Установка веб-приложения на сервер (Python-Flask-Gunicorn-Nginx)

Использование современных фреймворков таких как **Flask** или **Django** позволяют достаточно просто организовать разработку веб-приложения на языке **Python**, используя при этом для тестирования локальные серверы. Такого рода программные серверы, как правило устанавливаются при установке самих фреймворков и их использование не вызывает ни каких затруднений. Однако использование данных серверов для разворачивании приложения не допустимо, т.к. не позволяет обеспечить достаточный уровень безопасности приложения при его использовании.

В данной ситуации потребуется развертывание приложения на web сервере. В качестве такого сервера может быть использован как собственный сервер вашей компании или ваш домашний сервер, как и виртуальный или облачный сервер. Главным условием выбора сервера является наличие у него статического IP адреса.

В данной публикации будет рассмотрен вариант развертывания web приложения на виртуальной машине с установленной операционной системой `Linux Debian 11` с применением серверов **Gunicorn** и **Nginx**

Полный порядок подключения можно найти **здесь**

Базовая установка и настройка веб-сервера

Перед началом разработки приложения (если разработку планируется вести прямо на сервере) или публикации на сервере уже написанного приложения требуется выполнить некоторые подготовительные действия.

1. Создаем инфраструктуру папок, в которых будут находиться все файлы нашего приложения.

В нашем случае будет всего одна корневая директория, размещаемая в одной из директорий созданных в корне жесткого диска.

```
$ sudo mkdir /path/to/folder/my_site
```

2. Изменение владельца папки и прав доступа к ней

```
$ sudo chown user:group /path/to/folder/my_site
```

3. Создание виртуального окружения.

Виртуальное окружение в Python — способ изолировать зависимости (пакеты) для определённого проекта. Создаётся через модуль `venv`, который идёт в поставке `Python 3`.

Используется команда `python -m venv` и название директории, в которой будет создано виртуальное окружение.

```
cd /path/to/folder/my_site
python3 -m venv venv
```

Флаг `-m` указывает Python-у запустить `venv` как исполняемый модуль. `venv/` — название виртуального окружения (где будут храниться ваши библиотеки).

В результате будет создан каталог `venv/` содержащий копию интерпретатора Python, стандартную библиотеку и другие вспомогательные файлы.

4. Активация виртуального окружения

```
$ source /venv/bin/activate
```

5. Установка зависимостей

Для обеспечения процесса разработки веб-приложения установим следующие программные пакеты - `flask, gunicorn, wheel, sqlite3 libsqlite3-dev`. Установку сервера `Nginx` выполним чуть позднее.

Два последних будут использоваться для обеспечения работы из консоли с базой данных `sqlite3` в интерактивном режиме при отсутствии графической оболочки на сервере. Т.е. позволят пр необходимости вносить изменения в используемую базу данных.

```
(venv)$ pip install flask gunicorn wheel sqlite3 libsqlite3-dev
```

> Собственно далее должен выполняться процесс разработки приложения или размещения уже готового приложения в созданной нами директории. Данный процесс по понятным причинам опустим и перейдем сразу к настройке WSGI севера приложений

6. Создаем точку входа в приложение для `gunicorn`

```
(venv)$  nano /opt/my_project/wsgi.py
```

В созданный файл записываем следующий код:

```
from app import app

if __name__ == "__main__":
    app.run()
```

7. Первый запуск `gunicorn`

> В нашем проекте мы будем использовать привязку к ip `0.0.0.0` и порту `8080`

```
(venv)$ gunicorn --bind 0.0.0.0:8080 wsgi:app
```

> Если все было выполнено правильно, то можно будет увидеть подобный вывод на экране терминала:

```
    [2021-11-19 23:07:57 +0000] [8760] [INFO] Starting gunicorn 20.1.0
    [2021-11-19 23:07:57 +0000] [8760] [INFO] Listening at: http://0.0.0.0:5000 (8760)
    [2021-11-19 23:07:57 +0000] [8760] [INFO] Using worker: sync
    [2021-11-19 23:07:57 +0000] [8763] [INFO] Booting worker with pid: 8763
```

Останавливаем `gunicorn` нажав `Ctrl + C` и создаем для него службу

8. Создание службы

> Для постоянной работы сервер wsgi будет запускаться и работать на уровне службы

```
$ sudo nano /etc/systemd/system/my_site.service
```

9. Запуск службы

```
$ sudo systemctl start my_site
```

10. Конфигурирование Gunicorn

10. Создание и запуск демона

```
$ sudo systemctl enable my_site
```

▶ Продолжение

11. Установка сервера `Nginx`

12. Настройка проксирования `Nginx-Gunicorn`

> Дальнейший текс приведен только для проверки использования ***html*** разметки внутри маркдаун

- Пункт 1
- Пункт 2
- Пункт 3
- Пункт 4
- Список второго уровня
    - Подпункт 1
    - Подпункт 2
    - Подпункт 3
- Пункт 3
- Пункт 4

Текст с ^верхним индексом не работает^