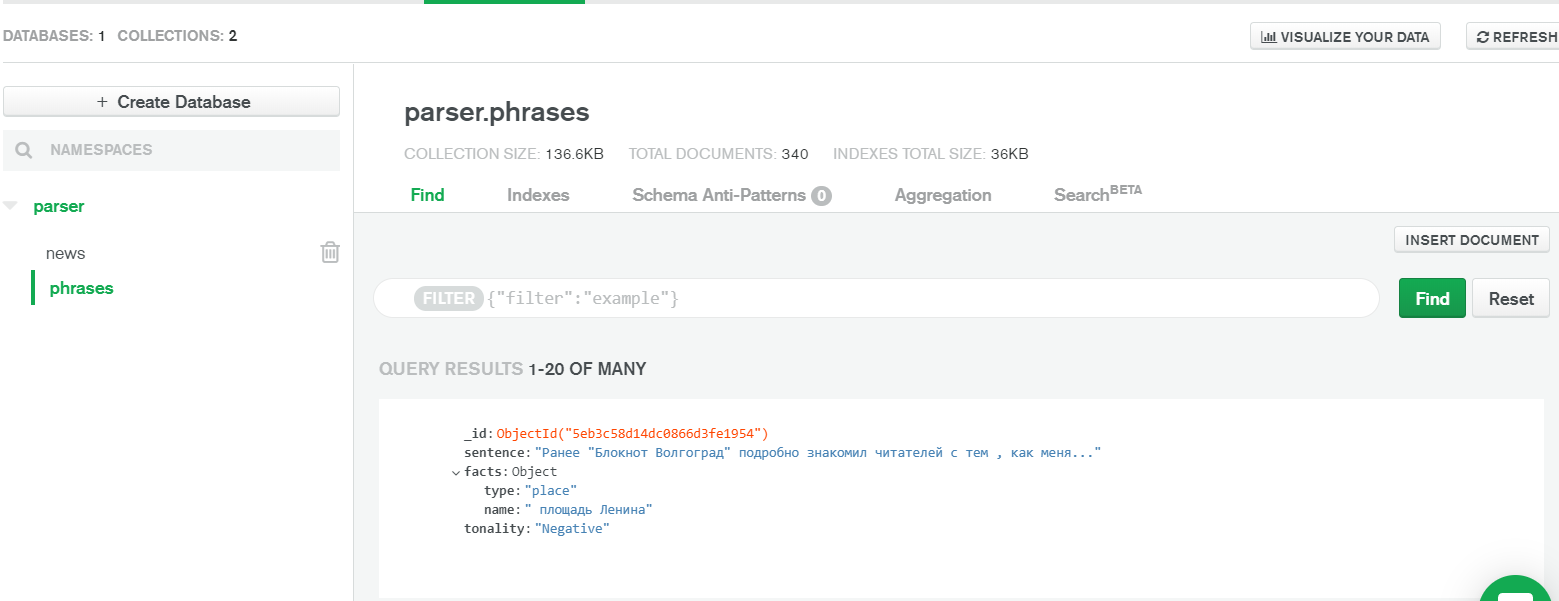


Рисунок 2 – Пример коллекции фраз

Для вывода данных в web-интерфейсе использована библиотека Flask [[3](https://flask.palletsprojects.com/en/1.1.x/)] (Приложение В).

1. Руководство пользователя

Для запуска парсера необходима версия Python 3.6 и выше, а также Библиотеки:

1. Pymongo;
2. bs4;
3. datetime;
4. requests;
5. flask.

Для запуска парсинга необходимо запустить скрипт parser.py.

Для запуска веб-приложения необходимо запустить скрипт server.py.

При запуске парсинга (parser.py), в консоли будет отображена текущая страница пагинации, которой начата обработка и запись в БД, рисунок 1.

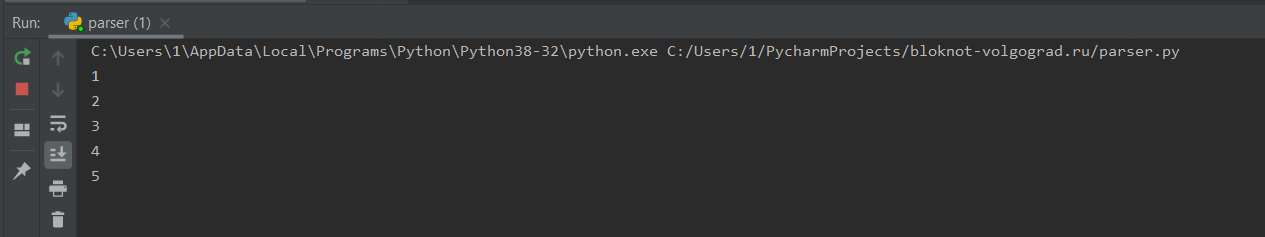


Рисунок 1 – Текущая страница пагинации

При запуске веб-приложения (server.py), в web-интерфейсе будут отображены спарсенные новости, информация о которых хранится в БД, рисунок 2. Выводится по 10 записей, для перехода на следующие 10 записей необходимо нажать на ссылку “След”.

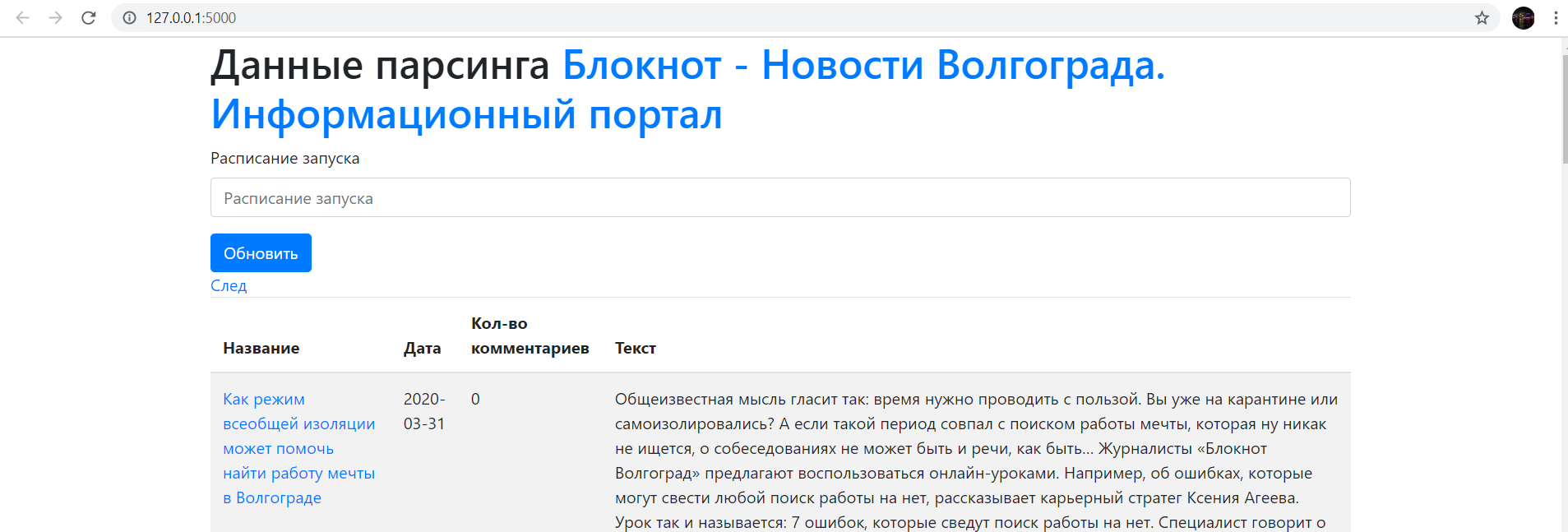


Рисунок 2 – Вывод данных в web-интерфейс

# Приложение А – Код парсера

from bs4 import BeautifulSoup

from database import Database

from datetime import datetime, timedelta

import requests

class Parser:

siteUrl = 'https://bloknot-volgograd.ru'

nextPage = ''

def doParsing(self):

database = Database()

# узнать общее кол-во страниц пагинации

first = requests.get(self.siteUrl)

soupFirst = BeautifulSoup(first.text, "html.parser")

lastPageNumber = int(soupFirst.find('div', class\_='navigation-pages').findAll('a')[-1].text) # для for page in range(1):

for page in range(lastPageNumber):

print(page + 1)

# главная или нет

if self.nextPage == '':

newsListUrl = self.siteUrl

else:

newsListUrl = self.siteUrl + self.nextPage

newsListUrl = requests.get(newsListUrl)

# страница

soupNewsList = BeautifulSoup(newsListUrl.text, "html.parser")

# получить адрес следующей страницы пагинации

self.nextPage = soupNewsList.find(id='navigation\_1\_next\_page').get('href')

# собрать новости на текущей странице

newsListArr = soupNewsList.find('ul', class\_='bigline').findAll('a', class\_='sys') # для for i in range

# обработать данные о каждой новости

for i in range(len(newsListArr)):

# ссылка на новость

currentNewsShortUrl = newsListArr[i].get('href')

currentNewsFullUrl = self.siteUrl + newsListArr[i].get('href')

# поля новости

newsData = requests.get(currentNewsFullUrl)

soupNews = BeautifulSoup(newsData.text, "html.parser")

# почистить от js

for script in soupNews(["script", "style"]):

script.decompose()

# поля новости

name = soupNews.find('h1').text

date = soupNewsList.find('a', href=currentNewsShortUrl + '#comments').find\_parent('span').next\_element.next\_element

# обработаем дату

date = str(self.converteDate(date))

href = currentNewsFullUrl

text = soupNews.find('div', class\_='news-text').text

count\_comments = int(soupNewsList.find('a', href=currentNewsShortUrl + '#comments').text)

# запись в БД

database.addNews({

"name" : name,

"date" : date,

"href" : href,

"text" : text,

"count\_comments" : count\_comments

})

def converteDate(self, date):

if "сегодня" in date:

newDate = datetime.now().date()

elif "вчера" in date:

newDate = datetime.strftime(datetime.now() - timedelta(1), '%Y-%m-%d')

else:

date = ''.join(date.split())

newDate = datetime.strptime(date, '%d.%m.%Y')

newDate = datetime.strftime(newDate, '%Y-%m-%d')

return newDate

parser = Parser()

parser.doParsing()

# Приложение Б – Код работы с базой данных

from pymongo import MongoClient

class Database:

client = MongoClient("")

#client = MongoClient('mongodb://localhost:27017/')

db = client["parser"]

collectionNews = db["news"]

collectionPhrases = db["phrases"]

# Добавить/обновить запись в БД (новости)

def addNews(self, data):

self.collectionNews.find\_one\_and\_update({

"name": data['name'],

"date": data['date'],

"href": data['href']

},

{

"$set": data

},

upsert=True

)

# Получить все записи из БД - не рекомендуется использовать при кол-ве > 1000 (новости)

def getAllNews(self):

data = self.collectionNews.find({})

return list(data)

# Пагинация (новости)

def getPaginationNews(self, pageNum, pageSize):

skips = int(pageSize) \* (int(pageNum) - 1)

data = self.collectionNews.find({}).skip(skips).limit(pageSize)

return list(data)

# Добавить запись в БД (фразы - предложения с фактами)

def addPhrase(self, data):

self.collectionPhrases.insert\_one({

"sentence": data['sentence'],

"facts": data['facts'], # [{ 'type': '' 'name': '' }, { 'type': '' 'name': '' }],

}

)

# Получить все записи из БД - не рекомендуется использовать при кол-ве > 1000 (фразы - предложения с фактами)

def getAllPhrases(self):

data = self.collectionPhrases.find({})

return list(data)

# Пагинация (фразы - предложения с фактами)

def getPaginationPhrases(self, pageNum, pageSize):

skips = int(pageSize) \* (int(pageNum) - 1)

data = self.collectionPhrases.find({}).skip(skips).limit(pageSize)

return list(data)

# Приложение В – Код web-сервиса

from flask import Flask, render\_template, redirect, url\_for, request

from database import Database

database = Database()

app = Flask(\_\_name\_\_)

@app.route('/', methods=['GET'])

def index():

page, next\_page = get\_pages\_numbers(request)

records = database.getPaginationNews(page, 10)

return render\_template('index.html', records=records, nextPage=next\_page)

@app.route('/change\_shedule', methods=['POST'])

def change\_shedule():

return redirect(url\_for('index'))

@app.route('/tonality', methods=['GET'])

def get\_tonality():

page, next\_page = get\_pages\_numbers(request)

records = database.getPaginationPhrases(page, 10)

return render\_template(

'tonality.html', records=records, next\_page=next\_page)

def get\_pages\_numbers(request):

page = request.args.get('page', 1)

next\_page = int(page) + 1

return page, next\_page

if \_\_name\_\_ == "\_\_main\_\_":

app.run()