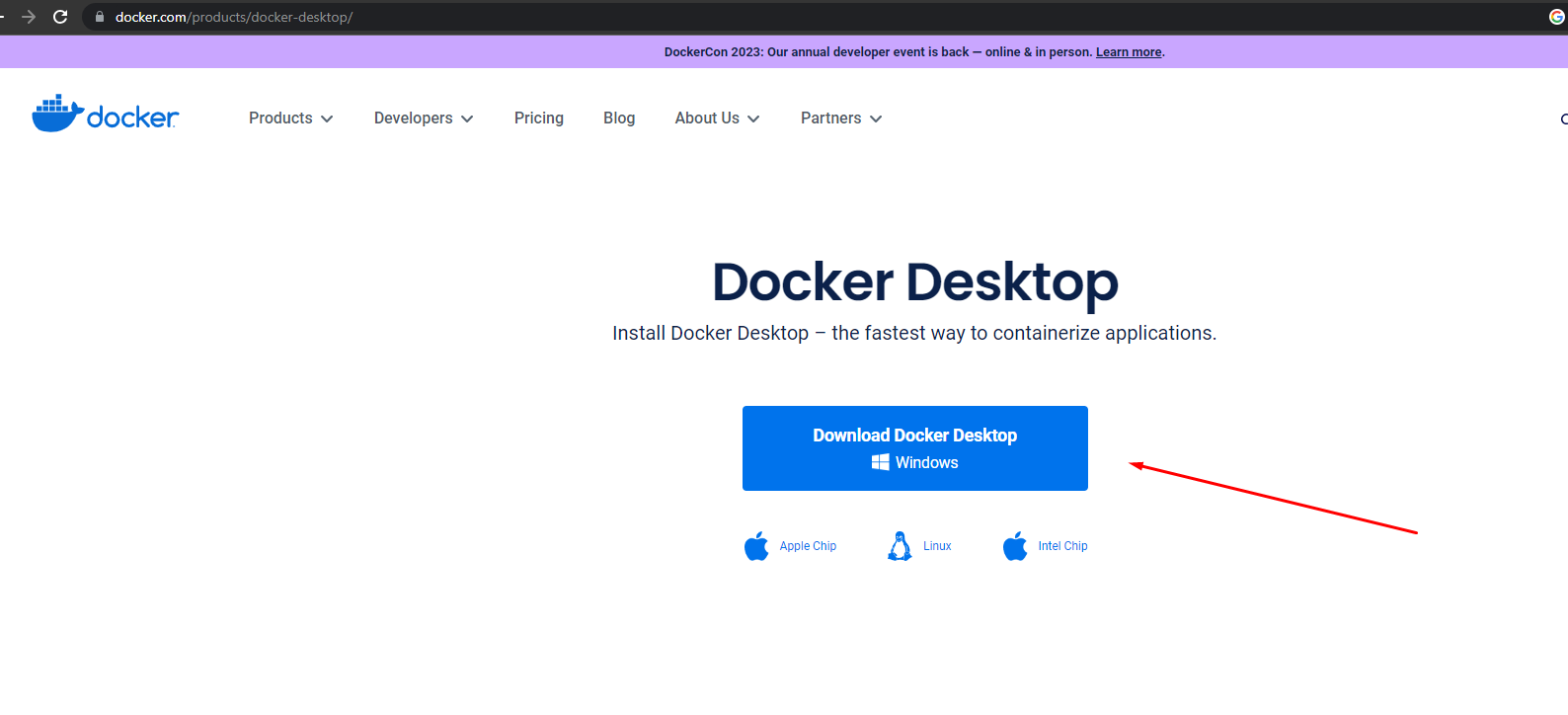
**Оптимизация Django. 1 - Django + Docker**

Docer ni o’rnatamiz.

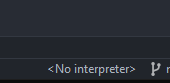
Brauzerga docer descktop deb yozamiz va



Shu sahifaga kirib docerni skachat qilamiz.

Docerni o’rnatib oldik

Endi pycharmga kirib, No interpretatorni tanlab qo’yamiz. Biz buni docerda ishlatamiz.

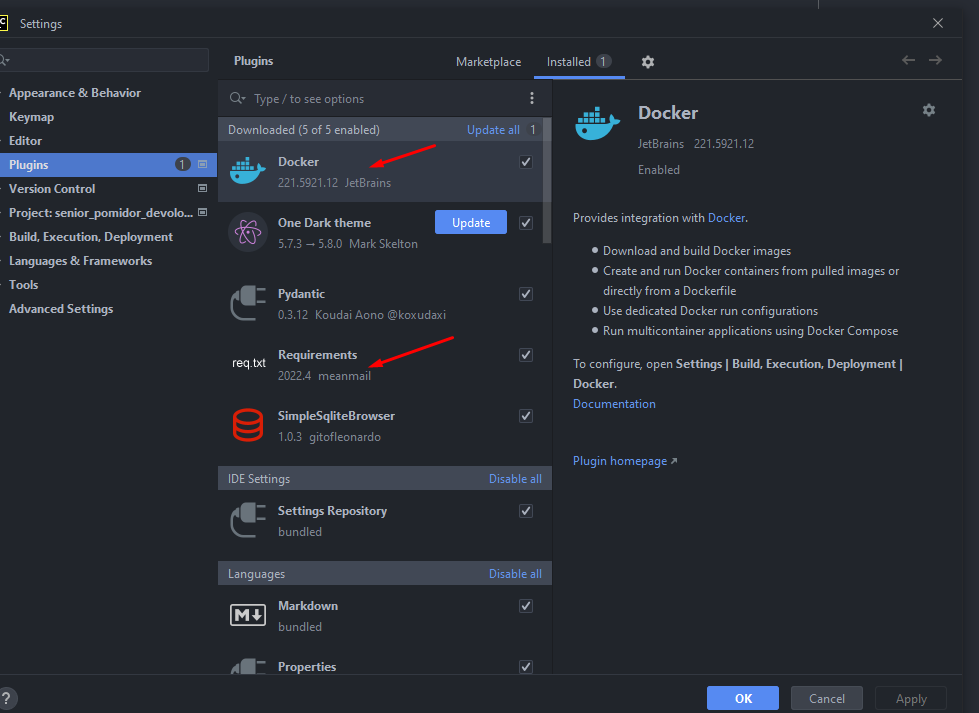


Endi requirements.txt fayl ochvolamiz va ichiga quyidagilarni yozamiz.

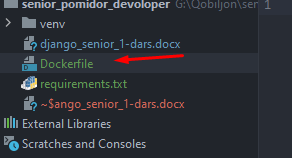
Django==3.2.16  
djangorestframework==3.14.0

Biz shu versiyalardan foydalanamiz.

Va plugins ga o’tib bu ikkita plaginni o’rnatamiz.



Endi Dockerfile qo’shamiz.



Dockerfile da biz loyixamizning asosiy image ni qilamiz. Image bu obraz, obraz bu .iso formatidagi faylga o’xshagan narsa bo’lsa kerak

Endi Dockerfile ni ichiga quyidagilarni yozamiz.

FROM python:3.9-alpine3.16  
  
COPY requirements.txt /temp/requirements.txt

Bu yerda alpine bu linux ni versiyasi buni menimcha docker hub dan ko’rishimiz mumkin.

Endi service nomli papka ochamiz va bu papkani Ichida bizni django loyixamiz joylashadi.

Dockerfile da yozganlarimiz

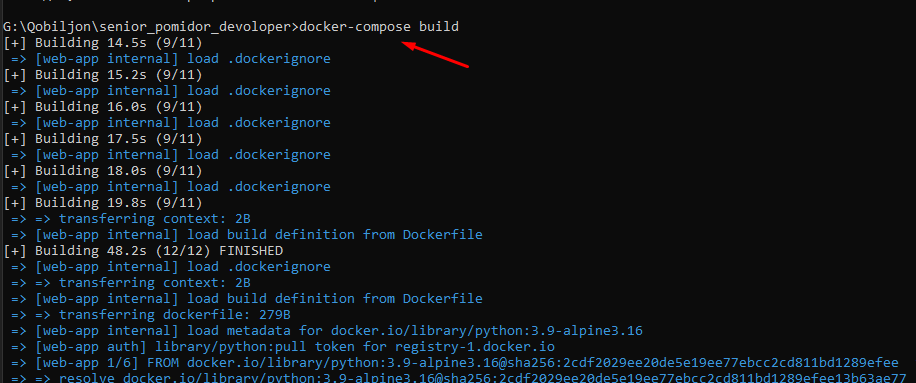
FROM python:3.9-alpine3.16  
  
COPY requirements.txt /temp/requirements.txt  
COPY service /service  
WORKDIR /service  
EXPOSE 8000  
  
RUN pip install -r /temp/requirements.txt  
RUN adduser --disabled-password service-user  
  
USER service-user

Bu yerda --disabled-password service-user ni vazifasi windows foydalanuvchisini hosil qilish. Va USER service-user deb user ni hosil qilamiz.

Endi docer obraz hosil qilishi uchun docker-compose.yml formatida fayl hosil qilamiz. Va bu faylga quyidagilarni yozamiz.

services:  
 web-app:  
 build:  
 context: .  
 ports:  
 - "8000:8000"  
 volumes:  
 - ./service:/service  
  
 command: >  
 sh -c "python manage.py runserver 0.0.0.0:8000"

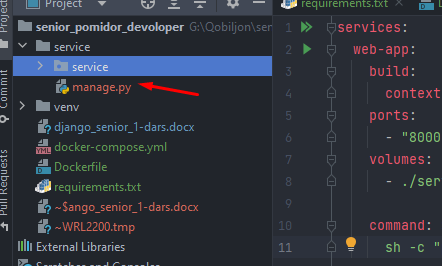
endi docker ga o’tib uni ishlatamiz. So’ng fayllarimiz joylashgan joyni cmd bilan ochvolamiz. So’ng docker-compose build deb yozamiz va bizga kerakli bo’lgan kutubxonalarni o’rnatishni boshledi



Endi loyixani ishlatish uchun docker-compose up deb yozamiz. Lekin bizda to’xtab qoladi. Sababi manage.py fayli yaratilmagan. Uni yasash uchun cmd ga yozamiz.

docker-compose run --rm web-app sh -c "django-admin startproject service ."

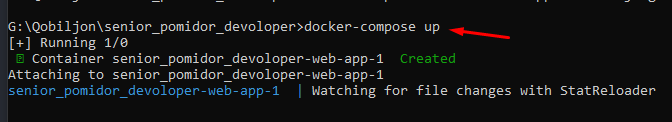
Endi pycharmga o’tib ko’radigan bo’lsak



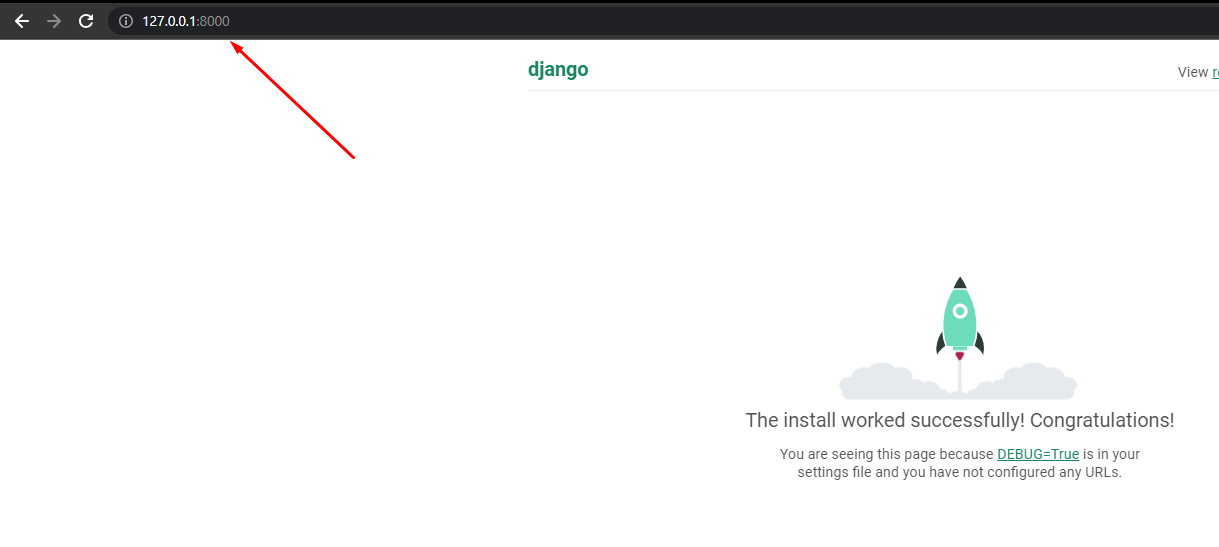
Mana loyixa yaratildi.

Endi bu loyixani ishga tushiramiz. Buning uchun terminalga yozamiz.

docker-compose up



Mana loyixa ishga tushdi dedi. Endi brauzerga o’tib tekshirib ko’ramiz.



Mana loyixa ishlamoqda.

Endi loyixaga postgres ni ulaymiz. Buning uchun docker-compose.yml faylimizga qo’shimchalar kiritamiz.

services:  
 web-app:  
 build:  
 context: .  
 ports:  
 - "8000:8000"  
 volumes:  
 - ./service:/service  
 environment:  
 - DB\_HOST=database  
 - DB\_NAME=dbname  
 - DB\_USER=dbuser  
 - DB\_PASS=pass  
  
 command: >  
 sh -c "python manage.py runserver 0.0.0.0:8000"  
  
 depends\_on:  
 - database  
  
 database:  
 image: postgres:14.6-alpine  
 environment:  
 - POSTGRES\_DB=dbname  
 - POSTGRES\_USER=dbuser  
 - POSTGRES\_PASSWORD=pass

Bu yerda environment depends\_on database larni ichidagi ma’lumotlarni bilan qo’shib oldik.

Endi service/settings.py ga o’tmiz va bu joyni topamiz.

DATABASES = {  
 'default': {  
 'ENGINE': 'django.db.backends.sqlite3',  
 'NAME': BASE\_DIR / 'db.sqlite3',  
 }  
}

Bu yerda database sqlite a yozilgan. Biz buni postgresga sozlaymiz.

DATABASES = {  
 'default': {  
 'ENGINE': 'django.db.backends.postgresql',  
 'HOST': os.environ.get("DB\_HOST"),  
 'NAME': os.environ.get("DB\_NAME"),  
 'USER': os.environ.get("DB\_USER"),  
 'PASSWORD': os.environ.get("DB\_PASS"),  
 }  
}

Mana postgresni uladik

Biz loyixaga o’zgartirish kiritdik. Shuning uchun cmd ga o’tamiz va

docker-compose build

qilamiz. Va requirements.txt ni qayta ishlatadi. Endi

docker-compose up

qilib loyixani ishlatamiz.

Endi ishlatadigan bo’lsak, bizda hatolik beradi. Sababi psycopg2 ni topolmadi.

Buni requirements.txt ni ichiga yozamiz.

*Django*==3.2.16  
*djangorestframework*==3.14.0  
*psycopg2*==2.8.6

Endi Docerfile ga o’tib, qo’shimcha qo’shamiz.

FROM python:3.9-alpine3.16  
  
COPY requirements.txt /temp/requirements.txt  
COPY service /service  
WORKDIR /service  
EXPOSE 8000  
  
RUN apk add postgresql-client build-base postgresql-dev  
  
RUN pip install -r /temp/requirements.txt  
  
RUN adduser --disabled-password service-user  
  
USER service-user

Bu yerda

RUN apk add postgresql-client build-base postgresql-dev

Linux to’g’ri ishlashi uchun 3 ta qo’shimcha paket o’rnatishini aytdik.

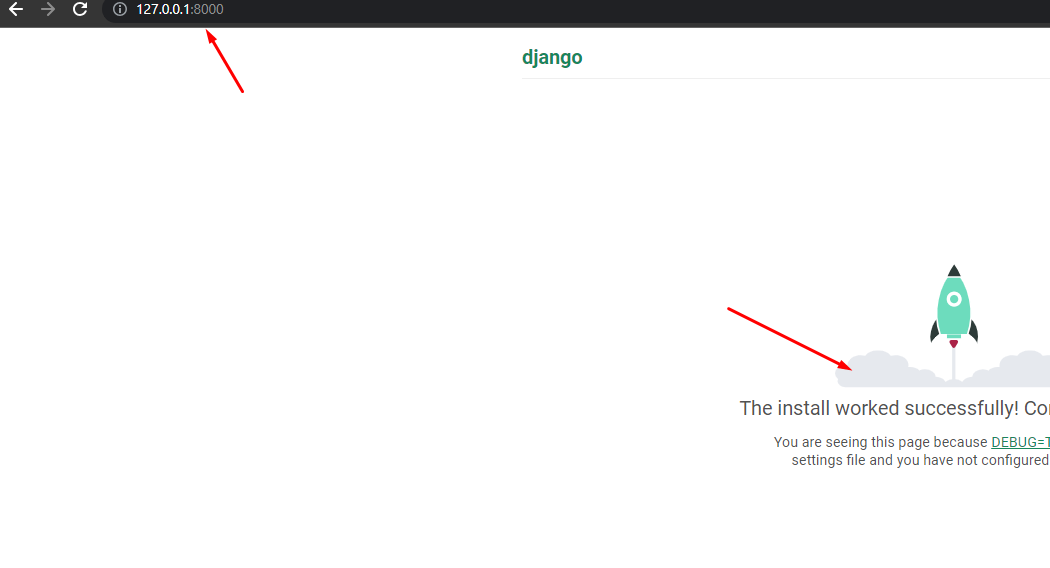
Endi cmd ga o’tamiz.

docker-compose build

qilamiz. Sababi requirements.txt va Dockerfile ga qo’shimcha paket qo’shdik

Endi loyixani ishlatamiz.

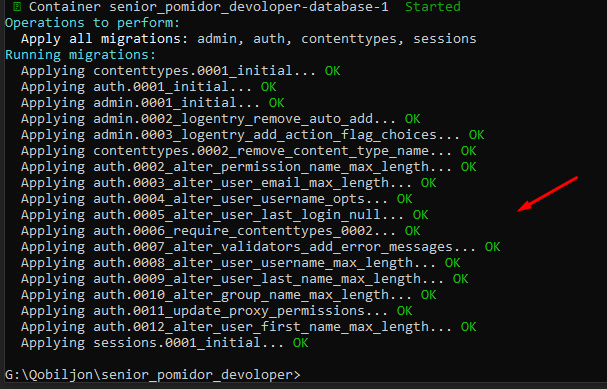
docker-compose up



Loyixa ishladi.

Endi admin panelga kirmoqchi bo’lsak, kirolmaymiz. Sababi bizda migratsiya qo’lllanilmagan. Shuning uchun cmdga o’tib migrate qilamiz.

docker-compose run --rm web-app sh -c "python manage.py migrate"

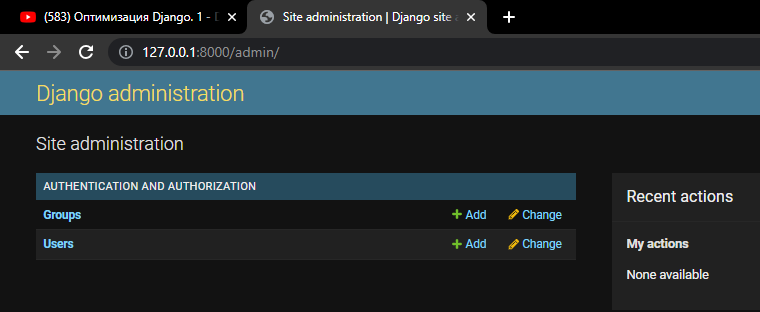


Migratsiya qo’llanildi.

Endi super user xosil qilishimiz kerak

docker-compose run --rm web-app sh -c "python manage.py createsuperuser"

endi admin panelga kirib ko’ramiz.



Mana admin panelga kirdi.

Bu dars tugadi)