Περιεχόμενα

Εισαγωγή

Σκοπός του συγκεκριμένου εργαστηριακού project, ήταν η δημιουργία ενός συνόλου από containers, που θα λειτουργούν με τη χρήση του docker compose, στοχεύοντας στο να παρέχουν στον χρήστη έναν εύκολο τρόπο να παρουσιάσει την προσωπικότητα του και να μοιραστεί τα ενδιαφέροντα και τις γνώσεις του online, καθώς και να περιηγηθεί στο διαδίκτυο διαφυλάσσοντας σε μεγάλο βαθμό, την ανωνυμία και τα προσωπικά του δεδομένα. Για να επιτευχθεί αυτό, αξιοποιήθηκαν αρκετές υπηρεσίες, κάθε μια από τις οποίες επιτελεί διαφορετικό σκοπό και τρέχει σε ένα ξεχωριστό container.

Δημιουργία Zola container

Το Zola, πρόκειται για έναν απλό static website generator (SSG), γραμμένο σε Rust, που χρησιμοποιώντας το Tera template engine και το pulldown-cmark, μπορεί να παράγει αυτόματα μια ιστοσελίδα blog, μέσα από περιεχόμενο γραμμένο σε CommonMark, γεγονός που καθιστά την όλη διαδικασία ιδιαίτερα εύκολη και γρήγορη. Παρακάτω φαίνεται το dockerfile που χρησιμοποιείται για το build του συγκεκριμένου container.

```
# Using the official Arch as the base image
FROM archlinux:latest

# Updating packages
RUN pacman -Syu --noconfirm

# Installing Zola from the repository
RUN pacman -S zola --noconfirm

# Setting myblog as the working directory
WORKDIR /myblog
```

Στο docker-compose.yml αρχείο, δηλώνεται ότι το build του συγκεκριμένου container θα γίνεται μέσω του dockerfile και δημιουργείται ένα volume, ώστε το zola να έχει πρόσβαση στα απαραίτητα templates και αρχεία για να παράγει την τελική ιστοσελίδα. Αυτή αποθηκεύεται στον φάκελο public. Η δημιουργία της σελίδας γίνεται μέσω της εντολής "zola build".

```
services:
```

```
  zola:
    build:
      context: ./
      dockerfile: dockerfile
    working_dir: /myblog
    command: sh -c "zola build"
    volumes:
      - ./myblog:/myblog/
    networks:
      - my-network
```

Δημιουργία yt-local container

Το youtube local, είναι ένα front end του youtube που παρέχει στον χρήστη τη δυνατότητα να περιηγηθεί στην πλατφόρμα ανώνυμα, χρησιμοποιώντας το tor για τη δρομολόγηση των αιτημάτων του και χωρίς τη διαμεσολάβηση του youtube API. Το build του συγκεκριμένου container γίνεται μέσω του αρχείου yt.dockerfile. Συγκεκριμένα, αφού πρώτα γίνει η εγκατάσταση του alpine και ορισμένων dependencies, κάνουμε clone το repository της εφαρμογής και στη συνέχεια, τροποποιούμε το αρχείο server.py ώστε να δέχεται αιτήματα από όλα τα interfaces και όχι μόνο το loopback

```
FROM alpine:latest

RUN apk update && apk add --no-cache git python3 py3-pip
RUN pip3 install gevent
RUN pip3 install flask
RUN git clone https://github.com/user234683/youtube-local.git
WORKDIR /youtube-local
RUN pip3 install -r requirements.txt
RUN sed -i 's/127\.0\.0\.1/0.0.0.0/g' server.py
```

Όπως φαίνεται και στο αρχείο docker-compose.yml, για το container yt-local αντιστοιχούμε την την πόρτα 8080 του, με αυτή του host (καθώς σε αυτήν ακούει ο yt-local server) και στην συνέχεια το τοποθετούμε στο ίδιο δίκτυο με τα υπόλοιπα.

```
  yt-local:
    build:
      context: ./
      dockerfile: yt.dockerfile
    working_dir: /youtube-local
    command: sh -c "python3 server.py"
    ports:
      - "0.0.0.0:8080:8080"
    networks:
      - my-network
```

Δημιουργία SearXNG container

Το SearXNG είναι ένα fork του searx που παρέχει ένα αρκετά βελτιωμένο UI και είναι σημαντικά ευκολότερο στην εγκατάσταση του καθώς δε βασίζεται στο Morty και το Filtron. Αποτελεί ένα metasearch engine γεγονός που διασφαλίζει τα προσωπικά δεδομένα του χρήστη. Μια διαφορά σε σχέση με το Searx είναι πως αποστέλλονται ανώνυμα metrics για τη βελτίωσή της εφαρμογής, ωστόσο υπάρχει η επιλογή απενεργοποίησής τους και είναι πολύ απλή. Η εγκατάστασή του γίνεται μέσω του script run.sh το οποίο κατεβάζει το repository και στην συνέχεια αλλάζει την πόρτα που ακούει η εφαρμογή (από 8080 σε 8888) ώστε να μην συμπίπτει με αυτή του yt-local.

```
cd /usr/local &&
git clone https://github.com/searxng/searxng-docker.git &&
cd searxng-docker &&
sed -i 's/127.0.0.1:8080:8080/0.0.0.0:8888:8080/g' docker-compose.yaml &&
sed -i "s|ultrasecretkey|$(openssl rand -hex 32)|g" searxng/settings.yml &&
docker-compose up -d
```

Δημιουργία Nginx container

Ο Nginx χρησιμοποιείται αφενός σαν web server για το προσωπικό blog του χρήστη και αφετέρου, ως reverse proxy ώστε να κατευθύνει τα requests στον κατάλληλο erver (SearXNG ή yt-local). Στην περίπτωση του yt-local η ανακατεύθυνση πραγματοποιείται απευθείας αφού τα δύο containers ανήκουν στο ίδιο network (my-network).

```
server {
    listen 80;
    server_name yt.andreasdimizas.xyz;

    location / {
        proxy_pass http://andreasdimizas.xyz:8080;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $http_cf_connecting_ip;
    }
}
```

Ωστόσο, επειδή αυτό δεν ισχύει και για τον SearXNG server, αν υπάρχει κάποιο αίτημα που προορίζεται για αυτόν, τότε ανακατευθύνεται στην πόρτα 8888 του host στην οποία ακούει.

```
server {
    listen 80;
    server_name searx.andreasdimizas.xyz;

    location / {
        proxy_pass http://andreasdimizas.xyz:8888;
        proxy_set_header Host $host;
```

```
        proxy_set_header Connection $http_connection;
        proxy_set_header X-Forwarded-For $http_cf_connecting_ip;
        proxy_set_header X-Real-IP $http_cf_connecting_ip;
        proxy_set_header X-Scheme $scheme;
    }
}
```

Για την δημιουργία του container χρησιμοποιούμε το official image και στη συνέχεια αντιστοιχίζουμε την πόρτα 80 του, με την πόρτα 80 του host. Παράλληλα ορίζουμε δύο volumes. Το πρώτο αντιστοιχίζει τον φάκελο public, στον οποία τοποθετούνται τα αρχεία της ιστοσελίδας από το zola, με τον /usr/share/nginx/html, απ' όπου διαβάζει ο nginx το περιεχόμενο που θα σερβίρει. Το δέυτερο αντιστοιχίζει τον φάκελο site.conf που περιέχει το αρχείο site1.comf (configuration του nginx) με τον /etc/nginx/conf.d απ' όπου ο nginx διαβάζει το configuration.

```
  nginx:
    image: nginx
    depends_on:
      - zola
      - yt-local
    ports:
      - "80:80"
    volumes:
      - ./myblog/public:/usr/share/nginx/html
      - ./site.conf:/etc/nginx/conf.d
    networks:
      - my-network
```