



Univerzitet u Zenici
Politehnički fakultet
2022/23



Operativni sistemi

Prva godina I. ciklus

Dokumentacija projekta

**-POSTAVLJANJE JUPYTERHUB SERVERA SA MINIMALNO TRI
PROGRAMSKA JEZIKA PO ŽELJI INSTALIRANA-**

Članovi/ce tima:

- Nejra Smajlović
- Delaida Muminović
- Ensar Krehmić
- Ismet-Abudullah Garanović

user: root

IP adresa: 178.128.204.54

password: operat1vniSistemi

Uvod

Tema projekta: Instalacija JupyterHub servera s minimalno tri programska jezika po želji instalirana.

Korišteni alati i tehnologije:

1| **Docker:** Alat za kreiranje i upravljanje kontejnerima. Nudi izolaciju kao i kapacitet za dupliciranje okruženja. Docker je platforma za pakovanje, distribuciju i pokretanje aplikacija u kontejnerima. Kontejneri su odvojena okruženja koja uključuju aplikaciju i sve njene zavisne komponente, omogućavajući jednostavan prijenos i skaliranje.

Docker se koristi u ovom projektu za izgradnju i izvršavanje kontejnera za Nginx server i Certbot servis, kao i JupyterHub. Certbot je uslužni program koji automatski dobija i obnavlja SSL/TLS certifikate od ovlaštenja LetsEncrypt.

2| **Nginx:** Ovaj server se koristi kao proxy za preusmjeravanje prometa na JupyterHub server. Nudi sigurnu i skalabilnu komunikaciju klijenta i servera.

Nginx je široko korišteni web server i reverse proxy za hosting web stranica, posluživanje sadržaja i usmjeravanje zahtjeva na backend servere. Nginx je poznat po svojoj brzini, skalabilnosti i prilagodljivosti.

3| **JupyterHub:** Glavna komponenta projekta. Instaliran i konfigurisan da omogući brojnim korisnicima na jednom serveru pristup okruženju Jupyter notebook računara.

4| **SSL certifikat:** Ovaj certifikat je instaliran kako bi se omogućila sigurna veza između korisnika i JupyterHub servera. Certifikat osigurava enkripciju podataka i valjanost servera.

5| **Namecheap:** Ova web stranica je korištena za registraciju domene (sistemi.me).

6| **Digital Ocean:** Ova platforma se koristi za kreiranje dropleta, koji su virtualni serveri na kojima su instalirani JupyterHub server i domen.

Opis JupyterHub servera

JupyterHub je platforma otvorenog koda koja omogućava nekoliko korisnika da komuniciraju sa interaktivnim Jupyter notebook-om. Omogućava korisnicima da kreiranje, distribuiranje i izvršavanje sadržaja pomoću Jupyter notebook-a preko web pretraživača. Nudi kolaborativno timsko okruženje za analizu podataka, izradu prototipa, obrazovanje i istraživanje.

Prednosti JupyterHub-a

- ® Omogućava jednostavnu dodjelu radnih okruženja i resursa brojnim korisnicima.
- ® Skalabilan je, što mu omogućava da podrži veliki broj korisnika.
- ® Interfejs je intuitivan, a programski jezici su fleksibilni.
- ® Olakšava saradnju i zajednički rad na projektu.

Nedostaci JupyterHub-a

- ® Za pravilno funkcioniranje zahtijeva odgovarajuće resurse i postavke.
- ® Postavljanje i održavanje servera zahtijeva tehničko znanje.

Ciljevi projekta

- ® Timska grupa radi na kreiranju JupyterHub servera koji podržava 3 programska jezika.
- ® Naučiti kako koristiti Docker, Nginx, JupyterHub, Namecheap, Digital Ocean i druge tehnologije.
- ® Naučiti osnove konfiguracije servera, sigurnosti i integracije s drugim tehnologijama.

Kontekst i inspiracija projekta

- ® Želja za prenošenjem znanja o savremenim IT tehnologijama i aplikacijama.
- ® Poznavanje uobičajenih industrijskih alata i tehnologija.
- ® Stvaranje radnog okruženja koje potiče saradnju, istraživanje i učenje.

Metodologija/Izrada/Koraci rada

Postavljanje okruženja

Korak 1: Uspostavljanje virtuelnog servera

Korišten je Digital Ocean za postavljanje virtuelnog servera (dropleta) sa preferiranim operativnim sistemom, a to je Ubuntu.

Ovo je bio primarni server projekta, gdje je postavljen JupyterHub i Nginx.

Korak 2: Postavljanje domene

Korišten je Namecheap kako bi se pomoću Github Student Developer Pack-a osigurala besplatna domena za JupyterHub server. Nakon što je preuzeta, pomoću Digital Ocean-a je aktivirali i povezana sa virtuelnom mašinom.

Korak 3: Instaliranje potrebnih paketa

Na virtuelnom serveru su instalirani paketi JupyterHub (unutar Dockerfile-a), Docker i Nginx.

Za instaliranje paketa baziranih na operativnom sistemu korištene su komande poput apt-get ili yum.

Korak 4: Konfiguracija sigurnosti i zaštitnog zida

Omogućena je sigurna komunikacija sa JupyterHub serverom aktiviranjem besplatnog SSL certifikata (Let's encrypt).

Korak 5: Izrada konfiguracijskih fajlova

Da bi postavljanje JupyterHub servera na domenu sistemi.me bilo uspješno, ključni korak bio je na ispravan način konfigurirati i ujediniti sve tehnologije koje su korištene. To je, prije svega uključivalo izradu Dockerfile-ova koji su bili osnova za izradu Docker slika. Te slike su omogućavale pokretanje kontejnera unutar kojih je konfigurirano sve što je bilo potrebno za JupyterHub server. Pored toga, u konfiguracijskim fajlovima definisan je način povezivanja nginx poslužitelja, način pristupa JupyterHub notebook-u, kao i postavke SSL certifikata.

Korak 6: Dodavanje programskih jezika unutar JupyterHub notebook-a

Nakon uspješnog pokretanja JupyterHub notebook-a na našoj domeni, izvršili smo instalaciju programskih jezika, tj instalaciju kernela na JupyterHub. Odabir su bili programski jezik Julia i C, dok je treći jezik - Python već došao instaliran. Prilikom instalacije Julie bilo je potrebno ući u "živ" jupyterhub kontejner, i to korištenjem naredbe `docker exec -it jupyterhub bash`. Unutar tog kontejnera instalirana je Julia, te u terminalu samog JupyterHub notebook-a pomoću naredbi `using Pkg` i `Pkg.add("IJulia")` instaliralo kernel i omogućeno je pokretanje tog jezika na serveru. Sličnim postupkom smo instalirali i kernel za programski jezik C.

Konfiguracija fajlova kao temelj izrade projekta

Projekat se sastoji od dva glavna direktorija koji su poslužili za konfiguraciju JupyterHub-a i nginx-a. Hijerarhija tih direktorija i fajlova izgleda ovako:

direktorij jupyterhub_docker:

 poddirektorij config:

 jupyterhub_config.py

 requirements.txt

 poddirektorij Docker:

 jupyterhub.Dockerfile

 docker-compose.yaml

direktorij nginx_https_docker:

 poddirektorij config:

 nginx.conf

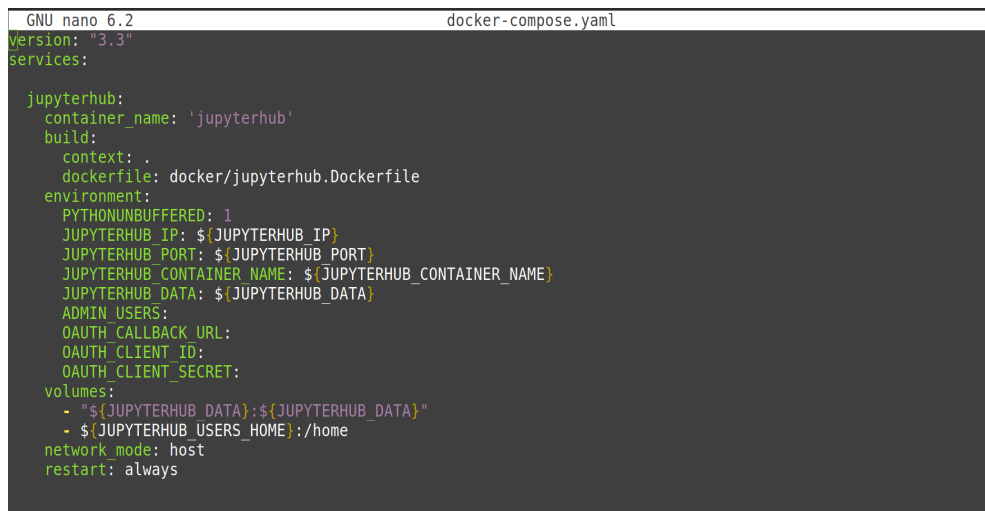
 poddirektorij docker:

 nginx.Dockerfile

 docker-compose-le.yaml

 docker-compose.yaml i .env

docker-compose.yaml:



```
GNU nano 6.2 docker-compose.yaml
version: "3.3"
services:
  jupyterhub:
    container_name: 'jupyterhub'
    build:
      context: .
      dockerfile: docker/jupyterhub.Dockerfile
    environment:
      PYTHONUNBUFFERED: 1
      JUPYTERHUB_IP: ${JUPYTERHUB_IP}
      JUPYTERHUB_PORT: ${JUPYTERHUB_PORT}
      JUPYTERHUB_CONTAINER_NAME: ${JUPYTERHUB_CONTAINER_NAME}
      JUPYTERHUB_DATA: ${JUPYTERHUB_DATA}
      ADMIN_USERS:
      OAUTH_CALLBACK_URL:
      OAUTH_CLIENT_ID:
      OAUTH_CLIENT_SECRET:
    volumes:
      - "${JUPYTERHUB_DATA}:${JUPYTERHUB_DATA}"
      - ${JUPYTERHUB_USERS_HOME}:/home
    network_mode: host
    restart: always
```

Slika 1. docker-compose.yaml

Kroz ovaj file docker se konfiguriše pokretanje jupyterhub servera u izoliranom okruženju, tj u kontejneru, sa specifičnim podešavanjima i okruženjem

version: "3.3" Verzija docker compose formata

services: U ovom bloku se definišu usluge

jupyterhub: U ovom slučaju postoji samo jedna usluga, a to je ona koja se odnosi na jupyterhub server

container_name: 'jupyterhub'

build: Definiše konfiguraciju za izgradnju kontejnera

context: . Specificira trenutni direktorij kao kontekst za izgradnju

dockerfile: `docker/jupyterhub.Dockerfile` putanja do dockerfile-a u kojem će se izgraditi kontejner

environment: postavlja varijable okruženja koje su potrebne za konfiguraciju jupyterhub servera

PYTHONUNBUFFERED: 1 Isključuje baferisanje izlaza u python, što znači da se ispis odmah prikazuje, a ne zadržava

JUPYTERHUB_IP: `${JUPYTERHUB_IP}` Postavlja IP adresu na kojoj jupyterhub sluša dolazne konekcije

JUPYTERHUB_PORT: `${JUPYTERHUB_PORT}` TCP port na kojem server sluša dolazne konekcije, postavljen kao varijabla okruženja (sistemska varijabla koju koristi operativni sistem)

JUPYTERHUB_CONTAINER_NAME:

`${JUPYTERHUB_CONTAINER_NAME}` naziv kontejnera jupyterhub servera

JUPYTERHUB_DATA: `${JUPYTERHUB_DATA}` Putanja do direktorija gdje će se čuvati podaci jupyterhub servera, sve vrijednosti se očekuju kao varijable okruženja

ADMIN_USERS:

OAUTH_CALLBACK_URL:

OAUTH_CLIENT_ID:

OAUTH_CLIENT_SECRET: Admin korisnici, identifikator i tajni ključ klijenta (ovdje je ostavljeno prazno jer je konfigurisano naknadno u posebnom file-u `.env` preko GitHub autentikacije)

volumes:

- `"${JUPYTERHUB_DATA}:${JUPYTERHUB_DATA}"`

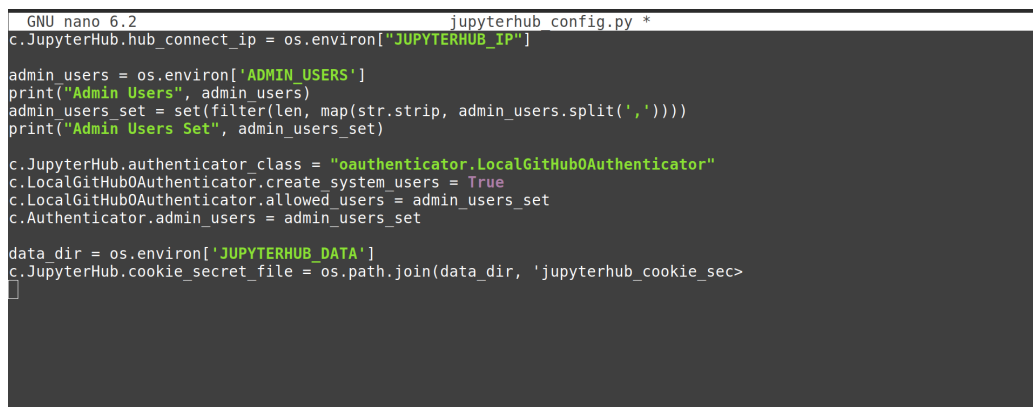
- `"${JUPYTERHUB_USERS_HOME}:/home`

Montiraju se direktoriji za pohranu podataka jupyterhub servera i korisničkih podataka unutar kontejnera

network_mode: host kontejner dijeli mrežu sa host sistemom na kojem je pokrenut (koristi istu mrežnu konfiguraciju kao i host, ovo ne znači da dijele istu IP adresu)

restart: always kontejner se uvijek ponovo pokrene ako se zaustavi ili se sistem ponovo pokrene

jupyterhub_config.py:



```
GNU nano 6.2 jupyterhub config.py *
c.JupyterHub.hub_connect_ip = os.environ["JUPYTERHUB_IP"]

admin_users = os.environ['ADMIN_USERS']
print("Admin Users", admin_users)
admin_users_set = set(filter(len, map(str.strip, admin_users.split(','))))
print("Admin Users Set", admin_users_set)

c.JupyterHub.authenticator_class = "oauthenticator.LocalGitAuthenticator"
c.LocalGitAuthenticator.create_system_users = True
c.LocalGitAuthenticator.allowed_users = admin_users_set
c.Authenticator.admin_users = admin_users_set

data_dir = os.environ['JUPYTERHUB_DATA']
c.JupyterHub.cookie_secret_file = os.path.join(data_dir, 'jupyterhub_cookie_sec>
^_
```

Slika 2. jupyterhub_config.py

import os uvodimo modul (biblioteku) **os** koja nam omogućava pristup različitim funkcionalnostima koje se tiču operativnog sistema, tj funkcionalnosti za interakciju sa operativnim sistemom

c.JupyterHub.ip = os.environ["JUPYTERHUB_IP"] Postavlja IP adresu na kojoj će JupyterHub server slušati za dolazne konekcije. Vrijednost se čita iz okruženjske varijable **JUPYTERHUB_IP**.

(server čeka dolazne zahtjeve na nekoj IP adresi, u ovom slučaju, **JUPYTERHUB_IP** je okruženjska varijabla koja sadrži željenu IP adresu za "slušanje" servera. Kôd čita tu vrijednost iz okruženja i postavlja je kao IP adresu za JupyterHub server)

c.JupyterHub.port = int(os.environ["JUPYTERHUB_PORT"]) (postavlja IP adresu na kojoj jupyterhub 'sluša' dolazne konekcije, ta vrijednost se čita iz varijable okruženja **JUPYTERHUB_PORT**, te s obzirom da se varijable okruženja često spremaju kao stringovi potrebno je tu vrijednost konvertovati u integer da se pravilno postavi TCP port za 'slušanje' konekcija

c.JupyterHub.hub_connect_ip = os.environ["JUPYTERHUB_IP"] postavlja IP adresu kojom se jupyterhub server povezuje s interaktivnim sesijama, tj sa hub kontrolerom, koji upravlja kreiranjem interaktivnih sesija za korisnike (da više korisnika može koristiti server istovremeno i nezavisno), te je važno da te adrese budu usklađene

jupyterhub auth settings

admin_users = os.environ['ADMIN_USERS'] Čita listu administratora iz varijable okruženja **ADMIN_USERS** i dodaje ih varijabli **admin_users**, oni su upisani kao lista korisničkih imena odvojeni zarezom: korisnik1, korisnik2

print("Admin Users", admin_users) Ispisuje listu administratora u konzoli

admin_users_set = set(filter(len, map(str.strip, admin_users.split(',')))) prerađuje tu listu korisnika u set tako što odvaja korisnike koji su odvojeni zarezom, uklanja prazne elemente i razmake

print("Admin Users Set", admin_users_set) ispisuje skup administratora

c.JupyterHub.authenticator_class = "oauthenticator.LocalGitHubOAuthenticator"

Postavlja autentifikatora jupyterhub servera, to se radi, tj koriste se github nalozi za prijavu

c.LocalGitHubOAuthenticator.create_system_users = True omogućava stvaranje korisničkog računa na jupyterhub serveru za svakog korisnika koji se prijavi pomoću github računa

c.LocalGitHubOAuthenticator.allowed_users = admin_users_set postavlja dozvoljene korisinike na već prethodno definition skup administratora, svim ostalim korisnicima neće biti dozvoljena prijava

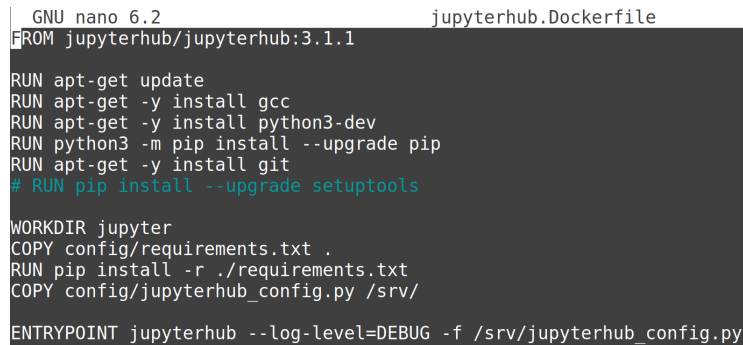
c.Authenticator.admin_users = admin_users_set

#c.JupyterHub.admin_access = True korisnici navedeni u prethodnom setu imaju administratorske privilegije

data_dir = os.environ['JUPYTERHUB_DATA'] iz varijable okruženja čita podatke o korisnicima i dodaje ih novoj varijabli

c.JupyterHub.cookie_secret_file = **os.path.join(data_dir, 'jupyterhub_cookie_sec>**
postavlja putanju do tajnog fajla za kolačiće koji sadrži tajni ključ koji se koristi za enkripciju, a sami kolačići su tu da održavaju korisničke sesije na serveru

jupyterhub.Dockerfile (ovo je file u kojem se gradi kontejner unutar kojeg se pokreće JupyterHub server):



```
GNU nano 6.2 jupyterhub.Dockerfile
FROM jupyterhub/jupyterhub:3.1.1

RUN apt-get update
RUN apt-get -y install gcc
RUN apt-get -y install python3-dev
RUN python3 -m pip install --upgrade pip
RUN apt-get -y install git
# RUN pip install --upgrade setuptools

WORKDIR jupyter
COPY config/requirements.txt .
RUN pip install -r ./requirements.txt
COPY config/jupyterhub_config.py /srv/

ENTRYPOINT jupyterhub --log-level=DEBUG -f /srv/jupyterhub_config.py
```

Slika 3. jupyterhub.Dockerfile

FROM jupyterhub/jupyterhub:3.1.1 bazna slika koja se koristi za izgradnju kontejnera

RUN apt-get update ažurira se popis paketa na sistemu da se izbjegnu problemi sa zastarjelim paketima

RUN apt-get -y install gcc instalacija gcc za pokretanje i kompajliranje programa unutar kontejnera

RUN apt-get -y install python3-dev

RUN python3 -m pip install --upgrade pip

RUN apt-get -y install git instalacija python3 razvojnog okruženja i njegovo ažuriranje na najnoviju verziju, kao i instalacija git alata unutar kontejnera

WORKDIR jupyter postavlja radni direktorij na jupyter u kojem će se izvršavati sve naredbe nakon ove

COPY config/requirements.txt . kopira datoteku iz lokalnog direktorija config u direktorij unutar kontejnera

RUN pip install -r ./requirements.txt instalacija python paketa navedenih u requirements.txt

COPY config/jupyterhub_config.py /srv/ kopira datoteku iz direktorija config u direktorij unutar kontejnera, jer smo u njemu postavili konfiguraciju za kontejner i na taj način osiguravamo da tu konfiguraciju jupyterhub server koristi

ENTRYPOINT jupyterhub --log-level=DEBUG -f /srv/jupyterhub_config.py ova naredba se izvršava prilikom pokretanja kontejnera, te se specificira putanja do filea u kojem je ispisana konfiguracija za jupyterhub server

.env:

```
GNU nano 6.2 .env
JUPYTERHUB_IP=127.0.0.1
JUPYTERHUB_PORT=8000
JUPYTERHUB_DATA=/home/ubuntu/jupyter_hub/data # update folder path
JUPYTERHUB_USERS_HOME=/home/ubuntu/jupyter_hub/home # update folder path

ADMIN_USERS=delaidam,itsnejra
OAUTH_CALLBACK_URL=https://sistemi.me/hub/oauth_callback
OAUTH_CLIENT_ID=90675eeb00e105d34acb
OAUTH_CLIENT_SECRET=11a205faca13d05e549e34e1655c047cc8d49e78
```

Slika 4. .env

JUPYTERHUB_IP=127.0.0.1 Jupyterhub server sluša dolazne konekcije samo na lokalnoj mašini

JUPYTERHUB_PORT=8000 Port na kojem sluša dolazne konekcije, što mora da se podudara sa callback ID-jem github oauth app-a

JUPYTERHUB_DATA=/home/ubuntu/jupyter_hub/data # update folder path Postavlja putanju za podatke

JUPYTERHUB_USERS_HOME=/home/ubuntu/jupyter_hub/home # update folder path Postavlja putanju za korisničke podatke

ADMIN_USERS=delaidam,itsnejra Definiše listu admina, povratnu adresu, ID i client secret koji se koriste prilikom autentifikacije putem oauth protokola

OAUTH_CALLBACK_URL=https://sistemi.me/hub/oauth_callback

OAUTH_CLIENT_ID=90675eeb00e105d34acb

OAUTH_CLIENT_SECRET=11a205faca13d05e549e34e1655c047cc8d49e78

Identifikatori koji se koriste za, i omogućuju prijavu i pristup aplikaciji preko github platforme

(imamo github profil sa identifikacijom aplikacije i url-om na koji se korisnik preusmjerava ako je autorizacija uspješna.

nginx.conf



```
GNU nano 6.2                               nginx.conf
upstream jupyterhub_service {
    server 127.0.0.1:8000;
}

server {
    listen 80;
    listen [::]:80;
    server_name sistemi.me;
    location / {
        return 301 https://$host$request_uri;
    }
    location ~ /.well-known/acme-challenge {
        allow all;
        root /tmp/acme_challenge;
    }
}

server {

    listen 443 ssl;
    listen [::]:443 ssl http2;
    server_name sistemi.me;
    ssl_certificate /etc/letsencrypt/live/sistemi.me/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/sistemi.me/privkey.pem;

    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
    ssl_prefer_server_ciphers on;
    ssl_ciphers 'ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDH>
    ssl_session_timeout 1d;
    ssl_session_cache shared:SSL:50m;
    ssl_stapling on;

    server_name sistemi.me;
    ssl_certificate /etc/letsencrypt/live/sistemi.me/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/sistemi.me/privkey.pem;

    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
    ssl_prefer_server_ciphers on;
    ssl_ciphers 'ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDH>
    ssl_session_timeout 1d;
    ssl_session_cache shared:SSL:50m;
    ssl_stapling on;
    ssl_stapling_verify on;
    add_header Strict-Transport-Security max-age=15768000;
    location / {
        proxy_pass http://jupyterhub_service;
        proxy_redirect off;
        proxy_set_header X-Real-IP $remote_addr;

        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-Proto $scheme;

        # websocket headers
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header X-Scheme $scheme;

        proxy_buffering off;
    }
}
```

Slike 5. i 6. nginx.conf

Konfiguracijski fajl koji služi kao proxy server koji preusmjerava zahtjeve ka JupyterHub serveru. Također, omogućava SSL enkripciju komunikacije putem važećeg SSL certifikata.

```
upstream jupyterhub_service {  
    server 127.0.0.1:8000;
```

```
}
```

Sekcija koja definiše upstream blok koji usmjerava zahtjeve ka IP adresi 127.0.0.1 na protu 8000. Kada se koristi ova konfiguracija, nginx će preuzimati zahtjeve koji dolaze na domenu sistemi.me, a zatim će ih prosljeđivati ka JupyterHub serveru na loopback adresi 127.0.0.1. To omogućava da se zahtjevi preusmjere sa javno dostupne domene ka lokalno pokrenutom JupyterHub serveru.

```
server {  
    listen 80;  
    listen [::]:80;  
    server_name sistemi.me;
```

Ovaj dio konfiguracije označava da nginx server treba da prihvati dolazne HTTP zahtjeve na portu 80 (kako za IPv4, tako i za IPv6) za domenu sistemi.me. Port 80 je standardni HTTP port i koriste se za neenkriptovane HTTP zahtjeve.

```
location / {  
    return 301 https://$host$request_uri;  
}
```

Definiše da će za svaki zahtjev koji stigne na korjensku putanju (‘/’) server izvršiti preusmjeravanje na HTTPS verziju tog zahtjeva. Return 301 znači da će server vratiti statusni kod 301 što označava trajno preusmjeravanje. Klijentski preglednik tada zna da treba da ažurira svoj zahtjev i da koristi HTTPS umjesto HTTP. \$host je ime domene koje je klijent koristio u zahtjevu, a \$request_uri je putanja i svi parametri koji su dio originalnog zahtjeva

```
location ~ /.well-known/acme-challenge {  
    allow all;  
    root /tmp/acme_challenge;  
}  
}
```

Ova konfiguracija omogućava da ACME serveru bude omogućen pristup i provjera vlasništva nad domenom. Ovo je važan korak u procesu dobijanja SSL sertifikata putem ACME protokola.

```
server {  
    listen 443 ssl;  
    listen [::]:443 ssl http2;  
    server_name sistemi.me;
```

Nginx osluškuje zahtjeve na portu 443 koristeći SSL. Osluškuje i IPv4 i IPv6 konekcije

```
ssl_certificate /etc/letsencrypt/live/sistemi.me/fullchain.pem;  
ssl_certificate_key /etc/letsencrypt/live/sistemi.me/privkey.pem;
```

Linije koje definišu putanje do SSL certifikata i privatnog ključa koji se koriste za enkripciju komunikacije.

ssl_protocols TLSv1 TLSv1.1 TLSv1.2; definiše dozvoljene verzije protokola za SSL komunikaciju

ssl_prefer_server_ciphers on; ova opcija govori serveru da koristi preferirane šifre koje su definisane na serveru umjesto šifri koje su poslone od strane klijenta

ssl_ciphers definiše kriptografske šifre koje će se koristiti za enkripciju

ssl_session_timeout 1d; definiše vrijeme trajanja keširane SSL sesije na jedan dan. Kada se koristi SSL enkripcija može se koristiti keširanje sesija kako bise ubrzao proces uspostavljanja veza. Kada klijent prvi put uspostavi sesiju sa serverom, server može keširati određene podatke o sesiji kako bi se sljedeći put ponovo iskoristili umjesto da se ponovo vrši kompletna SSL hand shake procedura.

ssl_session_cache shared:SSL:50m; Konfiguriše se SSL sesija. Parametri specificiraju vrst, naziv i količinu keša (ovdje je 50 megabajta)

ssl_stapling on; omogućava OCSP – Online Certificate Status Protocol. To omogućava serveru da provjeri status certifikata odmah pri pregovaranju SSL sesije umjesto da svaki put pravi dodatni zahtjev serveru koji izdaje certifikate. OBJAŠNJENJE: Kada se uspostavlja SSL sesija između klijenta i servera, server šalje svoj SSL certifikat klijentu. Klijent tada treba da provjeri validnost tog certifikata. Jedan od načina je OCSP protokol. On omogućava klijentu da provjeri da li je SSL certifikat i dalje važeći. Obično bi klijent morao da napravi dodatni zahtjev serveru koji izdaje certifikate kako bi provjerio validnost certifikata. Ovo dovodi do opterećenja SSL sesije. Uključivanjem ove opcije na serveru, server preuzima ulogu provjere validnosti certifikata umjesto klijenta. Server automatski pravi zahtjev OCSP serveru i kešira OCSP odgovor zajedno sa svojim SSL certifikatom. Kada klijent zatraži SSL certifikat, server šalje i keširani OCSP odgovor koji je prethodno preuzeo. Klijent tada može odmah provjeriti validnost certifikata koristeći taj OCSP protokol bez potrebe za dodatnim zahtjevima za OCSP server. Prednosti OCSP: efikasnost, sigurnost i pouzdanost.

ssl_stapling_verify on; Provjera potvrda OCSP odgovora

add_header Strict-Transport-Security max-age=15768000; ovdje se HTTP odgovoru dodaje zaglavlje koje govori pregledaču da koristi samo HTTPS za buduće zahtjeve prema ovoj domeni tokom određenog perioda

location / {

proxy_pass http://jupyterhub_service; definiše da će zahtjevi koji stignu na ovoj putanji biti prosljeđeni serveru koji se pokreće na adresi http://jupyterhub_service . To omogućava nginx serveru da se ponaša kao proxy, tj. da preusmjerava zahtjeve prema drugom serveru.

proxy_redirect off; osigurava da nginx server zadržava kontrolu nad preusmjeravanjem

proxy_set_header X-Real-IP \$remote_addr; Kada nginx prima zahtjev od klijenta, ova stavka je promjenjiva koja sadrži IP adresu tog klijenta. Postavljanje zaglavlja na tu vrijednost omogućava backend serveru da zna IP adresu klijenta čak i ako je zahtjev prošao kroz proxy server. Ovo je korisno u scenarijima kada backend server treba da zna stvarnu adresu klijenta, npr. za sigurnosne ili analitičke svrhe.

proxy_set_header X-Forwarded-For \$proxy_add_x_forwarded_for; pomaže backend serveru da zna odakle dolaze zahtjevi

proxy_set_header Host \$host; omogućava backend serveru da zna koja domena je cilj zahtjeva posebno kada nginx server služi kao proxy za više domena

proxy_set_header X-Forwarded-Proto \$scheme; sadrži informaciju o protokolu koji je korišten od strane klijenta prema serveru

proxy_http_version 1.1; postavlja verziju HTTP protokola za proxy komunikaciju

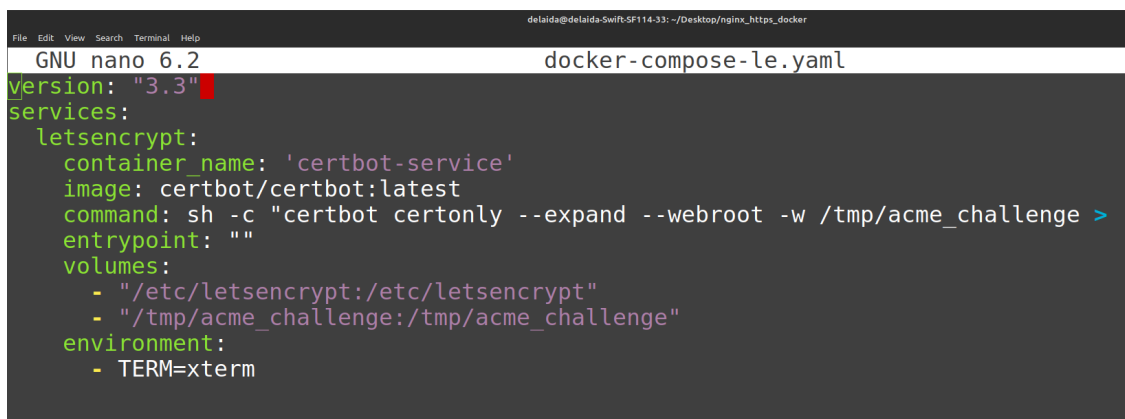
proxy_set_header Upgrade \$http_upgrade; Komunikacija dvosmjerna između klijenta i servera, omogućava pravilno funkcionisanje aplikacija koje koriste Web socket protokol

proxy_set_header Connection "upgrade"; Web socket protokol

proxy_set_header X-Scheme \$scheme; Ako je zahtjev poslan preko HTTPS protokola, vrijednost zaglavlja će biti HTTPS i obrnuto, tj. HTTP

proxy_buffering off; onemogućava keširanje odgovora što znači da će nginx direktno slati odgovor klijentu čim ga primi od backend servera

docker-compose-le.yml:



```
GNU nano 6.2 docker-compose-le.yml
version: "3.3"
services:
  letsencrypt:
    container_name: 'certbot-service'
    image: certbot/certbot:latest
    command: sh -c "certbot certonly --expand --webroot -w /tmp/acme_challenge >
    entrypoint: ""
    volumes:
      - "/etc/letsencrypt:/etc/letsencrypt"
      - "/tmp/acme_challenge:/tmp/acme_challenge"
    environment:
      - TERM=xterm
```

Slika 7. docker-compose-le.yml

version: "3.3" definiše verziju Docker compose specifikacije koja će se koristiti za konfiguraciju

Services: usluge koje će biti pokrenute

Letsencrypt: definiše naziv usluge koja će biti pokrenuta

container_name: 'certbot-service' dodjeljuje ime kontejneru

image: certbot/certbot:latest određuje Docker sliku koja će biti korištena za pokretanje kontejnera (u ovom slučaju se koristi slika **certbot/certbot:latest** koja se koristi za generisanje SSL certifikata

command: sh -c "certbot certonly --expand --webroot -w /tmp/acme_challenge definiše komandu koja će se izvršiti prilikom pokretanja kontejnera. Ova komanda koristi certbot servis za generisanje certifikata sa odgovarajućim opcijama. Expand se koristi za proširivanje postojećeg certifikata –webroot definiše putanju gdje će certbot tražiti verifikacione fajlove prilikom validacije domene

entrypoint: "" označava da se neće vršiti poseban skriptni fajl prilikom pokretanja kontejnera

Volumes: definiše vezivanje direktorija iz host sistema u kontejner

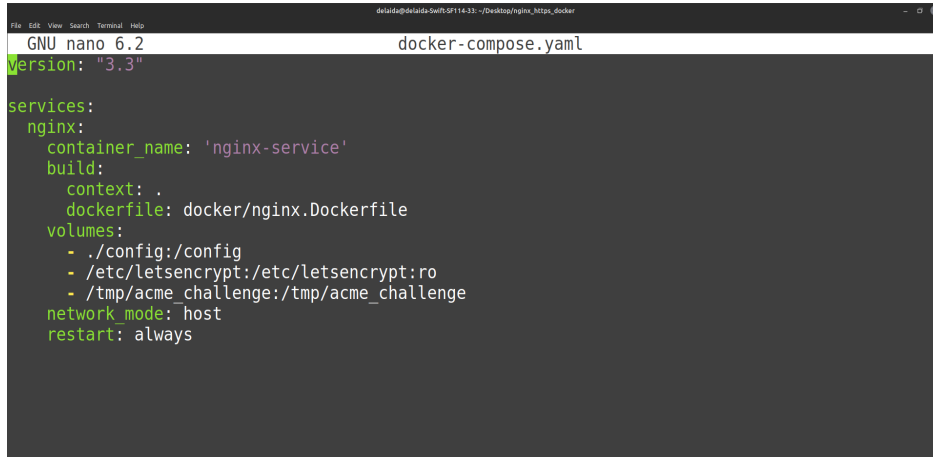
- **"/etc/letsencrypt:/etc/letsencrypt"** montira direktorij iz host sistema u kontejner što omogućava pristup Let's encrypt konfiguracionim fajlovima i generisanim certifikatima

- **"/tmp/acme_challenge:/tmp/acme_challenge"** radi isto kao i prethodna stavka

Environment: definiše okruženje promjenjivih za kontejner

- **TERM=xterm** postavlja promjenjivu TERM na vrijednost xterm. Koristi se za konfiguraciju terminala unutar kontejnera

docker-compose.yaml:



```
GNU nano 6.2 docker-compose.yaml
version: "3.3"

services:
  nginx:
    container_name: 'nginx-service'
    build:
      context: .
      dockerfile: docker/nginx.Dockerfile
    volumes:
      - ./config:/config
      - /etc/letsencrypt:/etc/letsencrypt:ro
      - /tmp/acme_challenge:/tmp/acme_challenge
    network_mode: host
    restart: always
```

Slika 8. docker-compose.yaml

version: "3.3" verzija Docker compose specifikacije

Services: definiše usluge koje će biti pokrenute

Nginx: naziv usluge koja će biti pokrenuta

container_name: 'nginx-service' dodjeljuje ime kontejneru

Build: definiše konfiguraciju za izgradnju kontejnera

context: . definiše trenutni direktorij kao kontekst izgradnje kontejnera. To znači da će Docker tražiti Dockerfile i sve druge relevantne fajlove unutar trenutnog direktorija

dockerfile: docker/nginx.Dockerfile definiše ime Dockerfile-a koji će se koristiti za izgradnju Docker kontejnera

volumes: definiše mount direktorija iz host sistema u kontejner

- **./config:/config** montira /config direktorij iz host sistema što omogućava pristup konfiguracionim fajlovima nginx servera


- **/etc/letsencrypt:/etc/letsencrypt:ro** montira /etc/letsencrypt direktorij iz host sistema u kontejneru u read-only modu. Ovo omogućava nginx serveru pristup certifikatima generisanim od strane Let's encrypt servisa.

- **/tmp/acme_challenge:/tmp/acme_challenge** montira direktorij iz host sistema u kontejner što omogućava nginx serveru pristup verifikacionim fajlovima tokom generisanja certifikata

network_mode: host postavlja mrežni režim kontejnera na host što znači da će kontejner dijeliti istu mrežu za host sistemom što omogućava direktan pristup portovima host sistema

restart: always omogućava da se kontejner automatski restartuje kada se zaustavi ili pokrene host sistem

nginx.Dockerfile:



```
GNU nano 6.2 nginx.Dockerfile
FROM nginx:1.23.0-alpine
RUN rm /etc/nginx/conf.d/default.conf
COPY /config/nginx.conf /etc/nginx/conf.d
```

Slika 9. nginx.Dockerfile

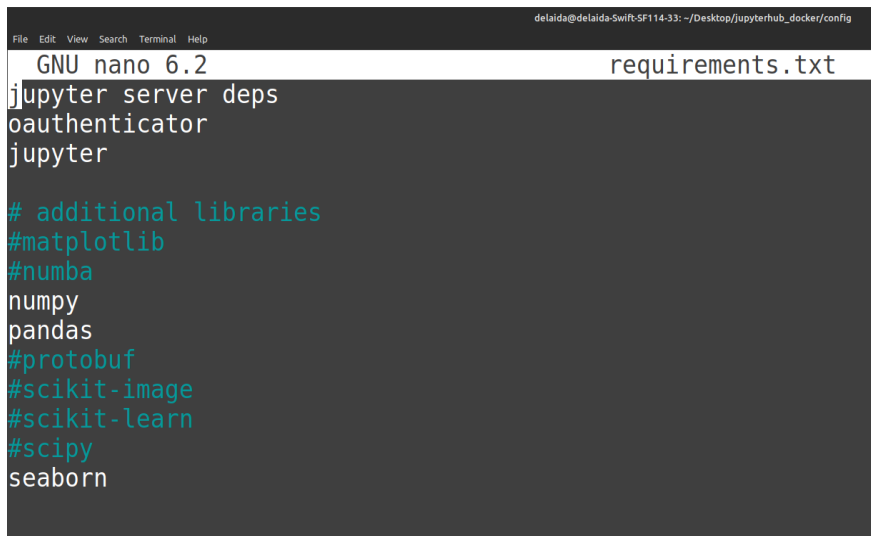
Koristi se za kreiranje Docker image-a za nginx server

FROM nginx:1.23.0-alpine određuje baznu sliku koja će biti korištena za izgradnju Docker image-a. U ovom slučaju koristimo **1.23.0-alpine** što znači da ćemo koristiti alpine Linux verziju kao baznu distribuciju i nginx verziju 1.23.0

RUN rm /etc/nginx/conf.d/default.conf izvršava se tokom izgradnje Docker image-a i uklanja podrazumijevani konfiguracioni fajl iz bazne slike nginx-a. Ovo radimo kako bismo mogli koristiti naš vlastiti konfiguracioni fajl

COPY /config/nginx.conf /etc/nginx/conf.d ova linija kopira naš konfiguracioni fajl sa lokacije **/config/nginx.conf** na određenu putanju unutar Docker image-a.

requirements.txt



```
GNU nano 6.2 requirements.txt
jupyter server deps
oauthenticator
jupyter

# additional libraries
#matplotlib
#numba
numpy
pandas
#protobuf
#scikit-image
#scikit-learn
#scipy
seaborn
```

Slika 10. requirements.txt

Koristi se za navođenje spoljašnjih modula (dependencies) koje projekat zahtjeva.

jupyter server deps oznaka za module zavisnosti za Jupyter server

oauthenticator autentifikacija pomoću protokola

jupyter Jupyter notebook okruženje

Ostale navedene su opcione biblioteke, tj. moduli

Zaključak

U zaključku, ovaj projekat se bazira na podešavanju i konfigurisanju NGINX servera i JupyterHub servera za pružanje sigurne i skalabilne usluge za pristup Jupyter okruženju preko web preglednika. Projekat se oslanja na upotrebu Docker kontejnera za izolaciju i upravljanje serverima. Kroz konfiguraciju NGINX servera, obezbjeđuje se preusmeravanje HTTP zahteva na HTTPS verziju i omogućava SSL enkripcija komunikacije radi zaštite podataka koji se prenose između klijenta i servera. Takođe se koristi i funkcionalnost staplinga OCSP odgovora kako bi se proverio status sertifikata na samom serveru.

JupyterHub server se konfiguriše da radi na određenoj adresi i portu, a NGINX server se koristi kao ulazna tačka za pristup JupyterHub serveru. Proxy podešavanja u NGINX serveru omogućavaju preusmeravanje zahtjeva sa pratećim zaglavljima na JupyterHub server, što omogućava ispravno prosljeđivanje zahtjeva i održavanje veze. Projekat takođe uključuje upotrebu Certbot alata i ACME protokola za automatsko izdavanje i obnovu SSL/TLS sertifikata putem Let's Encrypt servisa.

Željeli bismo izraziti iskrenu zahvalnost asistentici Narcisi Hadžajlić i asistentu Adinu Jahiću na neizmjerljivoj pomoći i podršci koju su nam pružili tijekom izrade našeg projekta. Njihova strpljivost, stručnost i predanost omogućili su nam da prebrodimo izazove i ostvarimo uspješan rezultat.

Izjavljujemo da je ovaj rad, pod nazivom “Postavljanje JupyterHub servera sa minimalno tri programska jezika po želji instalirana”, koji predajem kao projekat na Politehničkom fakultetu u Zenici na odsjeku Softversko inženjerstvo rezultat našeg vlastitog rada.