

Т.к. я не уверен, что у меня на виртуальной машине установлен Python, проверяю:

```
user@user-VirtualBox:~$ ~python3 --version
Команда «~python3» не найдена. Возможно, вы имели в виду:
  command 'python3' from deb python3 (3.10.6-1~22.04)
  command 'bpython3' from deb bpython (0.22.1-2)
  command 'ipython3' from deb ipython3 (7.31.1-1)
Try: sudo apt install <deb name>
user@user-VirtualBox:~$
```

Оказывается, что не установлен, значит устанавливаем:

```
user@user-VirtualBox:~$ sudo apt update
[sudo] пароль для user:
Суц:1 http://pl.archive.ubuntu.com/ubuntu jammy InRelease
Суц:2 http://pl.archive.ubuntu.com/ubuntu jammy-updates InRelease
Суц:3 http://security.ubuntu.com/ubuntu jammy-security InRelease
Суц:4 http://pl.archive.ubuntu.com/ubuntu jammy-backports InRelease
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Восстановление пакетов до последних версий.
UbuntuSoftware alBox:~$ sudo apt install python3
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Уже установлен пакет python3 самой новой версии (3.10.6-1~22.04).
python3 помечен как установленный вручную.
Следующий пакет устанавливался автоматически и больше не требуется:
  tcsd
Для его удаления используйте «sudo apt autoremove».
Обновлено 0 пакетов, установлено 0 новых пакетов, для удаления отмечено 0 пакетов, и 0 пакетов не обновлено.
user@user-VirtualBox:~$
```

Далее устанавливаем pip, командой:

```
sudo apt install python3-pip
```

## Далее приступаем к установке Docker

Нам необходимо убедиться, что нет никаких старых версий докера:

```
user@user-VirtualBox:~$ sudo apt remove docker docker-engine docker.io containerd runc
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
E: Невозможно найти пакет docker-engine
user@user-VirtualBox:~$
```

Далее удаляем зависимости

```
user@user-VirtualBox:~$ sudo apt autoremove
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Следующие пакеты будут УДАЛЕНЫ:
  tcsd
Обновлено 0 пакетов, установлено 0 новых пакетов, для удаления отмечено 1 пакетов, и 0 пакетов не обновлено.
После данной операции объем занятого дискового пространства уменьшится на 121 kB.
Хотите продолжить? [Д/н] у
(Чтение базы данных ... на данный момент установлено 214026 файлов и каталогов.)
Удаляется tcsd (7.6.q-31build2) ...
Обрабатываются триггеры для map-db (2.10.2-1) ...
user@user-VirtualBox:~$
```

Далее обновляем список пакетов:

```
sudo apt update
```

И устанавливаем пакеты, которые позволяют использовать репозиторий через HTTPS:

```
sudo apt install apt-transport-https ca-certificates curl software-properties-common
```

```
user@user-VirtualBox:~$ sudo apt update
Сущ:1 http://pl.archive.ubuntu.com/ubuntu jammy InRelease
Сущ:2 http://pl.archive.ubuntu.com/ubuntu jammy-updates InRelease
Пол:3 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Сущ:4 http://pl.archive.ubuntu.com/ubuntu jammy-backports InRelease
Получено 110 kB за 1с (213 kB/s)
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Все пакеты имеют последние версии.
user@user-VirtualBox:~$ sudo apt install apt-transport-https ca-certificates curl software-properties-common
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Уже установлен пакет ca-certificates самой новой версии (20230311ubuntu0.22.04.1).
ca-certificates помечен как установленный вручную.
Уже установлен пакет curl самой новой версии (7.81.0-1ubuntu1.13).
Уже установлен пакет software-properties-common самой новой версии (0.99.22.7).
software-properties-common помечен как установленный вручную.
Следующие НОВЫЕ пакеты будут установлены:
  apt-transport-https
Обновлено 0 пакетов, установлено 1 новых пакетов, для удаления отмечено 0 пакетов, и 0 пакетов не обновлено.
Необходимо скачать 1 510 В архивов.
После данной операции объем занятого дискового пространства возрастёт на 169 kB.
Хотите продолжить? [Д/н] y
Пол:1 http://pl.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 apt-transport-https all 2.4.9 [1 510 В]
Получено 1 510 В за 0с (18,3 kB/s)
Выбор ранее не выбранного пакета apt-transport-https.
(Чтение базы данных ... на данный момент установлено 214015 файлов и каталогов.)
Подготовка к распаковке .../apt-transport-https_2.4.9_all.deb ...
Распаковывается apt-transport-https (2.4.9) ...
Настраивается пакет apt-transport-https (2.4.9) ...
user@user-VirtualBox:~$
```

Добавляем официальный ключ GPG Docker в систему:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg
```

Добавляем репозиторий Docker в список источников пакетов APT:

```
echo "deb [signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null
```

Обновляем список пакетов с добавленным репозиторием Docker:

```
sudo apt update
```

```

user@user-VirtualBox:~$ sudo apt install apt-transport-https ca-certificates curl software-properties-common
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Уже установлен пакет ca-certificates самой новой версии (20230311ubuntu0.22.04.1).
ca-certificates помечен как установленный вручную.
Уже установлен пакет curl самой новой версии (7.81.0-1ubuntu1.13).
Уже установлен пакет software-properties-common самой новой версии (0.99.22.7).
software-properties-common помечен как установленный вручную.
Следующие НОВЫЕ пакеты будут установлены:
  apt-transport-https
Г: 0 пакетов, установлено 1 новых пакетов, для удаления отмечено 0 пакетов, и 0 пакетов не обновлено.
В: Справка, чтобы скачать 1 510 В архивов.
После данной операции объем занятого дискового пространства возрастет на 169 kB.
Хотите продолжить? [Д/Н] y
Пол:1 http://pl.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 apt-transport-https all 2.4.9 [1 510 B]
Получено 1 510 B за 0с (18,3 kB/s)
Выбор ранее не выбранного пакета apt-transport-https.
(Чтение базы данных -- на данный момент установлено 214015 файлов и каталогов.)
Подготовка к распаковке ./apt-transport-https_2.4.9_all.deb ...
Распаковывается apt-transport-https (2.4.9) ...
Настраивается пакет apt-transport-https (2.4.9) ...
user@user-VirtualBox:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
user@user-VirtualBox:~$ echo "deb [signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
user@user-VirtualBox:~$ sudo apt update
Сущ:1 http://pl.archive.ubuntu.com/ubuntu jammy InRelease
Сущ:2 http://security.ubuntu.com/ubuntu jammy-security InRelease
Сущ:3 http://pl.archive.ubuntu.com/ubuntu jammy-updates InRelease
Сущ:4 http://pl.archive.ubuntu.com/ubuntu jammy-backports InRelease
Пол:5 https://download.docker.com/linux/ubuntu jammy InRelease [48,9 kB]
Пол:6 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [21,4 kB]
Получено 70,3 kB за 1с (109 kB/s)
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Все пакеты имеют последние версии.
W: Пропускается получение настроенного файла «stable/binary-i386/Packages», так как репозиторий «https://download.docker.com/linux/ubuntu jammy InRelease» не поддерживает архитектуру «i386»
user@user-VirtualBox:~$

```

Сообщение о том, что “не поддерживает архитектуру “i386” можно проигнорировать.

Убеждаемся, что устанавливается пакет из репозитория Docker, а не пакет из официального репозитория Ubuntu:

```

user@user-VirtualBox:~$ apt-cache policy docker-ce
docker-ce:
  Установлен: (отсутствует)
  Кандидат:  5:24.0.5-1~ubuntu.22.04~jammy
  Таблица версий:
    5:24.0.5-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:24.0.4-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:24.0.3-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:24.0.2-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:24.0.1-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:24.0.0-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:23.0.6-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:23.0.5-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:23.0.4-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:23.0.3-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:23.0.2-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:23.0.1-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:23.0.0-1~ubuntu.22.04~jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages
    5:20.10.24~3-0~ubuntu-jammy 500
      500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages

```

Устанавливаем Docker:  
 sudo apt install docker-ce

Проверяем, что установлен и запущен:

`sudo systemctl status docker`

```
user@user-VirtualBox:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2023-08-08 16:36:12 CEST; 31s ago
 TriggeredBy: ● docker.socket
    Docs: https://docs.docker.com
   Main PID: 5922 (dockerd)
     Tasks: 10
    Memory: 27.1M
       CPU: 406ms
    CGroup: /system.slice/docker.service
            └─5922 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

sie 08 16:36:11 user-VirtualBox systemd[1]: Starting Docker Application Container Engine...
sie 08 16:36:11 user-VirtualBox dockerd[5922]: time="2023-08-08T16:36:11.558492989+02:00" level=info
sie 08 16:36:11 user-VirtualBox dockerd[5922]: time="2023-08-08T16:36:11.560639494+02:00" level=info
sie 08 16:36:11 user-VirtualBox dockerd[5922]: time="2023-08-08T16:36:11.682498998+02:00" level=info
sie 08 16:36:11 user-VirtualBox dockerd[5922]: time="2023-08-08T16:36:11.964590520+02:00" level=info
sie 08 16:36:12 user-VirtualBox dockerd[5922]: time="2023-08-08T16:36:12.026409968+02:00" level=info
sie 08 16:36:12 user-VirtualBox dockerd[5922]: time="2023-08-08T16:36:12.026543779+02:00" level=info
sie 08 16:36:12 user-VirtualBox dockerd[5922]: time="2023-08-08T16:36:12.084098156+02:00" level=info
sie 08 16:36:12 user-VirtualBox systemd[1]: Started Docker Application Container Engine.
user@user-VirtualBox:~$
```

Чтобы использовать Docker без sudo, добавляем пользователя в группу Docker (просто для удобства):

`sudo usermod -aG docker $USER`

Перезагружаем систему и проверяем работу:

`sudo systemctl status docker`

```
user@user-VirtualBox:~$ sudo systemctl status docker
[sudo] пароль для user:
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2023-08-08 16:40:53 CEST; 44s ago
 TriggeredBy: ● docker.socket
    Docs: https://docs.docker.com
   Main PID: 902 (dockerd)
     Tasks: 10
    Memory: 98.5M
       CPU: 1.055s
    CGroup: /system.slice/docker.service
            └─902 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

sie 08 16:40:52 user-VirtualBox dockerd[902]: time="2023-08-08T16:40:52.1610307>
sie 08 16:40:52 user-VirtualBox dockerd[902]: time="2023-08-08T16:40:52.1703882>
sie 08 16:40:52 user-VirtualBox dockerd[902]: time="2023-08-08T16:40:52.3532091>
sie 08 16:40:52 user-VirtualBox dockerd[902]: time="2023-08-08T16:40:52.3652189>
sie 08 16:40:52 user-VirtualBox dockerd[902]: time="2023-08-08T16:40:52.9198039>
sie 08 16:40:53 user-VirtualBox dockerd[902]: time="2023-08-08T16:40:53.0455002>
sie 08 16:40:53 user-VirtualBox dockerd[902]: time="2023-08-08T16:40:53.3127119>
sie 08 16:40:53 user-VirtualBox dockerd[902]: time="2023-08-08T16:40:53.3134919>
sie 08 16:40:53 user-VirtualBox dockerd[902]: time="2023-08-08T16:40:53.5850948>
sie 08 16:40:53 user-VirtualBox systemd[1]: Started Docker Application Containe>
lines 1-22/22 (END)...skipping...
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2023-08-08 16:40:53 CEST; 44s ago
 TriggeredBy: ● docker.socket
    Docs: https://docs.docker.com
   Main PID: 902 (dockerd)
     Tasks: 10
    Memory: 98.5M
       CPU: 1.055s
    CGroup: /system.slice/docker.service
            └─902 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
```

Устанавливаем Django:

pip3 install Django и sudo apt install python3-django

```
user@user-VirtualBox:~$ pip3 install Django
Defaulting to user installation because normal site-packages is not writeable
Collecting Django
  Downloading Django-4.2.4-py3-none-any.whl (8.0 MB)
    8.0/8.0 MB 12.2 MB/s eta 0:00:00
Collecting sqlparse>=0.3.1
  Downloading sqlparse-0.4.4-py3-none-any.whl (41 kB)
    41.2/41.2 KB 6.0 MB/s eta 0:00:00
Collecting asgiref<4,>=3.6.0
  Downloading asgiref-3.7.2-py3-none-any.whl (24 kB)
Collecting typing-extensions>=4
  Downloading typing_extensions-4.7.1-py3-none-any.whl (33 kB)
Installing collected packages: typing-extensions, sqlparse, asgiref, Django
WARNING: The script sqlformat is installed in '/home/user/.local/bin' which is not on PATH.
Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
WARNING: The script django-admin is installed in '/home/user/.local/bin' which is not on PATH.
Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed Django-4.2.4 asgiref-3.7.2 sqlparse-0.4.4 typing-extensions-4.7.1
Ubuntu Software alBox:~$
```

Создаем новый проект Django с именем "myproject" и переходим в директорию:

django-admin startproject myproject

cd myproject

Создаем новое Django-приложение с именем "helloworld":

python3 manage.py startapp helloworld

Открываем файл "views.py" внутри папки приложения "helloworld" и определяем простой view для вывода "Hello, World!" на главной странице:

nano helloworld/views.py

Изменяем содержимое файла:

```
GNU nano 6.2
from django.http import HttpResponse

def hello(request):
    return HttpResponse("Hello, World!")
```

Регистрируем созданный view, добавляем URL-маршрут в файле "urls.py" внутри приложения "helloworld":

nano helloworld/urls.py

```
GNU nano 6.2 helloworld/urls.py *
from django.urls import path
from . import views

urlpatterns = [
    path('', views.hello, name='hello'),
]
```

Включаем приложение "helloworld", добавив его имя в "INSTALLED\_APPS" в файле "settings.py" внутри проекта:  
nano myproject/settings.py

```
# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'helloworld',
]
```

Создаем Dockerfile для приложения Django:  
nano Dockerfile

```
GNU nano 6.2 Dockerfile *
FROM python:3.9 Используем образ Python в качестве базового образа

ENV DJANGO_SETTINGS_MODULE=myproject.settings Устанавливаем переменное окружение для запуска в режиме production

WORKDIR /usr/src/app Устанавливаем рабочую директорию внутри контейнера

COPY requirements.txt . Копируем файлы requirements.txt и

RUN pip install --no-cache-dir -r requirements.txt устанавливаем зависимости

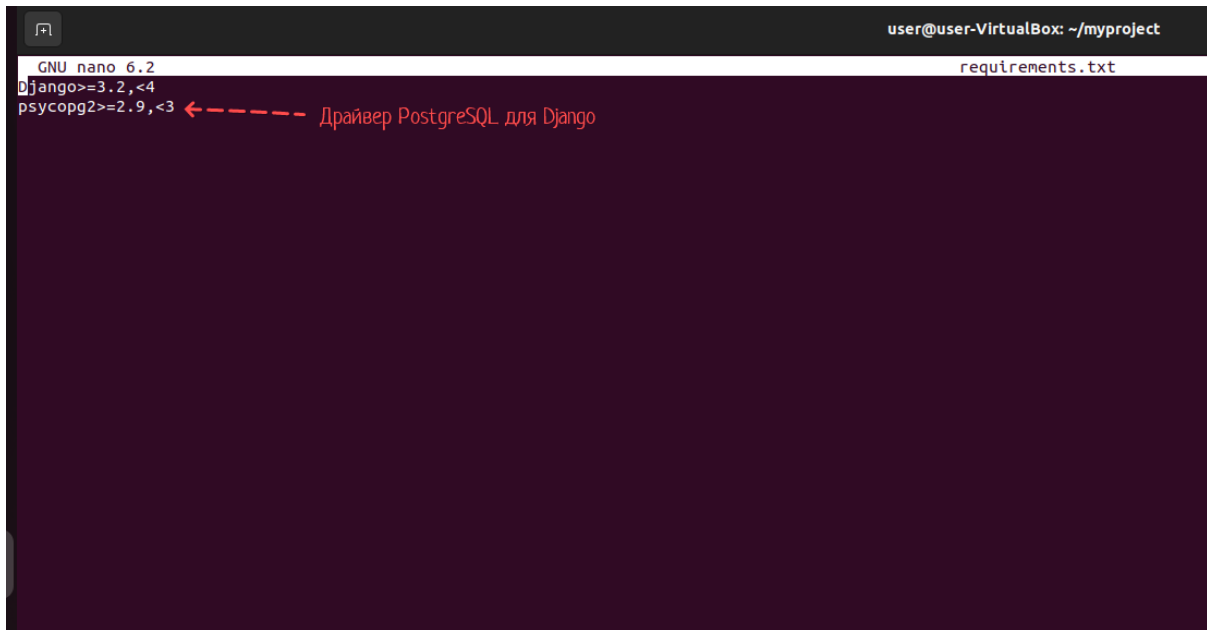
COPY . . Копируем исходный код нашего проекта в контейнер

RUN python manage.py collectstatic --noinput Собираем статистические файлы Django
```

Создаем новый файл с именем "requirements.txt" в корневой папке вашего проекта:  
nano requirements.txt



Добавляем в него зависимости:



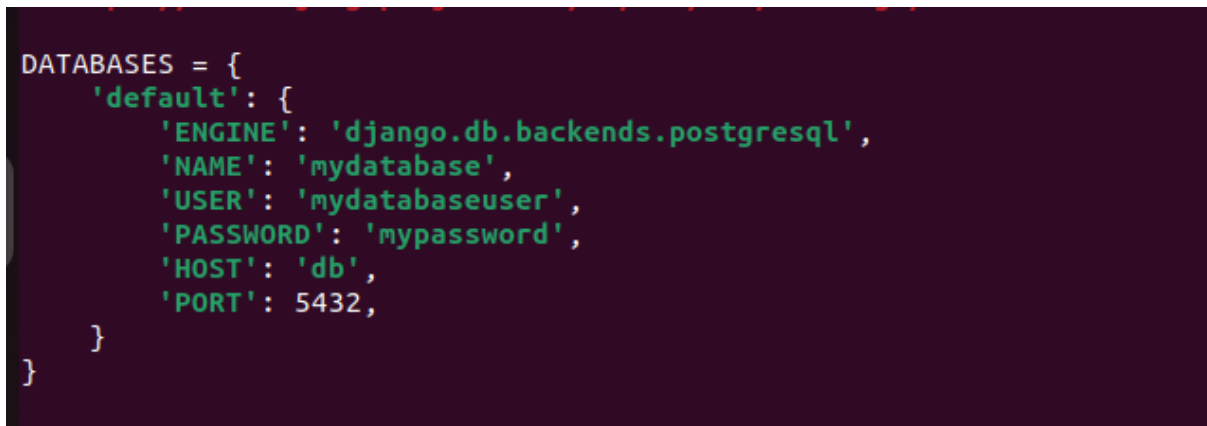
```
GNU nano 6.2 requirements.txt
Django>=3.2,<4
psycopg2>=2.9,<3 ← ----- Драйвер PostgreSQL для Django
```

Далее необходимо провести настройку для соединения с базой данных PostgreSQL:

Открываем файл с настройками:

```
nano myproject/settings.py
```

Ищем DATABASES и редактируем:



```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'mydatabase',
        'USER': 'mydatabaseuser',
        'PASSWORD': 'mypassword',
        'HOST': 'db',
        'PORT': 5432,
    }
}
```

Создаем docker-compose.yml для определения контейнеров:  
nano docker-compose.yml

```
GNU nano 6.2 docker-compose.yml *
version: '3'

services:
  web:
    build:
      context: .
      dockerfile: Dockerfile
    command: python manage.py runserver 0.0.0.0:8000
    volumes:
      - ./usr/src/app
    ports:
      - "8000:8000"
    depends_on:
      - db

  db:
    image: postgres:latest
    environment:
      POSTGRES_DB: mydatabase
      POSTGRES_USER: mydatabaseuser
      POSTGRES_PASSWORD: mypassword
    volumes:
      - postgres_data:/var/lib/postgresql/data

  nginx:
    image: nginx:latest
    volumes:
      - ./nginx/etc/nginx/conf.d
    ports:
      - "80:80"
    depends_on:
      - web

volumes:
  postgres_data:
```

Создаем папку с именем "nginx" в корневой папке проекта и в ней создаем файл "default.conf" с настройками для Nginx:

mkdir nginx

nano nginx/default.conf

```
GNU nano 6.2 nginx/default.conf *
server {
    listen 80;
    server_name localhost;

    location / {
        proxy_pass http://web:8000;
    }

    location /static/ {
        alias /usr/src/app/static/;
    }
}
```

Далее открываем файл "settings.py" нашего проекта Django и добавляем настройки для раздачи статических файлов:

nano myproject/settings.py

```
# https://docs.djangoproject.com/en/4.2/howto/static-files/

STATIC_URL = 'static/'
STATIC_ROOT = os.path.join(BASE_DIR, 'static')
```



## Настройка CI/CD с использованием Github Actions

Создаем новую папку ".github/workflows" в корневой папке вашего проекта:

```
mkdir -p .github/workflows
```

Создаем файл "main.yml" в папке ".github/workflows":

```
nano .github/workflows/main.yml
```

Настраиваем автоматическую сборку на сервере и развертывание проекта:

```
GNU nano 6.2 .github/workflows/main.yml *
name: Django CI/CD

on:
  push:
    branches:
      - main

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout repository
        uses: actions/checkout@v2

      - name: Set up Python
        uses: actions/setup-python@v2
        with:
          python-version: '3.x'

      - name: Install dependencies
        run: pip install -r requirements.txt

      - name: Run tests
        run: python manage.py test

      - name: Build and push Docker image
        env:
          DOCKER_USERNAME: ${ secrets.DOCKER_USERNAME }
          DOCKER_PASSWORD: ${ secrets.DOCKER_PASSWORD }
        run: |
          echo $DOCKER_PASSWORD | docker login -u $DOCKER_USERNAME --password-stdin
          docker build -t my_django_app .
          docker tag my_django_app $DOCKER_USERNAME/my_django_app:latest
          docker push $DOCKER_USERNAME/my_django_app:latest
```

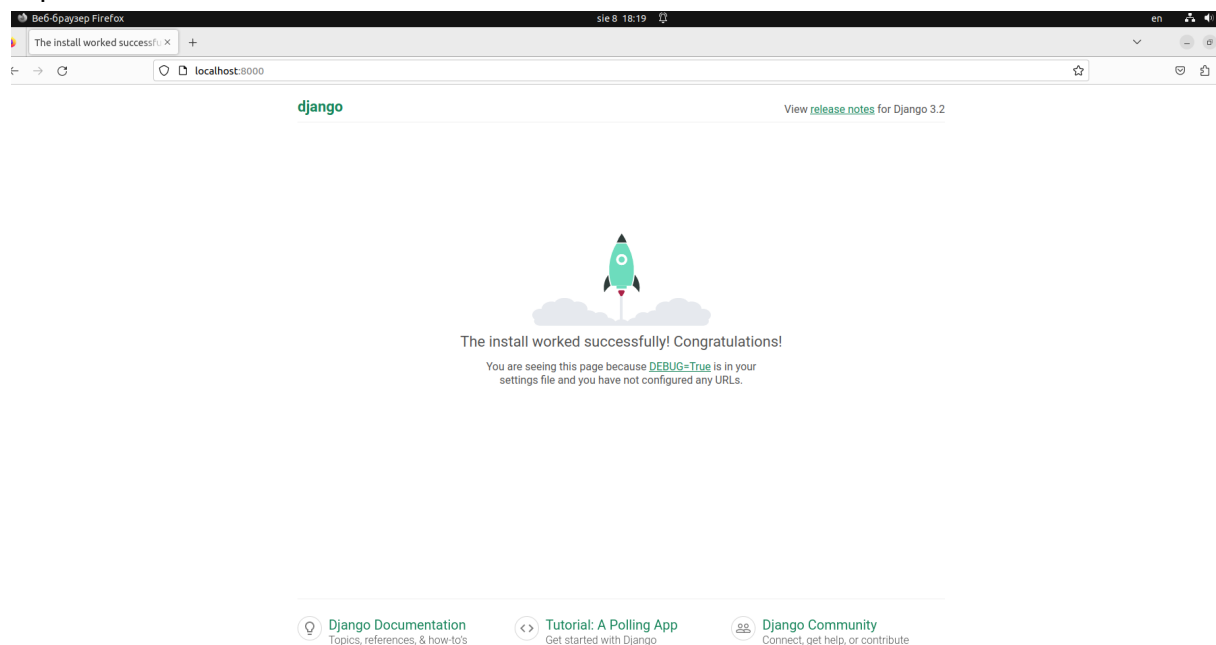
Проверяем контейнеры с помощью Docker Compose

Перед скриншотом проверял, поэтому сперва останавливаю и заново запускаю контейнеры

```
user@user-VirtualBox:~/myproject/myproject$ docker-compose down
Stopping myproject_nginx_1 ... done
Stopping myproject_web_1    ... done
Stopping myproject_db_1     ... done
Removing myproject_nginx_1  ... done
Removing myproject_web_1    ... done
Removing myproject_db_1     ... done
Removing network myproject_default
user@user-VirtualBox:~/myproject/myproject$ docker-compose up -d
Creating network "myproject_default" with the default driver
Creating myproject_db_1 ... done
Creating myproject_web_1 ... done
Creating myproject_nginx_1 ... done
user@user-VirtualBox:~/myproject/myproject$ docker-compose logs web
Attaching to myproject_web_1
web_1      | Watching for file changes with StatReloader
user@user-VirtualBox:~/myproject/myproject$
```

Теперь проверим, работает ли ваше Django приложение корректно:  
docker-compose logs web (Скриншот выше)

Далее проверяем работает ли Django приложение корректно, открыв веб-браузер и перейдя по адресу:  
<http://localhost:8000/>



Далее выполняем миграцию базы данных и создаем суперпользователя Django. Миграции применяют все изменения моделей к базе данных, а суперпользователь позволяет нам получить доступ к административной панели Django

Выполняем миграцию

`docker-compose exec web python manage.py migrate`

```
user@user-VirtualBox:~/myproject/myproject$ docker-compose exec web python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK
user@user-VirtualBox:~/myproject/myproject$
```

Создаем суперпользователя и следуем инструкциям для ввода логина, email и пароля:

`docker-compose exec web python manage.py createsuperuser`

```
user@user-VirtualBox:~/myproject/myproject$ docker-compose exec web python manage.py createsuperuser
Username (leave blank to use 'root'): Mikita
Email address: nikita.timohin2017@gmail.com
Password:
Password (again):
Superuser created successfully.
user@user-VirtualBox:~/myproject/myproject$
```

Заходим и проверяем

