В.А. Рубин, В.Р. Власенко

УО «Гомельский государственный университет
имени Франциска Скорины», Гомель, Беларусь

ИНСТРУМЕНТАРИЙ МОДЕЛИРОВАНИЯ И ИССЛЕДОВАНИЯ ПОЛЬЗОВАТЕЛЬСКОЙ АКТИВНОСТИ В ГЛОБАЛЬНОЙ СЕТИ

Сегодня любое продвижение интернет-ресурса, которое базируется на покупке ссылок, сталкивается с тем, что поисковики придумывают все новые и новые алгоритмы ранжирования и фильтрации, поэтому продвигаться становится все сложнее и сложнее. Приходится работать все с большим количеством пессимизированных сайтов, придумывать, как вывести их из-под санкций поисковых систем, и при всем при этом результат никогда невозможно предсказать, и, конечно же, все это сказывается на получаемом доходе ресурса.

Несмотря на все усилия поисковиков, направленные на борьбу с попытками оптимизаторов влиять на поисковую выдачу с помощью использования внешних ссылок, бэклинки остаются одним из ключевых факторов, которые учитываются при ранжировании сайтов. Последние тенденции в поисковой оптимизации подтверждают смещение в сторону качества, а не количества, поскольку ссылки, которые, по мнению поисковой системы, не являются достаточно качественными или «спамными», просто теряют «вес», и становятся абсолютно бесполезными. Больше того, за попытки манипулирования поисковой выдачей на штраф могут быть наложены штрафные санкции, что отразиться на позициях и усложнит продвижение.

Существует огромное количество ресурсов, которые помогают более эффективно использовать поисковую оптимизацию (search engine optimization, далее по тексту SEO). Например:

- Ahrefs (http://ahrefs.com)
- Plagspotter
- SemRush
- WhiteSpark

Каждый из указанных сервисов позволяет вести учет метрик ресурса и его отдельных страниц по различным параметрам.

К примеру, сервис Ahrefs является инструментом для анализа бэклинков, ведущих на сайт, с помощью которого можно анализировать ссылочную массу сайтов-конкурентов, и на основе полученных данных вырабатывать или совершенствовать свою стратегию наращивания внешних ссылок.

Защита авторских прав и уникальность контента являются одними из наиболее острых проблем, которые затрагивают множество пользователей интернета. На сегодняшний день разработано множество различного программного обеспечение и веб-сервисов, которые предоставляют возможность решать такие проблемы. Одним из таких веб-сервисов стал ресурс от компании Plagspotter. Изначально, данный функционал разрабатывался для личного использования внутри компании, но после проведения сложной работы, было принято решение представить данное «изобретение» каждому желающему. Благодаря веб приложению Plagspotter, каждый желающий может проверить тексты сайта, указывая адрес страницы или же провести полную проверку сайта. После ввода необходимого для проверки адреса, Plagspotter начинает выделять текст со страниц, разбивает его на несколько частей и проверяет на плагиаты, используя сразу три поисковых системы, это Google, Bing и Yohoo. Так же хотелось бы отметить, что проверяемые тексты проходят дополнительную проверку на опечатки и перестановку слов, благодаря чему пользователи добиваются уникальности текстов.

WhiteSpark предоставляет инструментарий для локального SEO-исследования бизнес-рынка. С помощью сервиса Whitespark можно найти упоминания вашего бизнеса локально(в рамках города или страны), с удобной визуализацией полученных данных, а так же четко отслеживать свой рейтинг в любом городе через карты.

SemRush – это сервис анализа ключевых слов, который имеет интуитивный интерфейс. Вводите домен, кликаете на кнопку и получаете данные позиций, анализ качества ключевых запросов, некоторые данные по конкурентам и многое-многое другое. Работает этот сервис пока только с Google. С помощью SemRush можно:

1. Быстро собрать список ключевых слов из Google.ru, по которым ваш сайт сейчас находится в топе с точным указанием страниц. Также в итоговом отчете можно увидеть частотность запроса по Google и среднюю стоимость клика в AdWords по указанному запросу. Есть такие показатели как конкуренция запроса, количество сайтов в выдаче по этому запросу, общее число трафика запроса по отношению ко всему трафику с Google на сайт.

2. Можно посмотреть конкурентов сайта. А после оценить каждого и посмотреть запросы, по которым он продвигается. Также можно увидеть общие запросы с конкурентами данного сайта. Для этого надо кликнуть по вкладке “Конкуренты в поисковой выдаче” и далее выбрать домен, с которым необходимо провести сравнение.

В результате, используя вышеуказанные сервисы и предоставляемый ими интерфейс программирования приложений (далее по тексту API), можно реализовать веб-сервис, который позволит анализировать всевозможные SEO-метрики сайта в едином месте, без необходимости отслеживания отдельных метрик по отдельным ресурсам.

Для реализации такого проекта были выбраны следующие инструменты:

1. Язык программирования Python.
2. Веб-фреймворк Django.
3. База данных PostgreSQL.
4. JavaScript-фреймворк AngularJS.
5. JavaScript-библиотека визуализации данных HighCharts.
6. HTML-шаблонизатор Bootstrap.

Django-фреймворк предоставляет разработчику MVC-подобный шаблон проектирования.

Этот фреймворк базируется на взаимодействии трех компонентов: модели, представление и контроллер. Контроллер принимает запросы, обрабатывает пользовательский ввод, взаимодействует с моделью и представлением и возвращает пользователю результат обработки запроса. Модель представляет

слой, описывающий логику организации данных в приложении. Представление получает данные из контроллера и генерирует элементы пользовательского интерфейса для отображения информации.

Проектирование веб-интерфейса проводилось при помощи библиотеки Twitter Bootstrap, что позволило разработать простейший интерфейс приложения в сжатые сроки.

Для упрощения части приложения, ведущей диалог с клиентом на стороне браузера, был использован AngularJS, что позволило упростить организацию кода на стороне клиента (браузер).

Взаимодействие пользователя с сервисом можно описать так:

1. Клиент выбирает сервис, по которому он желает получить метрики

2. Клиент вводит необходимую информация для запрос к сервису (адрес сайта, адрес страницы сайта, домен ресурса, текст, и т. д.)

3. Запрос клиента записывает в базу данных информацию о нем (сервис, запрос, дата запроса). Так же данный запрос попадает в очередь запросов клиентов

4. Все запросы клиентов поочередно достаются из базы данных, после чего отправляются на сторонний ресурс и после прихода ответа от них – обратно записываются в базу данных к нужному запросу.

5. Клиенту в удобном визуализированном диаграммами виде показывается информация по его запросу, на основании которой он может провести SEO-аудит своего ресурса и выдвинуть дальнейшую стратегию продвижения.

В таком проекте, где необходимо вести постоянный обмен данными со сторонними сервисами, посредством предоставляемого ими API, необходимо заранее приготовиться к серьезным нагрузкам на серверную часть.

В условиях высоких нагрузок недопустимо уже будет использовать синхронную модель взаимодействия, где ваш сервис произведет вызов на сторонний адрес, подождет от него ответ и потом произведет какие-либо операции по получении ответа. Ведь этого ответа можно не дождаться (недоступен ресурс), а возможно сервер просто «ляжет» под такими нагрузками.

Именно для таких нужд применяются асинхронные задания. Особенно красиво они реализованы в Python веб-фреймворке Django, а именно в одной из вспомогательных библиотек – Celery.

Celery – «distributed task queue». Это распределенная асинхронная очередь заданий, которая обладает широким функционалом.

И так, что же умеет Celery:

1. Выполнять задания асинхронно или синхронно.

2. Выполнять периодические задания.

3. Выполнять отложенные задания.

4. Распределенное выполнение (может быть запущен на N серверах).

5. В пределах одного worker'a возможно конкурентное выполнение нескольких задач (одновременно).

6. Выполнять задание повторно, если вылез exception.

7. Ограничивать количество заданий в единицу времени (rate limit, для задания или глобально).

8. Routing заданий (какому worker'у что делать).

9. Несложно мониторить выполнение заданий.

10. Выполнять подзадания.

11. Присылать отчеты об exception'ax на email.

12. Проверять выполнилось ли задание (удобно для построения Ajax приложений, где клиент ждет факта завершения).

Для использования Celery в Django-проекте необходимо использовать сервис очередей. В виду особенностей проекта был выбран RabbitMQ.

Такая организация проекта позволит ему не «упасть» под высокой нагрузкой под натиском большого количества запросов и не даст данным пользователя потеряться в случае неуспешного ответа со стороны сервиса.