Сотый хоуту. Приблизительно такой расклад (не по шагам ТЗ):

1. Пишем wsgi-application (hello.py). Это, наверное, самый, на текущем шаге, простой этап. За основу можно взять кусок [отсюда](https://gunicorn.org/#quickstart). В видео и в комментах есть подсказки (это относится и к остальным пунктам), повторяться не буду. Тут главное не забыть про то, что необходимо полученную строку, **перед возвратом закодировать**.

2. nginx.conf - тут всё просто. Копипастим прям целиком (кроме последней скобки) [отсюда](https://gunicorn.org/#deployment) и совсем чуть чуть (буквально одно слово и несколько циферок) меняем в соответствии с ТЗ.

3. init.sh - тут помимо своих задач нужно **разлинковать дефолтный конфиг** nginx, прилинковать свой, перезапустить nginx.

Хочется отметить, что с конфигом для зеленого единорога я не стал заморачиваться, поскольку во-первых, как ранее отмечалось, сейчас что-то вроде демон не идёт в комплекте и примера конфига не найти после установки. А во-вторых не так и много параметров нужно указать, так что расписывать при запуске параметр -конфиг файл\_конфига или -w количество\_воркеров -b 'х.о.с.т:порт' не такая уж и большая разница. Как бы то ни было я выбрал вариант с указанием параметров при запуске, но если хочется, то: раз. в комментах народ уже прилично обсосал эту тему; два. есть документация, где конечно же всё расписано.

4. Можно и даже желательно проверить всё локально у себя. Для этого нужно установить nginx, который есть в репозах и относительно свежий. Конечно должен быть установлен python3 вместе с pip, установить gunicorn. На ВМ в степике это всё предустановлено искаропки.

    4.1. повторить шаги из инит-скрипта применительно к своей системе

    4.2. запустить gunicorn с параметрами

    4.3 в соседнем терминале сделать запрос curl -vv '127.0.0.1:8080/?a=1&a=2&b=3' и убедиться что возвращаются верные значения. Проверить для прокси\_пасс таким же способом.

А дальше можно запускать таймер и гит клон и баш инит и гуникорн...

Специально не стал в подробностях расписывать, потому что сам собирал инфу из пройденного материала, комментов (спасибо комментаторам) и документации (в меньшей степени) и это доставило. А что напрягло, дак это отсутствие вышеизложенных основных шагов, вот их и изложил.

Хорошо помогли советы из [поста](https://stepik.org/lesson/14826/step/11?discussion=1353968&unit=4175), оказались очень актуальны.

Если вдруг кому-нибудь понадобится, так как курс уже устарел:

1) **забейте на gunicorn.conf!!!! он вам не понадобится (у меня пустой)**

**/etc/gunicorn.d у меня нет, да и вообще теперь служба по умолчанию не ставится.**

кому интересно: правильно его разворачивать в docker или в python виртуальном окружении. <https://docs.gunicorn.org/en/stable/deploy.html#systemd> - правильная настройка в systemd. судя по инструкции теперь надо явно указывать приложения, а также создать службу gunicorn.socket которая и будет кушать proxy\_pass с nginx

2) создаете файл hello.py. Примера из вики или лекций достаточно. но учтите что в теперь данные в ответе передаются в bytes. Пример:

def app(environ, start\_response):

"""Simplest possible application object"""

data = b'Hello, World!\n'

status = '200 OK'

response\_headers = [

('Content-type', 'text/plain'),

('Content-Length', str(len(data)))

]

start\_response(status, response\_headers)

return iter([data])

Подсказки:

а) для начала попробуйте print(environ), и найти переданные параметры ( [/?a=1&a=2&b=3](http://127.0.0.1/?a=1&a=2&b=3))

b) у строк есть функция split

c) ответ представляет собой **строку.**объединение строк join

d) bytes(СТРОКА) - перевод в bytes

3) запустите gunicorn ПАРАМЕТРЫ & (& - чтобы приложение работало в фоне)

P.S.на домашнем компе python3.7 gunicorn 19.9.0