Домашнее задание 2 “Настройка серверов”

Цель домашнего задания подготовить инфраструктуру для будущего проекта а заодно научиться настраивать web серверы. Внимание! Вместо **pupkin** используйте **свою фамилию**.

1. Создание директории проекта ask_pupkin

С помощью утилиты django-admin нужно создать следущую структуру:

```
ask_pupkin            - директория проекта
 |--- ask_pupkin      - библиотеки проекта (будут созданы django-admin.py)
 |--- manage.py       - скрипт управления (будет создан django-admin.py)
 |--- templates       - шаблоны
 |--- static          - статические файлы (JS, CSS, картинки)
 |--- uploads         - файлы загруженные юзером
```

При этом директории templates, static, uploads — придется создать руками.

2. Настроить gunicorn для запуска WSGI скриптов.

3. Создать простой WSGI скрипт

который:

- Выводит “привет, мир”
- Выводит список переданных GET и POST параметров
- Выполняется при запросе localhost:8081

4. Реализовать пункт 3 с помощью django.

Для этого:

- Создать новую вьюшку которая выводит GET и POST параметры (в views.py)
- Подключить ее к нужному URL в (urls.py)
- Прописать WSGI скрип, созданный Django (скорее всего wsgi.py) в gunicorn.

5. Настройка nginx для отдачи статического контента.

Необходимо настроить nginx следующим образом:
Все файлы с URL начинающимся с /uploads/ отдаются из ask_pupkin/uplaods
Все файлы с расширением (.js .css .jpeg и т.д) — из директории ask_pupkin/static
Файлы должны отдаваться с заголовками, кеширующими файлы на стороне браузера.
Файлы должны сжиматься на сервере для уменьшения размера передаваемых файлов.
Размер конфига nginx не должен превышать 50 строк. Полученную конфигурацию необходимо запустить и проверить, для этого нужно разместить какой-то файл (например sample.html) в директории static и загрузить его с помощью браузера localhost/sample.html

6. Настройка проксирования в nginx.

- настроить nginx для проксирования всех нестатических запросов (URL без расширения, например / или /login/) на gunicorn
- настроить upstream
- настроить proxy_cache и проверить его работу.

7. Сравнение производительности

С помощью утилиты Apache Benchmark (ab, идет в комплекте с Apache) сравните производительность nginx (отдача статики) и gunicorn (запуск wsgi скриптов).

- Насколько быстрее отдается статика по сравнению с WSGI ?
- Во сколько раз ускоряет работу proxy_cache ?

Результатом выполнения домашнего задания является

- Директория с созданным проектом
- Конфиг nginx
- Конфиг gunicorn
- Результаты нагрузочного тестирования (вывод утилиты ab) для nginx и gunicorn

Проект нужно положить в систему контроля версии bitbucket.com или github.com и выслать ссылку своему семинаристу письмом с пометкой **[TP]** в теме. Пометка именно такая, английскими буквами и обязательна.

Полезные ссылки

- nginx nginx.org/ru/docs/

- О проксирования в nginx — nginx.org/ru/docs/http/ngx_http_proxy_module.html#example
- gunicorn docs.gunicorn.org/en/latest/configure.html