

Chat App – Faza 3 - LINK GIT: <https://github.com/andu-diaconu/aso-3>

Diaconu Andrei

Obiectivul acestei faze cu numarul 3 din proiectul mare de Chat app prevede deploy-ul pe Docker. Toate restul operatiilor care vor fi aplicate se bazeaza pe ideea ca proiectul este deja functional si mai trebuie doar postat pe piata

Baza de date dbsql este stearsa si inlocuita cu una de tip Postgres. Pentru aceasta este creat un volum in dockerfile si docker-compose, mentionez ca va rula intr-un container separat. Aplicatia va fi pe 2 porturi in functie de modul de dezvoltare (development – 8000) (production - 1337)

Mod de rezolvare

M-am asigurat ca versiunile folosite pentru python, django, rest_framework sunt potrivite si am creat un fisier de requirments.txt ca vi rulat la fiecare comanda de docker build. Am instalat docker si creat fisiere compatibile cu acesta

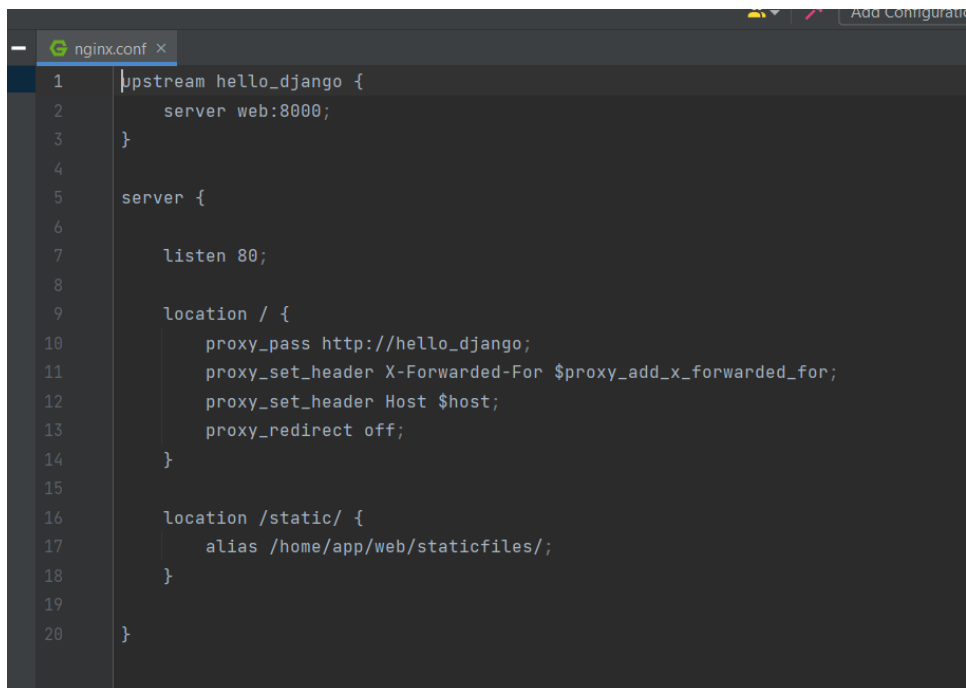
Initial fiserele contineau informatie putina cat sa ruleze in development adica acelasi lucru ca si cand am rula comanda python manage.py runserver. Odata trecut cu succes acest pas am aduagat informatii suplimentare cum ar fi baze de date, servere proxy, fisiere statice si fisiere media

Setarile le-am modificat sa foloseasca Postgres dupa ce am sters dbsql care juca pana acum rolul de baza de date. In docker-compose.yml sunt cuprinse atat imaginea pentru baza de date cat si imaginea pentru aplicatia propriu zisa (chat).

Tot legat de setari trebuie sa mentionez ca am sters hardocadrile si acum folosesc fisiere .env separate pentru fiecare mediu al aplicatie (productie / development).

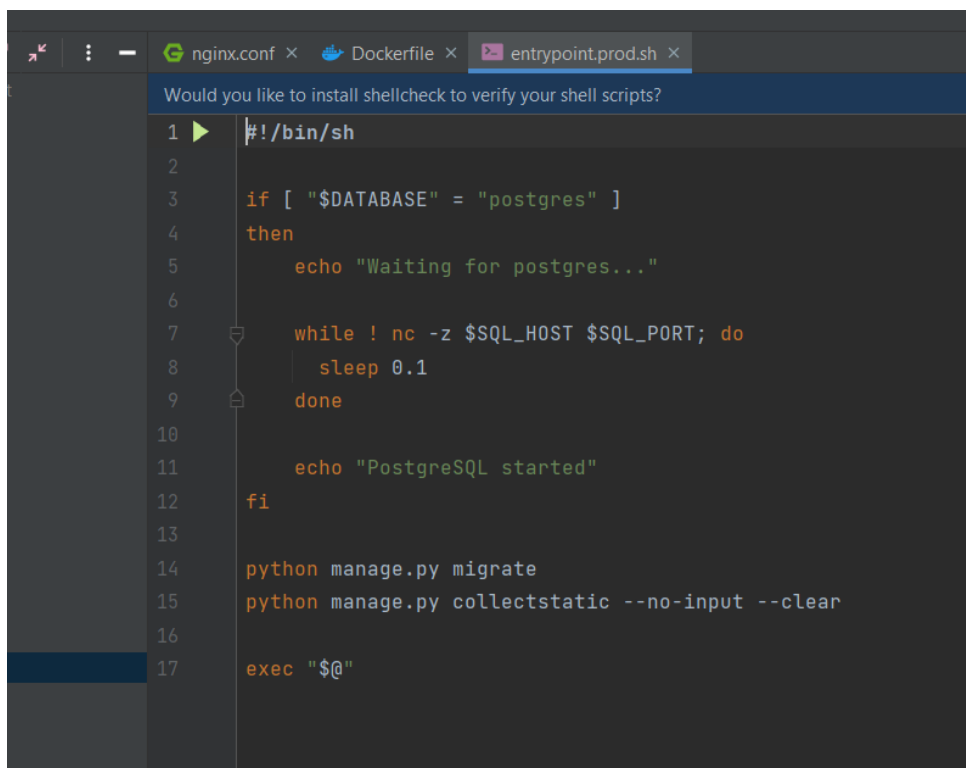
Daca pana acum foloseam serverul implicit generat de django acum folosim WSGI care este un server real provenit de la gunicorn. Fiserele mele statice (css – tailwind) sunt prelucrate si trimise prin intermediu nginx care intercepteaza toate requesturile – reverse proxy.

Pentru productie am un alt dockerfile cu extensia .prod . Le diferentiem pentru ca acolo in loc de ports avem expose pentru ca aplicata va rula la portul folosit de nginx nu cel 8000. Cel 8000 va fi functional si utilizat in continuare dar doar de ngx. Aplicatia vizibila pentru toti userii se vede numai si numai la 1337.



The image shows a code editor with a single tab titled 'nginx.conf'. The file contains an Nginx configuration. It starts with an 'upstream' block for 'hello_django' pointing to 'web:8000'. Then, a 'server' block is defined, listening on port 80. It has two 'location' blocks: one for the root path '/' which proxies requests to the 'hello_django' upstream and sets headers, and another for '/static/' which serves files from '/home/app/web/staticfiles/'.

```
1 upstream hello_django {
2     server web:8000;
3 }
4
5 server {
6
7     listen 80;
8
9     location / {
10        proxy_pass http://hello_django;
11        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
12        proxy_set_header Host $host;
13        proxy_redirect off;
14    }
15
16    location /static/ {
17        alias /home/app/web/staticfiles/;
18    }
19
20 }
```



The image shows a code editor with three tabs: 'nginx.conf', 'Dockerfile', and 'entrypoint.prod.sh'. The 'entrypoint.prod.sh' tab is active, showing a shell script. A notification banner at the top asks 'Would you like to install shellcheck to verify your shell scripts?'. The script starts with a shebang, checks if the database is 'postgres', and if so, it enters a loop waiting for the database to be ready. After the database is ready, it runs 'python manage.py migrate' and 'python manage.py collectstatic --no-input --clear', and finally executes the command passed to the container.

```
1 #!/bin/sh
2
3 if [ "$DATABASE" = "postgres" ]
4 then
5     echo "Waiting for postgres..."
6
7     while ! nc -z $SQL_HOST $SQL_PORT; do
8         sleep 0.1
9     done
10
11     echo "PostgreSQL started"
12 fi
13
14 python manage.py migrate
15 python manage.py collectstatic --no-input --clear
16
17 exec "$@"
```

Am atasat poze cu modul de functionare a nginx si a prelucrarii static fileurilor.

Probleme intampinate

De-a lungul acestui tutorial pe care l-am parcurs nu am intampinat probleme pana la partea de static files. Acolo inca persista o problema si anume in pagina de login nu am css-ul corespunzator. Nu pot sa-mi dau seama de ce , insa acesata este prezenta si pentru admin unde ne logam cu superuser-ul

De asemenea a trebuit sa introduc allowed origins ca sa ma pot conecta cu superuser-ul pentru a crea pagina principala, daca nu adaugam primeam eroare de securitate.

Concluzii

Ma bucur foarte mult ca am avut acest tutorial sa inteleg exact ce se intampla in spatele unui deploy si mai ales ca functioneaza pe baza a medii. Development care poate fi in continuare folosit pentru a aduce update-uri aplicatiei si Productia care o putem socoti ca aplicatia de pe piata utilizata direct de userii nostri.