Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

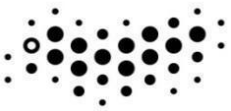
Лабораторная работа № 5-6

на тему
“**РАЗРАБОТКА ИНТЕРФЕЙСОВ ДЛЯ ВЫПОЛНЕНИЯ CRUD-ОПЕРАЦИЙ НАД БАЗОЙ ДАННЫХ СРЕДСТВАМИ PHP**”

в рамках проекта “Персонализация новостной ленты Hacker News”

по дисциплине «**Базы данных**»

Автор: Исхакова Эмина
Факультет: ИКТ
Группа: К3242



УНИВЕРСИТЕТ ИТМО

Санкт-Петербург, 2021

1. Цель работы: овладеть практическими навыками разработки форм для вставки, выборки и редактирования данных.

2. Программное обеспечение: SQLite, окружение PyCharm

3. Ход работы:

   В этой работе задачей является написание простого персонализированного новостного агрегатора. Для выполнения требуется собирать и размечать новости из социально-новостного сайта Hacker News.

- **Сбор данных:**

```
def extract_next_page(parser):
    """ Extract next page URL """
    more = parser.find('a', class_='morelink')
    return more['href']
```

```
def get_news(url, n_pages=1):
    """ Collect news from a given web page """
    news = []
    while n_pages:
        print("Collecting data from page: {}".format(url))
        response = requests.get(url)
        soup = BeautifulSoup(response.text, "html.parser")
        news_list = extract_news(soup)
        print(news_list)
        next_page = extract_next_page(soup)
        url = 'https://news.ycombinator.com/' + next_page
        news.extend(news_list)
        n_pages -= 1
    return news
```

Для работы использовался модуль языка Python requests, который позволяет отправлять запросы по HTTP-протоколу. В запросе типа GET информация, передаваемая серверу, расположена в ссылке, которую в случае использования веб-браузера вы можете видеть в адресной строке.

Для извлечения данных с веб-страниц существует множество разных модулей. Я пользовалась модулем Beautiful Soup 4.

Для использования Beautiful Soup нужно передать текст веб-страницы (в виде одной строки) функции BeautifulSoup. Чтобы он не "ругался", также следует указывать название парсера (той программы, которая осуществляет обработку HTML). С целью совместимости я использую html.parser ведь он входит в пакет Python и не требует дополнительной установки.

```
def extract_news(parser):
    """ Extract news from a given web page """
    news_list = []
    titles = []
    authors = []
    urls = []
    coms = []
    scores = []
    mya = parser.find_all('a', class_='storylink')
    for a in mya:
        titles.append(a.contents[0])
    author = parser.find_all('a', class_='hnuser')
    for a in author:
        authors.append(a.contents[0])
    myurl = parser.find_all('a', class_='storylink')
    for a in myurl:
        urls.append(a['href'])
    comments = parser.find_all('td', class_='subtext')
    for td in comments:
        coms.append(td.find_all('a')[-1].contents[0])
    score = parser.find_all('span', class_='score')
    for span in score:
        scores.append(span.contents[0])
    for i in range(len(titles)):
        new = {}
        new['title'] = titles[i]
        new['author'] = authors[i]
        new['urls'] = urls[i]
        new['comments'] = coms[i]
        new['score'] = scores[i]
        news_list.append(new)
    return news_list
```

Для обращения и сбора внутренних таблиц с html-страницы, а именно самих новостей, названия, автора, ссылки, оценки, количества комментариев была написана функция extract news.

В целом, на шаге сбора данных было написано 3 функции extract_news() и extract_next_page(), необходимые для корректной работы функции get_news(), которая в качестве аргументов принимает url и n_pages (число страниц, с которых необходимо собрать новости), а возвращает список словарей, где каждый словарь представляет собой запись об одной новости.

- **Сохранение данных в SQLite**
  Собираемые данные нужно где-то хранить. Для хранения используется SQLite - компактная встраиваемая реляционная база данных. В стандартной библиотеке языка Python есть модуль sqlite3, который предоставляет интерфейс для работы с SQLite. Этот модуль требует знания языка SQL, поэтому мы воспользуемся другой технологией, которая называется ORM (ORM (англ. object-relational mapping, рус. объектно-реляционное отображение) — технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая "виртуальную объектную базу данных").

  SQLAlchemy — библиотека на языке Python для работы с реляционными СУБД с применением технологии ORM. Используется для синхронизации объектов Python и записей реляционной базы данных. SQLAlchemy позволяет описывать структуры баз данных и способы взаимодействия с ними на языке Python без использования SQL.

```python
from sqlalchemy import Column, String, Integer
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker
from scruputils import *


Base = declarative_base()
engine = create_engine("sqlite:///news.db")
session = sessionmaker(bind=engine)


class News(Base):
    __tablename__ = "news"
    id = Column(Integer, primary_key=True)
    title = Column(String)
    author = Column(String)
    url = Column(String)
    comments = Column(Integer)
    points = Column(Integer)
    label = Column(String)

Base.metadata.create_all(bind=engine)
```

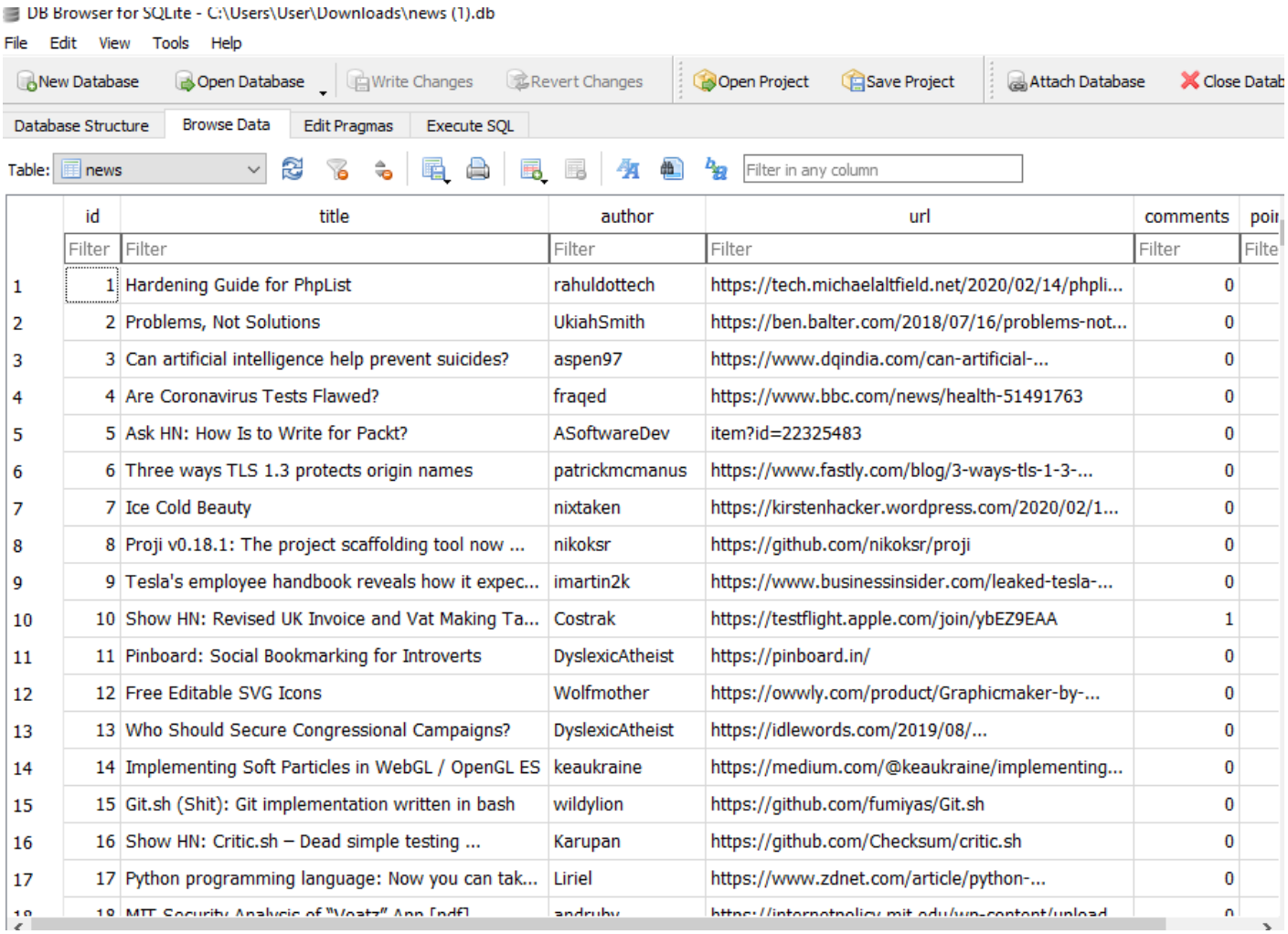
Каждая таблица описывается классом, который должен наследоваться от базового класса, создаваемого функцией sqlalchemy.ext.declarative.declarative_base(). В рассматриваемом проекте будет только один **класс News с следующими атрибутами**: заголовок, автор, ссылка, количество комментариев и число лайков. Поле label будет заполнено после разметки данных.

Создания объекта и сохранения его в БД:

```python
'''all = get_news('https://news.ycombinator.com/newest', 34)
s = session()
for new in all:
    news = News(title = new['title'],
                author = new['author'],
                url = new['urls'],
                comments= new['comments'],
                points = new['score'])
    s.add(news)
    s.commit()'''
```

Идентификатор объекта id содержит значение None до тех пор, пока не сделается коммит этого объекта в БД с помощью метода commit().

Мною было создано 1000 записей с новостного сайта в БД (на каждой странице по 34 новости). Содержимое файла news.db:



	id	title	author	url	comments	poi
	Filter	Filter	Filter	Filter	Filter	Filte
1	1	Hardening Guide for PhpList	rahuldottech	https://tech.michaelaltfield.net/2020/02/14/phpli...	0	
2	2	Problems, Not Solutions	UkiahSmith	https://ben.balter.com/2018/07/16/problems-not...	0	
3	3	Can artificial intelligence help prevent suicides?	aspen97	https://www.dqindia.com/can-artificial-...	0	
4	4	Are Coronavirus Tests Flawed?	fraqed	https://www.bbc.com/news/health-51491763	0	
5	5	Ask HN: How Is to Write for Packt?	ASoftwareDev	item?id=22325483	0	
6	6	Three ways TLS 1.3 protects origin names	patrickmcmanus	https://www.fastly.com/blog/3-ways-tls-1-3-...	0	
7	7	Ice Cold Beauty	nixtaken	https://kirstenhacker.wordpress.com/2020/02/1...	0	
8	8	Proji v0.18.1: The project scaffolding tool now ...	nikoksr	https://github.com/nikoksr/proji	0	
9	9	Tesla's employee handbook reveals how it expec...	imartin2k	https://www.businessinsider.com/leaked-tesla-...	0	
10	10	Show HN: Revised UK Invoice and Vat Making Ta...	Costrak	https://testflight.apple.com/join/ybEZ9EAA	1	
11	11	Pinboard: Social Bookmarking for Introverts	DyslexicAtheist	https://pinboard.in/	0	
12	12	Free Editable SVG Icons	Wolfmother	https://owwly.com/product/Graphicmaker-by-...	0	
13	13	Who Should Secure Congressional Campaigns?	DyslexicAtheist	https://idlewords.com/2019/08/...	0	
14	14	Implementing Soft Particles in WebGL / OpenGL ES	keaukraine	https://medium.com/@keaukraine/implementing...	0	
15	15	Git.sh (Shit): Git implementation written in bash	wildylion	https://github.com/fumiyas/Git.sh	0	
16	16	Show HN: Critic.sh – Dead simple testing ...	Karupan	https://github.com/Checksum/critic.sh	0	
17	17	Python programming language: Now you can tak...	Liriel	https://www.zdnet.com/article/python-...	0	

- **Разметка данных**
  Для разметки данных создается простую HTML-страницу, на которой будет выводиться список неразмеченных новостей, а рядом с каждой новостью будет несколько кнопок со следующими метками:
  «Интересно» - эта новость вам показалась интересной, и вы ее прочитали;
  «Не интересно» - эта новость вас не интересует;
  «Возможно прочитаю» - вы сомневаетесь - интересна вам эта новость или нет.

По нажатию на кнопку происходит добавление метки в БД к соответствующей новости и удаления новости из списка неразмеченных новостей, так как я ее уже разметила.

Для создания такой веб-страницы воспользовалась простым и популярным веб-фреймворком bottle. В bottle реализован механизм шаблонов, предназначенный для генерации веб-страниц. Для начала работы с шаблонизатором достаточно воспользоваться функцией template, которая в качестве первого аргумента принимает имя файла, содержащего текст шаблона (в нашем случае это news_template.tpl).

Затем следует список необязательных именованных аргументов, которые нужно передать шаблонизатору

```
<!DOCTYPE html>
<html>
    <head>
        <link rel="stylesheet" href="//cdnjs.cloudflare.com/ajax/libs/semantic-ui/2.2.12/semantic.min.css"></link>
        <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
        <script src="https://cdnjs.cloudflare.com/ajax/libs/semantic-ui/2.2.12/semantic.min.js"></script>
    </head>
    <body>
        <div class="ui container" style="padding-top: 10px;">
        <table class="ui celled table">
            <thead>
                <th>Title</th>
                <th>Author</th>
                <th>#Likes</th>
                <th>#Comments</th>
                <th colspan="3">Label</th>
            </thead>
            <tbody>
                %for row in rows:
                <tr>
                    <td><a href="{{ row.url }}">{{ row.title }}</a></td>
                    <td>{{ row.author }}</td>
                    <td>{{ row.points }}</td>
                    <td>{{ row.comments }}</td>
                    <td class="positive"><a href="/add_label/?label=good&id={{ row.id }}">Интересно</a></td>
                    <td class="active"><a href="/add_label/?label=maybe&id={{ row.id }}">Возможно</a></td>
                    <td class="negative"><a href="/add_label/?label=never&id={{ row.id }}">Не интересно</a></td>
                </tr>
                %end
            </tbody>
            <tfoot class="full-width">
```

```
from bottle import (
    route, run, template, request, redirect
)

from scruputils import get_news
from db import News, session
from bayas import NaiveBayesClassifier
import string


@route("/news")
def news_list():
    s = session()
    rows = s.query(News).filter(News.label == None).all()
    return template('news_template', rows=rows)
```

Функция, которая будет отвечать за маршрут news и выводить список неразмеченных новостей. Функция run запускает веб-сервер по адресу localhost:8080. route это функция-декоратор, которая отвечает за маршрутизацию и связывает адрес ресурса (веб-страницы) с функцией, которая должна быть вызвана при обращении к этому ресурсу. Запрос к БД: s.query(News).filter(News.label == None).all():

1. Мы обращаемся к таблице News с помощью query(News)
2. Фильтруем записи. Нам нужны только те, которые не имеют метки: filter(News.label == None)
3. Все полученные через all() записи передаем в шаблон

В шаблоне мы формируем таблицу со списком неразмеченных новостей.

Title	Author	#likes	#comments	Label		
Does Snapchat Need to Go Beyond Young Users to Succeed?	happy-go-lucky	1	0	Интересно	Возможно	Не интересно
Foodies Know: Boulder Has Become a Hub for New Producers	robertgk	1	0	Интересно	Возможно	Не интересно
Stop Expensive Bugs with These Proven Methods	dpweb	1	0	Интересно	Возможно	Не интересно
Coming Soon – Open Source Guide	citrusui	1	0	Интересно	Возможно	Не интересно
UBeam's wireless charging demo	perseusprime11	1	0	Интересно	Возможно	Не интересно
An Anonymous group just took down a fifth of the dark web	Spydar007	2	0	Интересно	Возможно	Не интересно
Trump's Tax Plan Could Tack $10 Trillion onto America's Debt	warsaw	1	0	Интересно	Возможно	Не интересно
LED Minecraft Server Status with Raspberry Pi	matt2000	2	0	Интересно	Возможно	Не интересно
SSH Chat	turrini	5	0	Интересно	Возможно	Не интересно
Best way to handle dependencies between components of a PHP framework?	webNeat	1	0	Интересно	Возможно	Не интересно

У нас есть три ссылки «Интересно», «Не интересно» и «Возможно». Переход по каждой из которых обрабатывается функцией, связанной с маршрутом add_label. Также происходит передача двух параметров label (нашей метки, со значениями good, maybe, never) и идентификатора новости id (каждая новость имеет уникальный числовой идентификатор, который она получает при добавлении в БД).

Задачи функции add_label

Получить значения параметров label и id из GET-запроса
Получить запись из БД с соответствующим id (такая запись только одна!)
Изменить значение метки записи на значение label
Сохранить результат в БД

```
@route("/add_label/")
def add_label():
    id = request.query['id']
    label = request.query['label']
    s = session()
    line = s.query(News).filter(News.id == id).first()
    line.label = label
    s.commit()
    redirect("/news")
```

Задачи функции update_news, которая добавляет свежие новости в БД:
Получить данные с новостного сайта
Проверить, каких новостей еще нет в БД. Будем считать,что каждая новость может быть уникально идентифицирован по совокупности двух значений: заголовка и автора
Сохранить в БД те новости, которых там нет

```
@route("/update")
def update_news():
    abc = get_news('https://news.ycombinator.com/newest')
    s = session()
    for g in abc:
        title = g['title']
        author = g['author']
        rows = s.query(News).filter(News.title == title).filter(News.author == author).first()
        if rows is None:
            new = News(title = g['title'],author = g['author'], url = g['urls'], comments = g['comments'], points = g['score'])
            s.add(new)
            s.commit()
    redirect("/news")
```

- **Классификация данных**

Необходимо написать простой классификатор, который бы выводил неразмеченные новости в следующем порядке: сначала идут интересные для нас новости, затем те, которые мы бы возможно прочитали, и в конце - неинтересные новости.

Итак, у нас есть корпус, состоящий из размеченных и неразмеченных документов (новостей). Возникает два вопроса: «Каким образом каждой свежей новости присвоить одну из меток (классов)?» и «Как оценить нашу классификацию?».
Для более качественной классификации буду использовать **наивный байесовский классификатор**.

Написанный класс можно посмотреть [тут.]

Задачи обработчика запроса для вывода ранжированной по label таблицы новостей:
Получить список неразмеченных новостей из БД
Получить прогнозы для каждой новости
Вывести ранжированную таблицу с новостями

```
@route("/classify")
def classify_news():
    bs = NaiveBayesClassifier(1)
    s = session()
    nolable = s.query(News).filter(News.label == None).all()
    X = processing(nolable)
    X_train = s.query(News).filter(News.label != None).all()
    y = []
    for item in X_train:
        y.append(item.label)
    X_train = processing(X_train)
    bs.fit(X_train, y)
    predictions = bs.predict(X)
    counter = 0
    for item in nolable:
        item.label = predictions[counter]
        counter += 1
    nolable.sort(key=lambda x: x.label)
    nolable.reverse()
    return template('news_template', rows=nolable)
    pass
```

4. Вывод:

Был создан простой персонализированный новостной агрегатора, выдающий новости, согласно ражнжировке. Ранжирование происходит благодаря собранным ранее новостями с того же сайте в БД, а также добавления к записи новости соответствующего лейбла согласно моим вкусам, после чего мешок слов из новостей был классифицирован с помощью наивного байесовского классификатора.