

Министерство образования и науки РФ
Федеральное государственное образовательное учреждение
высшего профессионального образования
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)
Кафедра комплексной информационной безопасности электронно-
вычислительных систем (КИБЭВС)

Место прохождения практики:
ТУСУР, каф. КИБЭВС, г.Томск

ОТЧЁТ
по технологической практике

Выполнили:
студентка гр. 722
_____ Мейта М.В.
студент гр. 742
_____ Шиповской В.В.
« » _____ 2015 г.

Руководитель практики от
предприятия:
к.т.н. каф. КИБЭВС
_____ Романов А.С.
« » _____ 2015 г.

Томск 2015

АННОТАЦИЯ

к технологической практике
студентов 3 курса
кафедры КИБЭВС, ТУСУРа,
г.Томска
Мейта М.В., Шиповской В.В.
2015, 27 стр.
14 ил., библиогр. список – 12
наим.

Целью технологической практики было начало разработки системы мониторинга настроения людей в социальных сетях для предотвращения массовых беспорядков и экстремизма, то есть такого программного комплекса, который позволит производить семантический анализ текстовых сообщений на предмет негативной эмоциональной окраски. Подобного рода система может быть использована в различных целях, как научно-исследовательских, так и коммерческих.

В рамках данной практической работы была разработана программа для извлечения коротких текстовых сообщений («tweets») по определенному ключевому слову («hashtag») в социальной сети «Twitter», последующего сохранения извлеченной информации в базу данных, а также веб-интерфейс для отображения полученных результатов на основе программной платформы Django версии 1.8.0.

Планируется дальнейшая разработка программного комплекса для семантического анализа высказываний в социальных сетях, а также написание научно-исследовательской статьи в рамках X международной научно-практической конференции «Электронные средства и системы управления».

СОДЕРЖАНИЕ

Введение.....	4
1 Кафедра КИБЭВС	5
2 Разработка системы мониторинга настроения людей в социальных сетях с целью предотвращения массовых беспорядков и экстремизма.....	6
2.1 Используемые программные средства, обоснование выбора и их описание.....	9
2.1.1 Язык программирования Python.....	10
2.1.2 Программная платформа Django	12
2.1.3 Система контроля версий Git и веб-сервис для создания удаленного репозитория GitHub	14
3 Особенности работы Django	15
4 Проектирование базы данных. Создание моделей в Django	19
5 Twitter API. Программа для поиска сообщений по ключевому слову	21
Заключение	25
Список использованных источников	26

Введение

В период 29 июня 2015 года по 12 июля 2015 года нами была пройдена технологическая практика на кафедре КИБЭВС ТУСУРа. В ходе прохождения практики была проведена следующая работа:

1) изучение предметной области проблемы, существующих разработок в области семантического анализа, инструментов и методов для решения поставленной задачи;

2) написание программы для поиска текстовых сообщений («твитов») на интерпретируемом языке высокого уровня Python версии 2.7.9;

3) проектирование базы данных для хранения полученной выборки сообщений с сопутствующей ей информации об авторах, дате публикации и др.;

4) изучение программной платформы Django, ее установка, а также настройка и создание шаблона проекта для разработки веб-интерфейса системы;

5) разработка веб-интерфейса системы для отображения результатов работы программы, а именно содержимого спроектированной базы данных.

1 Кафедра КИБЭВС

Местом прохождения практики была выбрана кафедра ТУСУРа – КИБЭВС (Кафедра комплексной информационной безопасности электронно-вычислительных систем).

Кафедра организована в ТУСУР в 1971 году как кафедра «Конструирования и производства электронно-вычислительной аппаратуры» (КиП ЭВА) вскоре переименованной в кафедру «Конструирования электронно-вычислительной аппаратуры» (КЭВА).

21 сентября 1999 г. в связи с открытием новой актуальной специальности 090105 – «Комплексное обеспечение информационной безопасности автоматизированных систем» кафедра КЭВА была переименована в кафедру «Комплексной информационной безопасности электронно-вычислительных систем» (КИБЭВС).

Заведующим кафедрой КИБЭВС на сегодняшний день является ректор ТУСУРа,

Александр Александрович Шелупанов, лауреат премии Правительства Российской Федерации, действительный член Международной Академии наук высшей школы РФ, действительный член Международной Академии информации, Почетный работник высшего профессионального образования РФ, заместитель Председателя Сибирского регионального отделения учебно-методического объединения вузов России по образованию в области информационной безопасности, профессор, доктор технических наук.

С 2008 г. кафедра КИБЭВС входит в состав Института «Системной интеграции и безопасности».

На базе кафедры КИБЭВС ТУСУР в 2002 году организовано «Сибирское региональное отделение учебно-методического объединения Вузов России по образованию в области информационной безопасности [1].

2 Разработка системы мониторинга настроения людей в социальных сетях с целью предотвращения массовых беспорядков и экстремизма

В современном мире набрали огромную популярность такие средства массовой коммуникации, как социальные сети. Подобные информационные площадки являются мощным инструментом маркетинговых, социальных и иных исследований. Также социальные сети становятся источниками распространения экстремизма, социальной, расовой, национальной и религиозной ненависти и вражды. В связи с этим возникает необходимость отслеживания и возможного урегулирования настроений людей в социальных сетях с целью предотвращения массовых беспорядков, нарушений конституционных прав граждан, защиты национальных интересов страны. Данная задача может быть осуществлена при помощи программного комплекса, включающего в себя инструменты для анализа тональности текста сообщений и их классификации по эмоциональной окраске.

Классификация высказываний по эмоциональной окраске может быть проведена различными способами. Условно можно разделить все высказывания на три подмножества: позитивно, негативно или нейтрально окрашенные. Далее встает проблема обучения автоматизированной системы семантическому анализу тональности произвольного высказывания из конечного множества сообщений.

Существует множество разработок в области семантического анализа и создания нейронных сетей, способных решать задачи компьютерной лингвистики. Среди них – разработанная учеными из Стэнфорда нейросеть, способная, по заявлению разработчиков, определять тональность англоязычного текста с точностью 85% [2]. Также есть различные отечественные разработки – как коммерческие, так и доступные для исследования и научной работы. Однако по-прежнему данная тема остается актуальной и интерес к ней активно возрастает.

В качестве первого шага в разработке данной системы была поставлена задача создать программу для извлечения коротких текстовых сообщений («твитов»), публикуемых в социальной сети «Twitter» по ключевому слову («хэштегу») с последующим сохранением в базу данных как самих сообщений, так и сопутствующей им информации (об авторе, дате публикации, локации пользователя и др.). Также разработать веб-интерфейс для отображения полученных данных и продолжить исследование данной предметной области для написания научно-исследовательской статьи в рамках X Международной научно-практической конференции «Электронные средства и системы управления» [3] и дальнейшей разработки системы.

Архитектура системы в целом для дальнейших разработок примет вид, представленный на рисунке 2.1.

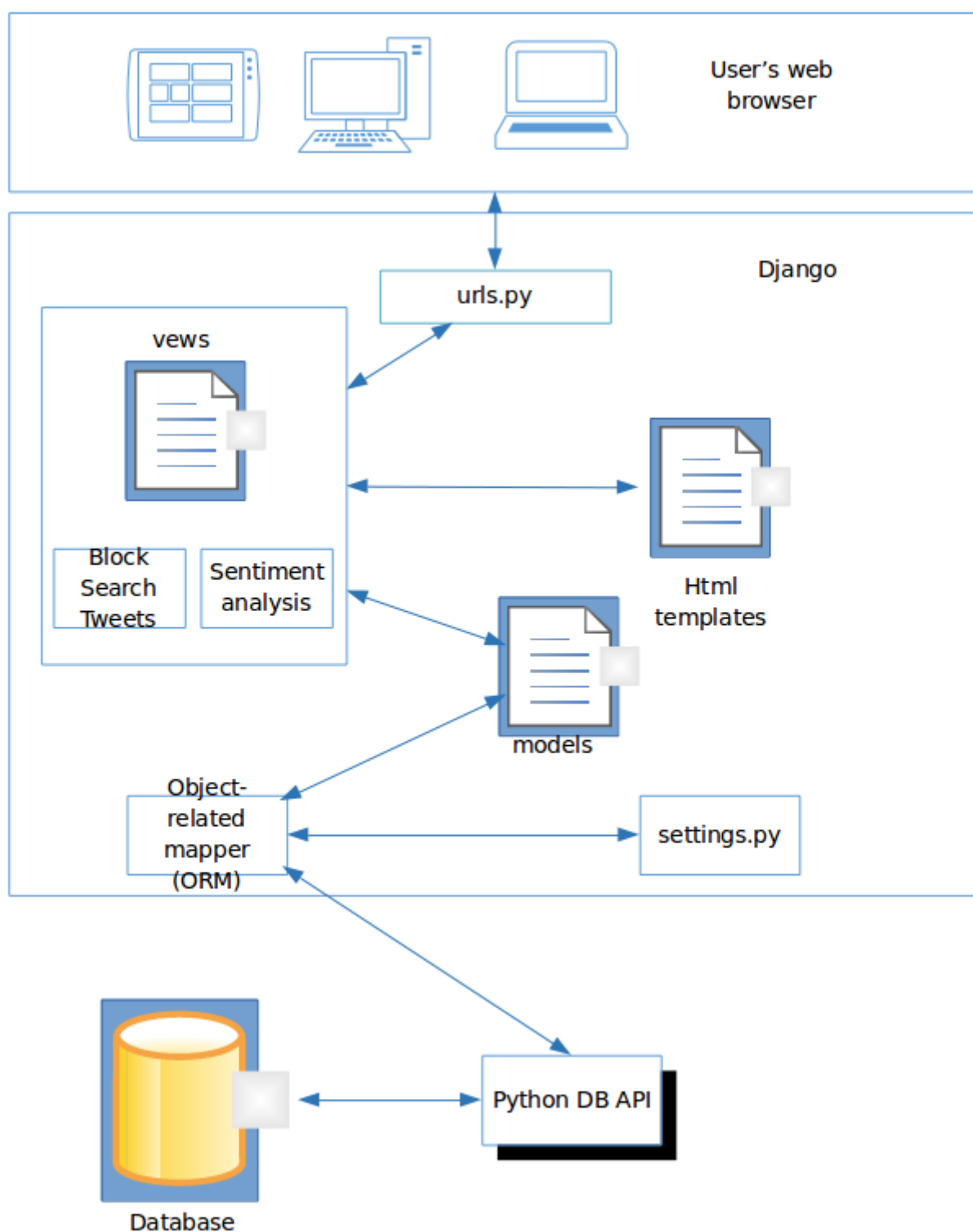


Рисунок 2.1 – Примерная архитектура системы мониторинга настроения людей в социальных сетях

2.1 Используемые программные средства, обоснование выбора и их описание

В качестве инструментов разработки были выбраны:

- 1) интерпретируемый высокоуровневый язык программирования Python версии 2.7.9;
- 2) программная платформа Django версии 1.8.0;
- 3) система контроля версий Git и веб-сервис для создания удаленного репозитория GitHub;
- 4) кроссплатформенный текстовый редактор Sublime Text 2 для написания кода.

2.1.1 Язык программирования Python

Python — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций.

Python поддерживает несколько парадигм программирования, в том числе структурное, объектно-ориентированное, функциональное и аспектно-ориентированное. Основные архитектурные черты — динамическая типизация (переменная связывается с типом во время присвоения значения, а не во время объявления переменной), автоматическое управление памятью, полная интроспекция (возможность определять тип и структуру объекта во время выполнения программы), механизм обработки исключений, поддержка многопоточных вычислений и удобные высокоуровневые структуры данных. Код в Python организовывается в функции и классы, которые могут объединяться в модули (они в свою очередь могут быть объединены в пакеты) [4].

Можно выделить следующие преимущества данного языка программирования:

- низкий порог вхождения;
- объектно-ориентированный;
- простота и удобство интуитивно понятного синтаксиса;
- обширный набор различных библиотек;
- метапрограммирование и другие возможности в ООП;
- качественная документация;
- постоянно растущее комьюнити разработчиков;
- высокая скорость работы.

Кроме всего вышеперечисленного, Twitter имеет множество готовых библиотек с необходимыми API-функциями для различных языков программирования, представленными на официальном сайте в разделе для разработчиков [5]. В том числе среди них представлены и библиотеки для Python – например, tweepy [6], которая использовалась в ходе нашей работы.

2.1.2 Программная платформа Django

Django — свободный фреймворк (программная платформа) для веб-приложений на языке Python, использующий шаблон проектирования MVC. Проект поддерживается организацией Django Software Foundation.

Сайт на Django строится из одного или нескольких приложений, которые рекомендуется делать отчуждаемыми и подключаемыми. Это одно из существенных архитектурных отличий этого фреймворка от некоторых других (например, Ruby on Rails). Один из основных принципов фреймворка — DRY (англ. Don't repeat yourself)

Также, в отличие от других фреймворков, обработчики URL в Django конфигурируются явно при помощи регулярных выражений, а не выводятся автоматически из структуры моделей контроллеров.

Для работы с базой данных Django использует собственный ORM, в котором модель данных описывается классами Python, и по ней генерируется схема базы данных.

Архитектура Django похожа на «Модель-Представление-Контроллер» (MVC). Контроллер классической модели MVC примерно соответствует уровню, который в Django называется Представление (англ. View), а презентационная логика Представления реализуется в Django уровнем Шаблонов (англ. Template). Из-за этого уровневую архитектуру Django часто называют «Модель-Шаблон-Представление» (MTV). Первоначальная разработка Django, как средства для работы новостных ресурсов, достаточно сильно отразилась на его архитектуре: он предоставляет ряд средств, которые помогают в быстрой разработке веб-сайтов информационного характера. Так, например, разработчику не требуется создавать контроллеры и страницы для административной части сайта, в Django есть встроенное приложение для управления содержимым, которое можно включить в любой сайт, сделанный на Django, и которое может управлять сразу несколькими сайтами на одном сервере. Административное приложение позволяет создавать, изменять и

удалять любые объекты наполнения сайта, протоколируя все совершённые действия, и предоставляет интерфейс для управления пользователями и группами (с пообъектным назначением прав) [7].

Преимущества использования Django:

- использование Python в качестве языка программирования, а следовательно, все преимущества данного языка;
- качественная документация;
- встроенный ORM (Object-relational mapper) – технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных»;
- автоматически генерируемый интерфейс администратора;
- поддержка MVT (Model-Template-View) – паттерн проектирования, предполагающий использование моделей, шаблонов и представлений (view-функций);
- наличие встроенного обработчика ошибок;
- удобство настройки;
- широкое коммьюнити разработчиков;
- высокая скорость работы.

2.1.3 Система контроля версий Git и веб-сервис для создания удаленного репозитория GitHub

Git — распределённая система управления версиями. Проект был создан Линусом Торвальдсом для управления разработкой ядра Linux, первая версия выпущена 7 апреля 2005 года. Программа является свободной и выпущена под лицензией GNU GPL версии 2 [8].

При работе над одним проектом как команде разработчиков, так и отдельного программиста, необходим инструмент для совместного написания, бэкапирования и тестирования программного обеспечения. Используя Git, мы имеем:

- возможность удаленной работы с исходными кодами;
- возможность создавать свои ветки, не мешая при этом другим разработчикам;
- возможность бэкапирования проекта;
- доступ к последним изменениям в коде;
- возможность откатиться к любой стабильной стадии проекта.

Для создания удаленного репозитория использовался веб-сервис GitHub.

GitHub — веб-сервис для хостинга IT-проектов и их совместной разработки. Основан на системе контроля версий Git и разработан на Ruby on Rails и Erlang компанией GitHub, Inc (ранее Logical Awesome). Сервис абсолютно бесплатен для проектов с открытым исходным кодом

Создатели сайта называют GitHub «социальной сетью для разработчиков». Кроме размещения кода, участники могут общаться, комментировать правки друг друга, а также следить за новостями знакомых. С помощью широких возможностей Git программисты могут объединять свои репозитории — GitHub предлагает удобный интерфейс для этого и может отображать вклад каждого участника в виде дерева [9].

3 Особенности работы Django

Для начала необходимо было создать директорию для хранения проекта, создать виртуальную изолированную оболочку, установить в нее Django, создать и настроить проект. После создания основного шаблона проекта были прописаны необходимые модели, шаблоны и представления, необходимые для осуществления поставленной задачи. Рассмотрим подробнее принципы и особенности работы Django.

При начальном запуске сервера соответствующий скрипт ищет в текущем каталоге (в том же, где находится `manage.py`) файл `settings.py`, где указаны основные настройки проекта. Оттуда он считывает основные параметры, такие как `DATABASES` (параметры соединения с БД) и `TEMPLATES` (директории с html-шаблонами и стилями CSS). Значение данных параметров можно увидеть на рисунках 3.1 и 3.2 соответственно.

```
DATABASES = {  
    'default': {  
        'ENGINE': 'django.db.backends.sqlite3',  
        'NAME': os.path.join(BASE_DIR, 'db/test.db'),  
    }  
}
```

Рисунок 3.2 – Значение параметра DATABASES

```
58 ▼ TEMPLATES = [  
59 ▼     {  
60         'BACKEND': 'django.template.backends.django.DjangoTemplates',  
61         'DIRS': [os.path.join(BASE_DIR, 'static', 'templates')],  
62         'APP_DIRS': True,  
63 ▼         'OPTIONS': {  
64 ▼             'context_processors': [  
65                 'django.template.context_processors.debug',  
66                 'django.template.context_processors.request',  
67                 'django.contrib.auth.context_processors.auth',  
68                 'django.contrib.messages.context_processors.messages',  
69             ],  
70         },  
71     },  
72 ]  
73
```

Рисунок 3.2 – Значение параметра TEMPLATES

Самым главным параметром является `ROOT_URLCONF`, который указывает Django, какой модуль Python следует использовать в качестве привязки для данного сайта. Когда приходит запрос на определенный URL, Django загружает файл привязок, указанный параметром `ROOT_URLCONF` (рис. 3.3).

```
56 ROOT_URLCONF = 'web_interface.urls'
```

Рисунок 3.3 – Значение параметра `ROOT_URLCONF`

Затем Django проверяет каждый шаблон этого файла по порядку, сравнивая запрошенный URL с шаблонами (рис. 3.4), пока не найдет подходящий.

```
6 urlpatterns = [
7     # Examples:
8     url(r'^$', 'web_app.views.home', name='home'),
9     url(r'^search$', 'web_app.views.search', name='search'),
10    # url(r'^blog/', include('blog.urls')),
11
12    url(r'^admin/', include(admin.site.urls)),
13 ]
14
```

Рисунок 3.4 – Шаблоны URL, представленные в виде регулярных выражений

Если совпадение найдено, Django вызывает функцию представления, ассоциированную с шаблоном, передавая ей `HttpRequest` в качестве первого аргумента. Функция представления должна возвращать `HttpResponse`. В данном случае она обращается к шаблону, находящемуся в директории, указанной в `TEMPLATES` (рис. 3.5). Функция загружает базовую страницу, а затем – страницу-наследника.

Удобство использования шаблонов заключается, разумеется, в том, что разработчик не тратит время на повторное написание или копирование кода, а значит, выполняется основной постулат Django – DRY (Don't Repeat Yourself – «не повторяй себя»), что позволяет сэкономить время на

разработку, избежать ошибок и иметь удобную и логичную структуру всего проекта в целом.

Блок-схема описанного процесса работы Django представлена на рисунке 3.6.

```
{% load staticfiles %}

<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <link rel="stylesheet" href="{% static 'css/base_css.css' %}">
  </head>
  <body>
    <form action="/search/" method="post">
      <input type="text" id="fieldsearch" size="30">
      {% csrf_token %}
      <button id="search" >Поиск</button>
    </form>
    <form action="/" method="post">
      {% csrf_token %}
      <button id="refreshtable">Обновить таблицу</button>
    </form>
    <table>
      <tr>
        <th>id</th>
        <th>text</th>
        <th>hashtags</th>
        <th>created at</th>
      </tr>
      {% for item in latest_question_list %}
      <tr>
        <td>{{ item.tweet_id }}</td>
        <td>{{ item.tweet_text }}</td>
        <td>{{ item.hashtags }}</td>
        <td>{{ item.created_at }}</td>
      </tr>
      {% endfor %}
    </table>
  </body>
```

Рисунок 3.5 – Шаблон html-страницы, который, в свою очередь, подгружает шаблоны-наследники

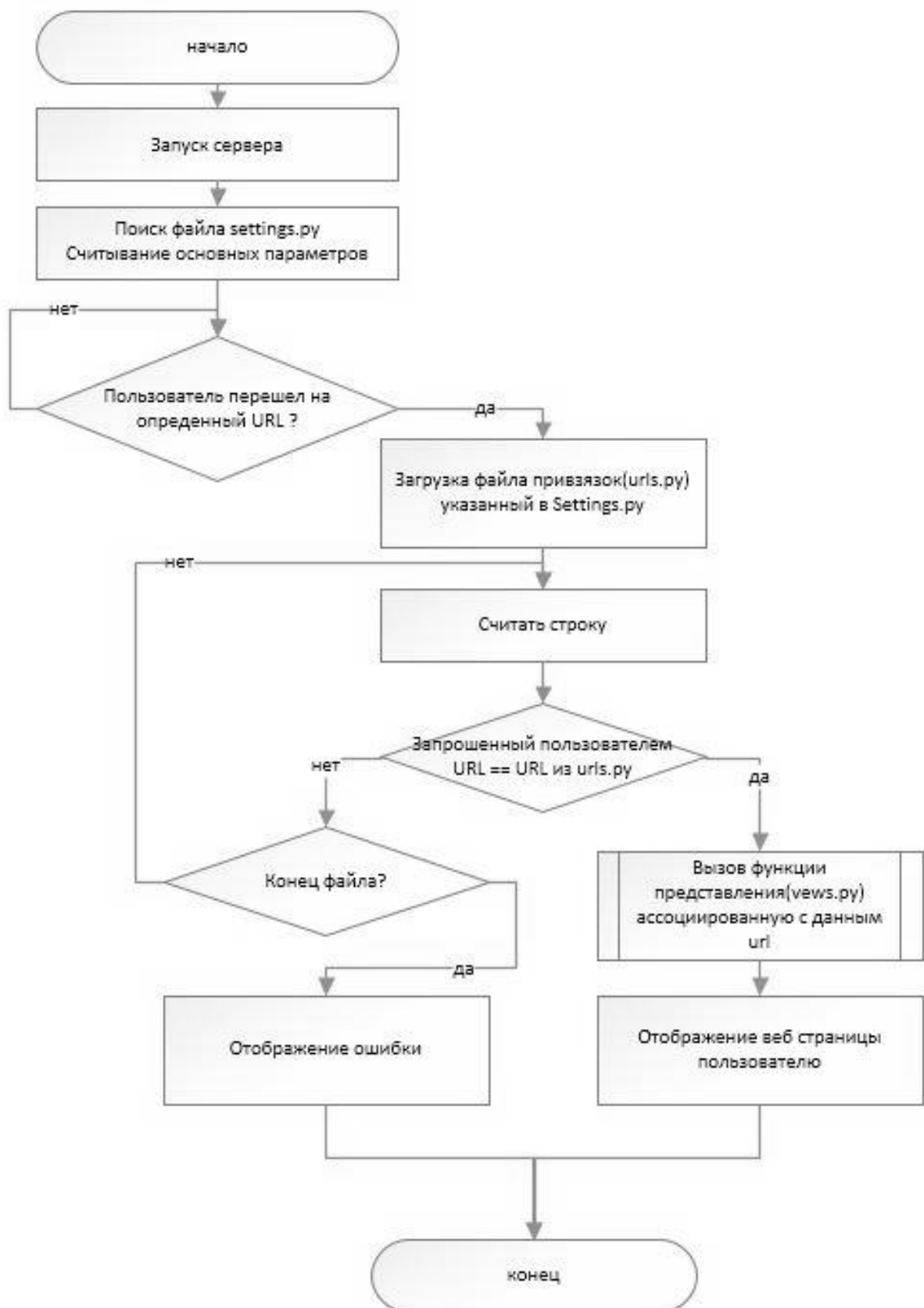


Рисунок 3.6 – Блок-схема процесса работы Django

4 Проектирование базы данных. Создание моделей в Django

Модели отображают информацию об обрабатываемых данных и содержат поля и поведение этих данных. Обычно одна модель представляет одну таблицу в базе данных и описывается классом python. С помощью данного класса существует возможность создавать, получать, обновлять и удалять записи в таблице базы данных, используя простой код на языке Python вместо использования повторяющихся SQL-команд.

Особенности моделей в Django:

- 1) каждая модель — это класс, унаследованный от модели `django.db.models.Model`;
- 2) каждый атрибут модели представляет собой поле в базе данных;
- 3) Django предоставляет автоматически созданное API для доступа к данным.

В качестве базы данных была выбрана SQLite 3, поскольку она является компактной, встраиваемой, легко переносимой и производительной. Встраиваемая означает, что SQLite не использует парадигму клиент-сервер, движок SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а предоставляет библиотеку, с которой программа компонуется и движок становится составной частью программы. SQLite является бестиповой базой данных. Точнее, есть только два типа – целочисленный («integer») и текстовый («text»). Причём «integer» используется преимущественно для первичного ключа таблицы, а для остальных данных – «text». Длина строки, записываемой в текстовое поле, может быть любой. Поскольку движок базы и интерфейс к ней реализованы как единое целое, огромным преимуществом SQLite является высокая производительность.

Модели создаются в файле `models.py` (рис. 4.1) и «мигрируют» в базу данных с помощью специального скрипта `migrate`.

```

1 from django.db import models
2
3 # Create your models here.
4 class Tweets(models.Model):
5     tweet_id = models.BigIntegerField(primary_key = True)
6     tweet_text = models.CharField(max_length = 140)
7     hashtags = models.TextField()
8     created_at = models.TextField()
9     user_name = models.TextField()
10    lang = models.TextField()
11    time_zone = models.TextField()
12    location = models.TextField()
13

```

Рисунок 4.1 – Файл models.py

В данной модели Tweets присутствуют следующие поля:

- Tweet.id – уникальный номер твита;
- Tweet_text – текст твита;
- Hashtags – хештеги используемые в твите;
- Created_at – время и дата опубликования;
- User_name – имя пользователя;
- Lang – язык сообщения;
- Time_zone – часовой пояс;
- Location – местоположение.

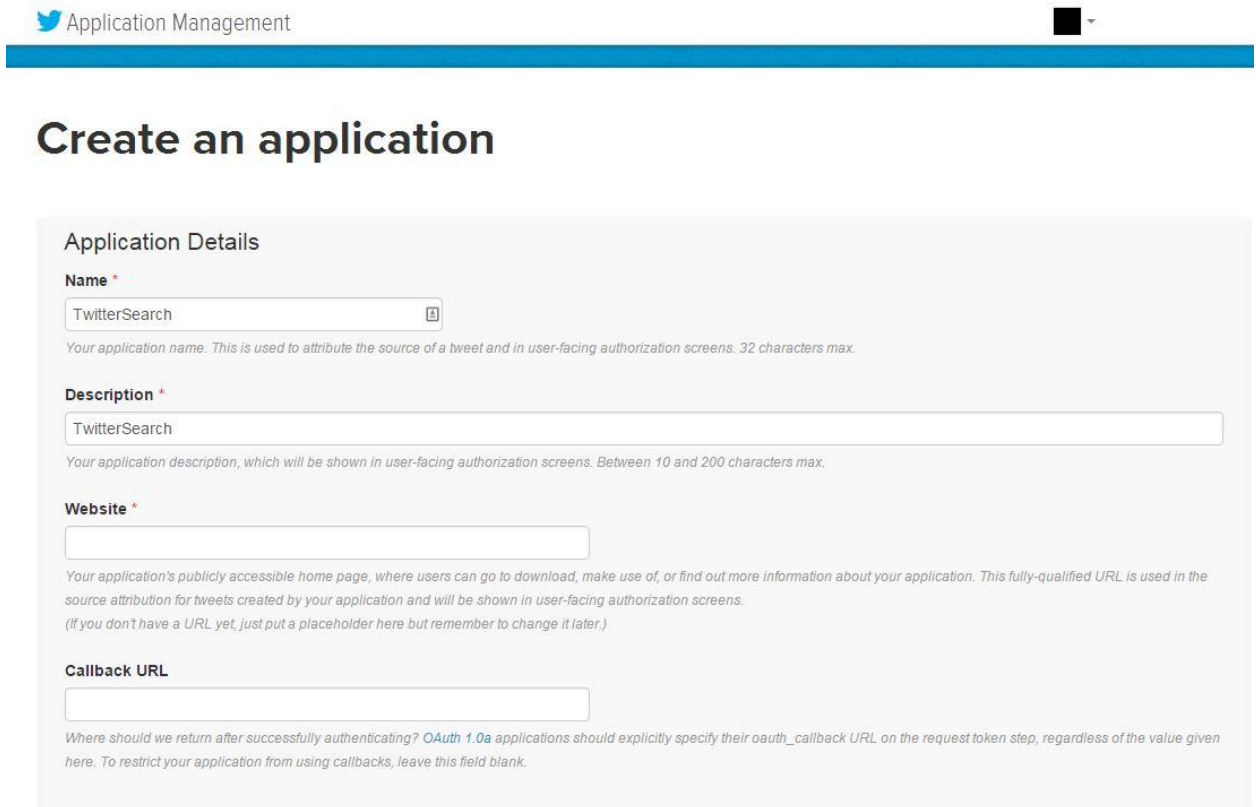
Содержание соответствующей таблицы в базе данных можно увидеть на рисунке 4.2.

rowid	tweet_id	tweet_text	hashtags	created_at	user_name	lang	time_zone	location
1	6394122...	3 сентября. П...	3сентября,с...	2015-09-03 12:...	Irina Antip...	ru	Bangkok	Tomsk
4	6387773...	Началась нов...	тусур	2015-09-01 18:...	Вероника...	ru	None	Томск
5	6386485...	Отличный ден...	ТУСУР,Том...	2015-09-01 09:...	Irina Antip...	ru	Bangkok	Tomsk
6	6386364...	Теперь я с ва...	РКФ,ТУСУР	2015-09-01 08:...	Marina Se...	ru	Bangkok	
7	6385695...	Все на парах, ...	каникулы,с...	2015-09-01 04:...	Ксения Гл...	ru	Bangkok	Северск
10	6372846...	Завтра уже ед...	ТУСУР,LIVE	2015-08-28 15:...	Максим ...	ru	Abu Dhabi	Прокопьевск-Томск
11	6369289...	Откапал свою...	такт,тусур,т...	2015-08-27 15:...	Dmitry Le...	ru	Moscow	Tyumen Salekhard ...
12	6391195...	#профсоюз #т...	профсоюз,т...	2015-09-02 16:...	не рич бич	ru	Bangkok	

Рисунок 4.3 – Содержание таблицы базы данных с полученными данными

5 Twitter API. Программа для поиска сообщений по ключевому слову

Для начала необходимо зарегистрировать приложение в twitter application management (рис.5.1) [10].



The screenshot shows the 'Create an application' page in the Twitter Application Management interface. The page has a blue header with the Twitter logo and 'Application Management' text. Below the header, the title 'Create an application' is displayed. The form is titled 'Application Details' and contains four main sections: 'Name', 'Description', 'Website', and 'Callback URL'. Each section has a text input field and a small informational note below it. The 'Name' field contains 'TwitterSearch'. The 'Description' field also contains 'TwitterSearch'. The 'Website' and 'Callback URL' fields are empty. The form is styled with a light gray background and blue accents.

Application Management

Create an application

Application Details

Name *

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website *

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens.
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL

Where should we return after successfully authenticating? OAuth 1.0a applications should explicitly specify their oauth_callback URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

Рисунок 5.1 – Регистрация приложения

Результатом регистрации является получение API-ключа (рис. 5.2), который состоит из четырех блоков:

- API key;
- API secret;
- Access token;
- Access token secret.

Эти данные необходимы для аутентификации приложения в социальной сети Twitter с целью получения возможности использовать twitter-API функции.

TwitterSearchBot

[Test OAuth](#)
[Details](#) [Settings](#) [Keys and Access Tokens](#) [Permissions](#)

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key)	OUbN0hOILkmGloZue1WglHEdQ
Consumer Secret (API Secret)	uc37EPwzBJDMQ4qYuDyGIK9cb2akIdu9dStT0A0ktK9VqKedmF
Access Level	Read and write (modify app permissions)
Owner	J3onX
Owner ID	1545853322

Application Actions

[Regenerate Consumer Key and Secret](#) [Change App Permissions](#)

Your Access Token

This access token can be used to make API requests on your own account's behalf. Do not share your access token secret with anyone.

Access Token	1545853322-YL8N6URUxoTQ47LHI8sYnOQA8NoS42bwEYOX46w
Access Token Secret	WFoJ8UirQpk0UWxlSzlUwWKHSBJcs1KFuaX2e1W25frxS
Access Level	Read and write
Owner	J3onX
Owner ID	1545853322

Рисунок 5.2 – API-ключ

Блок-схема алгоритма работы программы поиска сообщений по ключевому слову представлена на рисунке 5.3, программный код – на рисунке 5.4, результат работы веб-приложения – на рисунке 5.5.



Рисунок 5.3 – Блок-схема алгоритма работы программы для поиска сообщений по ключевому слову

```

views.py
1  #!/usr/bin/python
2  # -*- coding: utf8 -*-
3
4  from django.shortcuts import render, render_to_response, RequestContext ,loader,HttpResponse
5  from .models import Tweets
6  import forms
7  import tweepy
8  import time
9  import sqlite3
10
11 def home(request):
12     latest_question_list = Tweets.objects.all()
13     template = loader.get_template('index.html')
14     context = RequestContext(request, {'latest_question_list': latest_question_list,})
15     return HttpResponse(template.render(context))
16
17 def search(request):
18     consumer_key = 'OUBN0h0ILkmGloZue1WglHEdQ'
19     consumer_secret = 'uc37EPwzBJDMQ4qYuDyGIK9cb2akIdu9dStT0A0ktK9VqKedmF'
20     access_token = '1545853322-YL8N6URUxoTQ47LHI8sYn0QA8NoS42bwEY0X46w'
21     access_token_secret = 'WfoJ8UirQpk0UWxLSzjUwWKHSBJcs1KFuaX2e1W25frxS'
22     auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
23     auth.set_access_token(access_token, access_token_secret)
24     api = tweepy.API(auth)
25
26     results = api.search(q = "ведьмак", count = 30)
27     for result in results:
28         hash_list=[]
29         hash_str=""
30         for hashtags in result.entities['hashtags']:
31             hash_list.append(hashtags['text'])
32             hash_str = ",".join(hash_list)
33         q = Tweets(tweet_id=result.id_str, tweet_text=result.text, hashtags=hash_str, created_at=str(result
34         q.save()
35     return(home(request))

```

Рисунок 5.4 – Исходный код программы

<div> <div>192.168.1.36:8000</div> <div>Поиск</div> <div>Обновить таблицу</div> </div>				
id	text	hashtags	created at	user
639412260490858496	3 сентября. Пишу деловое письмо по КРИДО. Да, да я теперь деловая дама))) #3сентября #студентка #ТУСУР #Томск	3сентября,студентка,ТУСУР,Томск	2015-09-03 12:18:57	Irin Antip
638777388801859584	Началась новая жизнь)отличная группа и прекрасный факультет,я рада© #тусур http://t.co/vTVqpb08DK	тусур	2015-09-01 18:16:12	Верон Аверк
638648501207965696	Отличный день. Отличная группа) #ТУСУР #Томск #1сентября2015 #студентка http://t.co/i6882LYQL8	ТУСУР,Томск,1сентября2015,студентка	2015-09-01 09:44:02	Irin Antip
638636475278577664	Теперь я с вами #ПКФ #ТУСУР http://t.co/n4dGCFHclF	ПКФ,ТУСУР	2015-09-01 08:56:15	Mari Serb

Рисунок 5.5 – Результат работы веб-приложения

Заключение

Результатом прохождения производственной практики стало веб-приложение для поиска коротких текстовых сообщений («твитов») в социальной сети «Twitter» с последующим сохранением их в базу данных. Также нами было проведено исследование предметной области проблемы и закреплены знания и навыки в проектировании автоматизированных систем и написании программ.

Список использованных источников

- 1 Официальный сайт КИБЭВС [Электронный ресурс] // kibevs.tusur.ru: образовательный портал. 2015. URL: <http://kibevs.tusur.ru/pages/kafedra/index> (дата обращения: 9.07.2015).
- 2 The Stanford Natural Language Processing Group [Электронный ресурс] // nlp.stanford.edu: образовательный портал. 2015. URL: <http://nlp.stanford.edu/software/corenlp.shtml> (дата обращения: 30.06.2015).
- 3 X международная научно-практическая конференция «Электронные средства и системы управления» (ТУСУР) [Электронный ресурс] // www.tusur.ru: образовательный портал. 2015. URL: <http://www.tusur.ru/ru/science/events/conferences> (дата обращения: 30.06.2015).
- 4 Python Software Foundation. Official website [Электронный ресурс] // www.python.org: официальный портал с документацией. 2015. URL: <https://www.python.org> (дата обращения: 1.07.2015).
- 5 Twitter Developers [Электронный ресурс] // dev.twitter.com: официальный портал для разработчиков. 2015. URL: <https://dev.twitter.com/overview/api/twitter-libraries> (дата обращения: 2.07.2015).
- 6 Tweepy: Twitter for Python! [Электронный ресурс] // github.com: репозиторий проекта. 2015. URL: <https://github.com/tweepy/tweepy> (дата обращения: 2.07.2015).
- 7 Django official website [Электронный ресурс] // www.djangoproject.com: официальный портал с документацией. 2015. URL: <https://www.djangoproject.com> (дата обращения: 1.07.2015).
- 8 Git official website [Электронный ресурс] // git-scm.com: официальный портал с документацией. 2015. URL: <https://git-scm.com> (дата обращения: 29.06.2015).

- 9 GitHub [Электронный ресурс] // github.com: веб-сервис для разработчиков. 2015. URL: <https://github.com> (дата обращения: 29.06.2015).
- 10 Twitter application management [Электронный ресурс] // apps.twitter.com: официальный портал для разработчиков. 2015. URL: <https://apps.twitter.com> (дата обращения: 2.07.2015).
- 11 Образовательный стандарт ВУЗа //: Стандарт для студентов ВУЗа. – Томск: ТУСУР, 2013.
- 12 Методические указания по проведению производственной практики //: Учебное пособие для студентов ВУЗа. – Томск: ТУСУР, 2013.
- 13 Наш проект на GitHub [Электронный ресурс] // github.com: репозиторий проекта. 2015. URL: https://github.com/MarinaMeyta/social_network_analisys.