Урок 6. Профилирование и нагрузочное тестирование проекта, оптимизация работы с базой данных

1. Установить приложение «django-debug-toolbar». Оценить время загрузки страниц. Найти самые медленные контроллеры. Заполнить таблицу с количеством запросов и дубликатов на страницах проекта.

|  |  |  |  |
| --- | --- | --- | --- |
| **Адрес** | **Число запросов** | **Число дублей** | **После оптимизации** |
| / | 3 | 0 | - |
| /products/ | 6 | 0 | - |
| /authapp/profile/2 (без товаров) | 6 | 2 | - |
| [/authapp/profile/2 (3](http://194.67.111.199/authapp/profile/2%20(3) товара) | 18 | 12 | 2 |
| [/ordersapp/create/](http://194.67.111.199/ordersapp/create/) (3 товара) | 38 | 36 | 3 |
| [/ordersapp/update/3/](http://194.67.111.199/ordersapp/update/3/) (3 товара) | 39 | 32 | 4 |
| /ordersapp/ | 4 | 0 | - |
| /products/detail/2/ | 4 | 0 | - |
| /ordersapp/read/2/ | 8 | 5 | - |

В профиле, я полагаю лишние запросы (ну, в том числе) из-за обращения к цене, соответственно можно использовать selectrelated.

В контроллере заказа товара очень много запросов лишних делается при получении строкового представления продукта, если его убрать, то количество запросов сокращается почти в 3 раза) Это модель продукта, это я думаю просто так не убрать, нужно переделывать решение. Убрал для красоты.

Так же переделал расчет общей суммы и количества товаров в корзине через контекстный процессор.

Изменений немного в пуллреквесте они есть, я взял на себя смелость предположить, что так даже удобнее) если нет, то скажите, я переделаю.

<https://github.com/KTo1/django-optimo/commit/06a07a184d9909002de9724e41360cfcd55ef0c1>

<https://github.com/KTo1/django-optimo/commit/99ce9da9fd0d8b8b1aa2211deaaceb3110855f62>

2. Визуализировать структуру моделей проекта при помощи «django\_extensions», создать файл «geekshop\_urls.txt» с URL-адресами проекта.

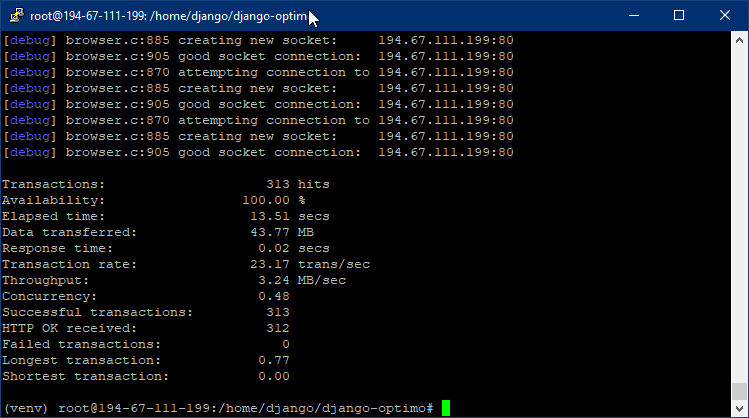
Файлы «gb.png» и «geekshop\_urls.txt».

3. Установить утилиту «siege» и провести функциональное тестирование. Зафиксировать результаты в текстовом файле (какие контроллеры работали с ошибками).

Адреса в файле «urls.txt»

siege -f urls.txt –d1 –r22 -c1 --debug > result

Все хорошо.



4. Провести нагрузочное тестирование отдельных страниц и записать результаты в таблицу.

siege -f urls.txt –d0 –r25 –c50 --debug > result

Папка «test» с файлами отдельных страниц.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Адрес** | **Переходов** | **Время теста, с** | **Транзакций в с** | **Время отклика, с** |
| / | 15100 | 202.43 | 74.59 | 0.66 |
| /ordersapp | 13900 | 235.25 | 59.09 | 0.83 |
| read | 15150 | 326.55 | 46.39 | 1.06 |
| Update | 13900 | 984.53 | 14.12 | 3.48 |
| prod | 461 | 39.91 | 11.55 | 3.88 |
| Cat | 20150 | 246.71 | 81.67 | 0.60 |
| detail | 16400 | 227.46 | 72.10 | 0.68 |
| profile | 16400 | 275.52 | 59.52 | 0.83 |

5. Провести тестирование в режиме интернета. Записать данные в таблицу. Определить условия, при которых начинаются отказы.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Параметр** | **-r50 -c50** | **-r25 -c100** | **r17 -c150** | **-r12 -c200** |
| Число переходов | 34850 | 32471 | 35961 | 34136 |
| Доступность сервера | 100.00 | 99.36 | 100.00 | 99.64 |
| Время теста, с | 628.96 | 638.10 | 498.63 | 459.52 |
| Время отклика, с | 0.90 | 1.94 | 2.02 | 2.61 |
| Запросов в секунду | 55.41 | 50.89 | 72.12 | 74.29 |
| Пропускная способность, МБ/сек | 8.39 | 8.07 | 10.85 | 12.89 |
| Согласованность | 49.64 | 98.73 | 145.44 | 193.66 |
| Удачных транзакций | 34850 | 32471 | 36018 | 34244 |
| Неудачных транзакций | 0 | 210 | 0 | 122 |
| Самая долгая транзакция, с | 39.85 | 60.10 | 60.01 | 60.01 |

Отказы начинаются при числе пользователей более 100 и одновременных запросах более 20.

6. Провести оптимизацию работы с БД в проекте. Оценить эффект.