Инструкция по разворачиванию проекта на Ubuntu

# Настройка сервера и локального компьютера

Убедитесь в том, чтобы ваш сервер работал и не был приостановлен

В проекте необходимо скопировать .env файл и назвать его .env.dummy и прописать туда шаблонные данные. Этот шаблон нам в дальнейшем понадобится

Для начала необходимо подключиться к серверу по такому сетевому протоколу, который называется SSH.

*SSH (англ. Secure Shell — «безопасная оболочка») — сетевой протокол прикладного уровня, позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений (например, для передачи файлов). Схож по функциональности с протоколами Telnet и rlogin, но, в отличие от них, шифрует весь трафик, включая и передаваемые пароли. SSH допускает выбор различных алгоритмов шифрования. SSH-клиенты и SSH-серверы доступны для большинства сетевых операционных систем.*

Далее в зависимости от того на какой операционной системе вы работаете, вам необходимо скачать следующее приложение

Putty

1. Откройте PuTTY и перейдите в Connection > SSH > Auth.
2. Нажмите кнопку Обзор рядом с полем Private key file for authentication
3. Выберите созданный файл .ppk и нажмите « Открыть »
4. Перейдите в категорию «Сеанс» и введите IP-адрес вашего экземпляра в поле «Имя хоста». В поле «Сохраненные сеансы» дайте этому соединению имя.
5. Нажмите кнопку «Сохранить» справа, чтобы сохранить этот сеанс.
6. Щелкните имя сеанса, которое вы только что сохранили, и нажмите кнопку «Открыть»
7. PuTTY подключается к вашему экземпляру и запрашивает ваше имя пользователя.
8. Введите своего пользователя и нажмите Enter . Теперь вы вошли в экземпляр.

Для обновления всех пакетов запустите команды

apt-get update — обновить доступные обновления

apt-get upgrade — обновить все пакеты

Теперь можно начинать настраивать проект!

# Установка необходимых программ

Для удобства использования терминала установим пакет mc

sudo apt-get install mc

Для начала проверьте какая версия Python у вас установлен

$ python3 --version # или 'python3 -V'

Чтобы установить Django с pip, нам нужно установить пакет python3-pip в систему Ubuntu 18.04.

sudo apt-get install python3-pip

Проекты советую устанавливать в репозиторий home

Для этого перейдем в home и запустим команду

$ mkdir django && cd django

Это создаст нам папку Django и перейдет в него

Теперь, когда у нас установлен pip, это означает, что у нас есть возможность быстро установить другие необходимые пакеты для среды Python.

Но перед установкой Django нам также нужно установить virtualenv. Установите «virtualenv», выполнив команду

$ sudo apt-get install virtualenv

Теперь создайте виртуальную среду с любым именем (например, ‘venv’) с помощью установленного пакета virtualenv, а также активируйте среду, выполнив команду

$ virtualenv -p python3 venv && source venv/bin/activate

При работе в виртуальной среде python автоматически ссылается на правильную версию, поэтому мы можем использовать python вместо python3.

Теперь нам нужно установить различные библиотеки разработки, необходимые для сборки приложения. Они включают python3-dev для компиляции любых расширений Python и поддерживающих их стандартных библиотек, а также пакет libmysqlclient-dev , необходимый для сборки клиента MySQL

Это можно сделать, выполнив следующие команды в терминале

(venv) $ sudo apt-get install python3-dev

(venv) $ sudo apt-get install libmysqlclient-dev

Теперь нам необходимо загрузить наш проект на сервер

Мы будем это делать через Git. Для этого на сервере запускаем

git clone https://your\_repository\_name.git

Если это уже существующий проект, вам может потребоваться подготовить список требований, необходимых для запуска вашего проекта. Обычно вы можете просто создать файл requirements.txt на основе вашей среды Python.

Теперь у нас есть все важные зависимости. Итак, мы наконец-то можем установить Django и все другие инструменты, необходимые для разработки веб-приложения django.

(venv)$ pip install –r requirements.txt

# Установка и настройка MySQL

Теперь давайте установим MySQL. Можем пока деактивировать виртуальную среду

(venv) $ deactivate

$ sudo apt-get install mysql-server

Эта команда установит MySQL, но при этом вам не будет предложено задать пароль или внести какие-либо правки в конфигурацию. Поскольку это делает установку небезопасной, рассмотрим далее, как настроить установленную MySQL.

В случае новой установки MySQL вам необходимо выполнить скрипт безопасности. Он изменяет некоторые настройки по умолчанию на более безопасные, например, удалённый вход для пользователей root и пользователи, созданные по умолчанию. В старых версиях MySQL вам было необходимо также инициализировать директорию данных вручную, теперь это делается автоматически.

Настройка аутентификации и привилегий

На серверах с Ubuntu, использующей MySQL 5.7 (и более поздние версии), пользователь root в MySQL по умолчанию аутентифицируется с помощью плагина auth\_socket, а не по паролю. Это в целом более безопасно и удобно во многих случаях, но не в случае, когда вам необходимо организовать доступ к MySQL со стороны сторонней программы.

Для того, чтобы пользователь root в MySQL мог использовать пароль для входа в систему вам необходимо изменить метод аутентификации с auth\_socket на mysql\_native\_password. Для этого войдите в оболочку MySQL следующей командой:

sudo mysql

Далее изменяем метод аутентификации на метод с паролем

ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql\_native\_password BY 'password';

Далее выполните команду FLUSH PRIVILEGES, которая применит внесённые изменения:

FLUSH PRIVILEGES;

Далее давайте попробуем зайти с нашим новым паролем

exit

mysql -u root –p

Далее создайте нового пользователя и задайте для него надёжный пароль:

CREATE USER 'codifyadmin'@'localhost' IDENTIFIED BY 'password';

Далее задайте для пользователя подходящий набор привилегий. Например, вы можете дать пользователю доступ ко всем таблицам в базе данных, а также права на добавление, изменение и удаление пользовательских привилегий следующей командой:

GRANT ALL PRIVILEGES ON \*.\* TO 'codifyadmin'@'localhost' WITH GRANT OPTION;

С помощью GRANT OPTION пользователь может предоставлять другим пользователям разрешения на доступ к базе данных.

Чтобы выйти из оболочки MySQL, введите команду

exit;

Затем войдите в систему как пользователь, чтобы проверить, создан ли пользователь или нет.

Примечание. Когда вы вводите пароль, он не будет виден.

$ mysql -u codifyadmin -p

Password: \*\*\*\*\*\*\*\*\*\*\*

После входа в систему создайте любую базу данных с именем «django» для нашего проекта. Чтобы создать базу данных в MySQL, выполните команду

create database django;

Чтобы проверить, создана база данных или нет, выполните команду

show databases;

Затем выйдите из оболочки MySQL, выполнив команду EXIT.

exit;

# Связываем базу данных с Django

Удостоверьтесь что в вашем проекте установлен environ

Удостоверьтесь в том, чтобы в настройках вашего проекта были следующие поля

env = environ.Env(DEBUG=(bool, False))

environ.Env.read\_env(os.path.join(BASE\_DIR, '.env'))

SECRET\_KEY = env('SECRET\_KEY')

DEBUG = env('DEBUG')

ALLOWED\_HOSTS = env('ALLOWED\_HOSTS').split(',')

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': env("DB\_NAME"),

'USER': env("DB\_USER"),

'PASSWORD': env("DB\_PASSWORD"),

'HOST': 'localhost',

'CHARSET': 'utf8',

'COLLATION': 'utf8\_general\_ci',

}

}

STATIC\_URL = '/static/'

MEDIA\_URL = ' /media/'

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

MEDIA\_ROOT = os.path.join(BASE\_DIR, 'media')

Далее заходим в папку с нашим проектом и находим там .env.dummy

Переименовываем его в .env и меняем его на нормальные данные

Далее выполняем миграции

(venv) $ python manage.py makemigrations

(venv) $ python manage.py migrate

Теперь нам нужно создать администратора/суперпользователя.

(venv) $ python manage.py createsuperuser

Суперпользователь Django успешно создан, теперь мы можем снова выполнить команду runserver

(venv) $ python manage.py runserver 0.0.0.0:8000

Удостоверяемся что 8000 порт у нас открыт

Если порт закрыт удостоверяемся что порт не блокируется

sudo ufw status numbered

Если нужного нам порта нет в списке, включите его с помощью

sudo ufw allow 8000

Если все работает нормально чтобы собрать статические файлы в настроенном ранее месте, мы можем использовать команду collectstatic

(venv)python manage.py collectstatic

Все статические файлы будут собраны в каталоге, указанном выше. В нашем случае он static.

# Установка и настройка Gunicorn

После успешной настройки вашего проекта Django теперь мы можем установить в проект наш Gunicorn.

pip install gunicorn

Поскольку наше приложение работает нормально с runserver, теперь пришло время запустить его с помощью gunicorn. Gunicorn — это Python WSGI (интерфейс шлюза веб-сервера). Он действует как шлюз для отправки запросов в наше приложение django. Поскольку gunicorn используется здесь как модуль pip, мы должны сначала активировать нашу виртуальную среду для запуска gunicorn.

source venv/bin/activate

(venv)$ gunicorn -b 0.0.0.0:8000 mysite.wsgi:application

Если все работает нормально, то выходим из виртуальной среды

deactivate

переходим по следующему пути и создаем там файл

sudo touch /etc/systemd/system/gunicorn.socket

откроем данные файл через mc нажатием f4

и пропишем следующие конфигурационные данные

[Unit]

Description=gunicorn socket

[Socket]

ListenStream=/run/gunicorn.sock

[Install]

WantedBy=sockets.target

[Unit] — для описания сокета

[Socket] — для определения местоположения сокета

[Установить] — убедиться, что сокет создан в нужное время.

Далее там же создаем файл gunicorn.service

touch gunicorn.service

и добавим туда следующие данные

!#/bin/sh

[Unit]

Description=gunicorn daemon

Requires=gunicorn.socket

After=network.target

[Service]

User=root

Group=www-data

WorkingDirectory=your\_directory\_path/project\_name

ExecStart=your\_directory\_path/project\_name/venv/bin/gunicorn \

--access-logfile - \

--workers 3 \

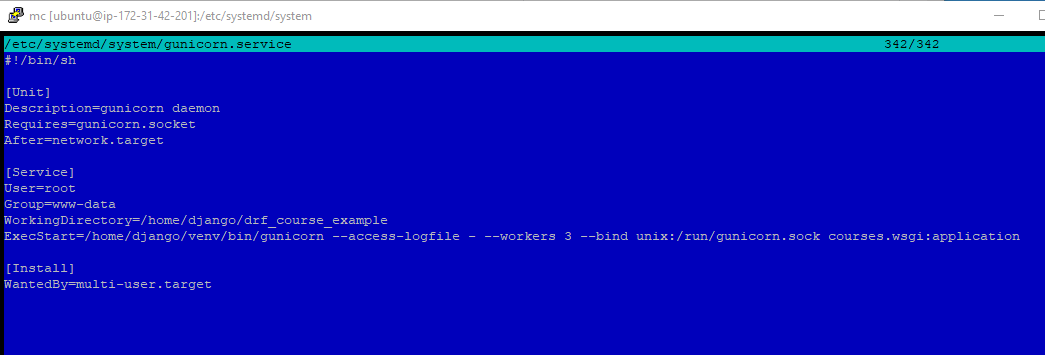
--bind unix:/run/gunicorn.sock \

project\_name.wsgi:application

[Install]

WantedBy=multi-user.target

!#bin/sh Он называется shebang и сообщает родительской оболочке, что интерпретатор должен использоваться для выполнения script.



[Unit] — используется для указания метаданных и зависимостей.

[Service] — указываем пользователя и группу, под которой мы хотим запустить процесс

[Install] — это сообщит системе, с чем связать эту службу, если мы включим ее запуск при загрузке.

Теперь запустите и включите сокет gunicorn

sudo systemctl enable gunicorn.socket

sudo systemctl start gunicorn.socket

Теперь запустите и включите сервис gunicorn

sudo systemctl enable gunicorn.service

sudo systemctl start gunicorn.service

если вы хотите проверить состояние сокета и сервисов. Вы можете использовать эти команды.

sudo systemctl status gunicorn.socket

sudo systemctl status gunicorn.service

# Настройка NGINX

Для начала надо установить nginx

sudo apt-get install nginx

Прямо сейчас нам нужно настроить наш последний компонент — веб-сервер, который обрабатывает наши сетевые запросы. Мы уже установили пакет на первом этапе настройки среды. Теперь давайте создадим файл конфигурации.

sudo touch /etc/nginx/sites-available/your\_project\_name #тут должно быть имя вашего проекта

И добавьте туда эту запись

server{

listen 80;

server\_name your\_domain\_name\_or\_ip\_address;

access\_log /var/log/nginx/access.log;

error\_log /var/log/nginx/error\_log;

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

location /static/ {

root your\_directory\_path/your\_project\_name;

}

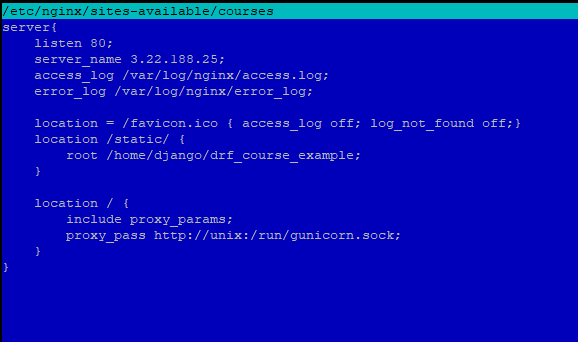
location / {

include proxy\_params;

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

}

}



Сохраните и активируйте файл, связав его с каталогом с поддержкой сайтов. Это необходимо для того, чтобы любые изменения в файле конфигурации были актуальными.

sudo ln -s /etc/nginx/sites-available/**your\_project\_name** /etc/nginx/sites-enabled

Теперь перезапустите сервер Nginx.

sudo systemctl restart nginx

Бывают случаи, когда у нас может быть некоторая опечатка в синтаксисе в файле конфигурации, вы можете проверить это с помощью этой команды.

sudo nginx -t

Наконец, нам нужно настроить параметры безопасности на сервере, чтобы получать запросы.

sudo ufw allow 'Nginx Full'

Готово!