Министерство науки и высшего образования Российской Федерации

ФГАОУ ВО «УрФУ имени первого Президента России Б.Н. Ельцина»

Нижнетагильский технологический институт (филиал)

Кафедра информационных технологий

Оценка работы \_\_\_\_\_\_\_\_\_\_\_\_

Руководитель от УрФУ Карелова Р.А.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2022 г.

Загрузка Backend приложения на хостинг сервер OC Ubuntu

ОТЧЕТ СТУДЕНТА

по производственной практике

Руководитель от предприятия/организации Карелова Р.А.

Студент Паньишн А.А.

Специальность (направление подготовки) 09.03.02 ИСТ

Группа Т-393901

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_2022 г. Подпись студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Тагил

2022

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 3](#_Toc114399869)

[ТЕОРЕТИЧЕСКИЕ АСПЕКТЫ РАЗРАБОТКИ 4](#_Toc114399870)

[РЕАЛИЗАЦИЯ 6](#_Toc114399871)

[ЗАКЛЮЧЕНИЕ 11](#_Toc114399872)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 12](#_Toc114399873)

# ВВЕДЕНИЕ

Данный отчет – продолжение работы над веб-приложением для магазина «Ника». В данный момент у нас на руках имеется готовая «Backend» часть приложения, которую необходимо загрузить на хостинговый сервис для публичного пользования и удобной разработки со стороны «Frontend» разработчика.

До этого момента вы работали в каком-то рабочем окружении - чтобы получать отладочную и другую частную информацию, мы использовали веб-сервер Django в локальной сети при этом запускали сайт с (небезопасными) настройками разработки. Перед тем как разместить сайт публично, вы должны сделать следующее:

* Сделать несколько изменений в настройках проекта.
* Выбрать/Настроить окружение для хостинга приложения Django.
* Выбрать/Настроить окружение для размещения статических файлов.
* В целях обслуживания сайта настроить инфраструктуру для его развёртывания.

# ТЕОРЕТИЧЕСКИЕ АСПЕКТЫ РАЗРАБОТКИ

Окружение развёртывания — это среда, которое предоставляет сервер, на котором вы будете размещать свой веб-сайт для публичного запуска и доступа. Данное окружение включает в себя: [1]

* Характеристики аппаратного обеспечения, на котором будет запускаться сайт.
* Операционную систему (Linux).
* Языки программирования времени выполнения (скриптовые) и библиотеки, которые использует ваш сайт.
* Веб-сервер, используемый для обслуживания страниц и другого контента (Nginx).
* Сервер приложений, который передаёт "динамические" запросы между сайтом Django и веб-сервером.
* Базу данных, от которой зависит ваш сайт (postgresql)

nginx [engine x] — это HTTP-сервер и обратный прокси-сервер, почтовый прокси-сервер, а также TCP/UDP прокси-сервер общего назначения. [2]

Основная функциональность HTTP-сервера

* Обслуживание статических запросов, индексных файлов, автоматическое создание списка файлов, кэш дескрипторов открытых файлов;
* Акселерированное обратное проксирование с кэшированием, распределение нагрузки и отказоустойчивость;
* Акселерированная поддержка FastCGI, uwsgi, SCGI и memcached серверов с кэшированием, распределение нагрузки и отказоустойчивость;
* Модульность, фильтры, в том числе сжатие (gzip), byte-ranges (докачка), chunked ответы, XSLT-фильтр, SSI-фильтр, преобразование изображений; несколько подзапросов на одной странице, обрабатываемые в SSI-фильтре через прокси или FastCGI/uwsgi/SCGI, выполняются параллельно;
* Поддержка SSL и расширения TLS SNI;
* Поддержка HTTP/2 с приоритизацией на основе весов и зависимостей.

**Secure Sockets Layer** (SSL) - это криптографический протокол, обеспечивающий безопасное общение пользователя и сервера по небезопасной сети. Располагается между транспортным уровнем и уровнем программы-клиента [3]

PostgreSQL — свободная объектно-реляционная система управления базами данных. Существует в реализациях для множества UNIX-подобных платформ, включая AIX, различные BSD-системы, HP-UX, IRIX, Linux, macOS, Solaris/OpenSolaris, Tru64, QNX, а также для Microsoft Windows. [4]

Gunicorn — это WSGI-сервер, созданный для использования в UNIX-системах. Название — сокращенная и комбинированная версия слов «Green Unicorn». На самом сайте проекта есть зеленый единорог. Gunicorn был перенесен из проекта «Unicorn» из языка Ruby. Он относительно быстрый, ресурсоёмкий, легко запускается и работает с широким спектром веб-фреймворков. [5]

# РЕАЛИЗАЦИЯ

Начнем с установки пакетов (вся работа будет вестись в рамках OC Ubuntu через удаленный доступ). (См. листинг 1)

apt-get update

apt-get install nginx

Листинг 1 – Установка nginx

В нашем случае мы будем хранить файлы сайта в каталоге /var/www/, поэтому перемещаемся в каталог и создаем новый каталог (см. листинг 2)

cd /var/www/  
mkdir nikka

Листинг 2 – Создание каталога для файлов сайта

В данном каталоге активируем виртуальное окружение и устанавливаем в него Django, и сразу же устанавливаем пакет gunicorn для Django (См.листинг 3)

source geekhero\_env/bin/activate   
pip install Djangopip install gunicorn

Листинг 3 – Установка Django и gunicorn

Теперь мы импортируем в данную папку проект, мы это сделаем с помощью системы контроля версий git. (См. листинг 4)

git clone https://github.com/Weenty/nikka.git

Листинг 4 – Клонирование проекта в каталог

Заходим в Settings.py и прописываем (См. листинг 5). Это необходимо для корректной работы Django с файлами.

STATIC\_URL = '/static/'

STATIC\_ROOT = os.path.join(BASE\_DIR, 'static/')

Листинг 5 – Настройка Settings.py

Идем в каталог /etc/systemd/system/ и создаем два файла: gunicorn.service и gunicorn.socket. (см. листинг 6, 7)

[Unit]

Description=gunicorn daemon

Requires=gunicorn.socket

After=network.target

[Service]

User=root

WorkingDirectory=/var/www/nikka

ExecStart=/var/www/geekhero/geekhero\_env/bin/ geekhero--workers 5 --bind unix:/run/gunicorn.sock ghproj.wsgi:application

[Install]

WantedBy=multi-user.target

Листинг 6 – Содержимое файла gunicorn.service

[Unit]

Description=gunicorn socket

[Socket]

ListenStream=/run/gunicorn.sock

[Install]

WantedBy=sockets.target

Листинг 7 – Содержимое файла gunicorn.socket

Далее идем в каталог: /etc/nginx/sites-available/ и создаем файл nikka без расширения. (См. листинг 8)

**server** {

**listen** 80;

**server\_name** market.ruban.xyz/;

**location** = /favicon.ico { **access\_log** off; **log\_not\_found** off; }

**location** /static/ {

**root** /var/www/nikka;

}

**location** /media/ {

**root** /var/www/nikka;

}

**location** / {

**include** proxy\_params;

**proxy\_pass** http://unix:/run/gunicorn.sock;

}

}

Листинг 8 – Содержание файла nikka

Для того, чтобы создать символическую ссылку на файл в каталоге /etc/nginx/site-enabled/ используем следующую команду (см. листинг 9)

sudo ln -s /etc/nginx/sites-available/nikka /etc/nginx/sites-enabled/

Листинг 9 – Создание символической ссылки

При любых изменениях оригинального файла, ярлык из sites-enabled нужно удалять и пересоздавать заново командой выше или выполнять команду:

sudo systemctl restart nginx

Проверяем конфигурацию nginx командой

sudo nginx -t

Далее запускаем службу gunicorn и создаем socket:

sudo systemctl enable gunicorn  
sudo systemctl start gunicorn

Также, при изменении любых данных в HTML-шаблонах, необходимо перезапускать сервис gunicorn командой:

service gunicorn restart

Чтобы посмотреть статус запущенного сервиса необходимо использовать команду:

sudo systemctl status gunicorn

Для проверки создания сокета, необходимо ввести команду:

file /run/gunicorn.sock

В нашем случае проблем не возникло и мы можем запускать nginx:

sudo service nginx start

Теперь нам нужно получить сертификат SSL для домена:

Установим certbot от Let's Encrypt:

 sudo apt-get install certbot python-certbot-nginx

Произведем первичную настройку certbot:

sudo certbot certonly –nginx

И наконец автоматически поправим конфигурацию nginx:

 sudo certbot install –nginx

Осталось только перезапустить сервис nginx:

sudo systemctl restart nginx

Для установки дополнительных проектов Django, необходимо повторить весь процесс от создания venv и до установки SSL-сертификата для вашего нашего домена, только название сервиса и сокета нужно изменять с nikka на что-то другое, например:

Сайт1: site1.service и site1.socket  
Сайт2: site2.service и site2.socket

Это будет необходимо, когда на домен будет заливаться помимо Backend-a ещё и Frondend.

Настроим логгирование в Django для откладки ошибок (См.листинг 10)

import os

LOGGING = {

'version': 1,

'disable\_existing\_loggers': False,

'handlers': {

'console': {

'class': 'logging.StreamHandler',

},

},

'root': {

'handlers': ['console'],

'level': 'WARNING',

},

}

Листинг 10 – Настройки логов в Settings.py

Для подключения логгирования в gunicorn добавим в параметры:

error-logfile /var/log/gunicorn/error.log --access-logfile /var/log/gunicorn/access.log --capture-output --log-level debug

# ЗАКЛЮЧЕНИЕ

В условия поставленной задачи по работе с хостинговым сервисом на базе OC Ubuntu были выполнены следующие задачи:

* Была выведена Backend часть веб-приложения «Ника» в production
* Установлен и настроен nginx
* Реализован gunicorn
* Получен и настроен сертификат SSL благодаря certbot от Let's Encrypt
* Были закреплены, полученные в ходе обучения проектирования и разработки, теоретические знания.
* Приобретены важные навыки в работе с OC Ubuntu и Linux Server в целом.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Django Руководство часть 11: Разворачивание сайта на сервере - Изучение веб-разработки | MDN [Электронный ресурс]. URL: https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/Deployment (дата обращения: 18.08.2022).

2. nginx [Электронный ресурс]. URL: https://nginx.org/ru/ (дата обращения: 18.08.2022).

3. Основы HTTPS, TLS, SSL. Создание собственных X.509 сертификатов. Пример настройки TLSv1.2 в Spring Boot // Хабр [Электронный ресурс]. URL: https://habr.com/ru/post/593507/ (дата обращения: 18.08.2022).

4. Сведения о базе данных PostgreSQL | Microsoft Azure [Электронный ресурс]. URL: https://azure.microsoft.com/ru-ru/resources/cloud-computing-dictionary/what-is-postgresql/ (дата обращения: 18.08.2022).

5. Введение в WSGI-серверы: Часть первая / Хабр [Электронный ресурс]. URL: https://habr.com/ru/post/426957/ (дата обращения: 18.08.2022).