

مستندات پروژه نهایی کنگرو (فاز اول)

هدف این پروژه ایجاد یک محیط توسعه وب است که از تکنولوژی‌های مختلفی مانند Nginx ، WordPress و MariaDB استفاده می‌کند. با استفاده از Docker Compose ، این پروژه امکان ایجاد و مدیریت محیط توسعه وب را فراهم می‌کند. برخی از اهداف و ویژگی‌های این پروژه عبارتند از:

فراهم کردن یک سرور وب با استفاده از Nginx به عنوان وب سرور اصلی که به کلاینت‌ها ارتباط می‌دهد.

ایجاد دو نمونه از وردپرس (WordPress) که به عنوان سیستم مدیریت محتوا (CMS) برای ایجاد وبسایت‌ها استفاده می‌شود.

استفاده از MariaDB به عنوان سیستم مدیریت پایگاه داده (DBMS) برای ذخیره اطلاعات وردپرس و دیگر اطلاعات مربوط به وبسایت‌ها.

ایجاد یک محیط توسعه قابل تکثیر با استفاده از Docker و Docker Compose که اجازه می‌دهد به راحتی محیط‌های مشابه برای توسعه و تست ایجاد شوند.

توضیحات قسمت اول فایل:

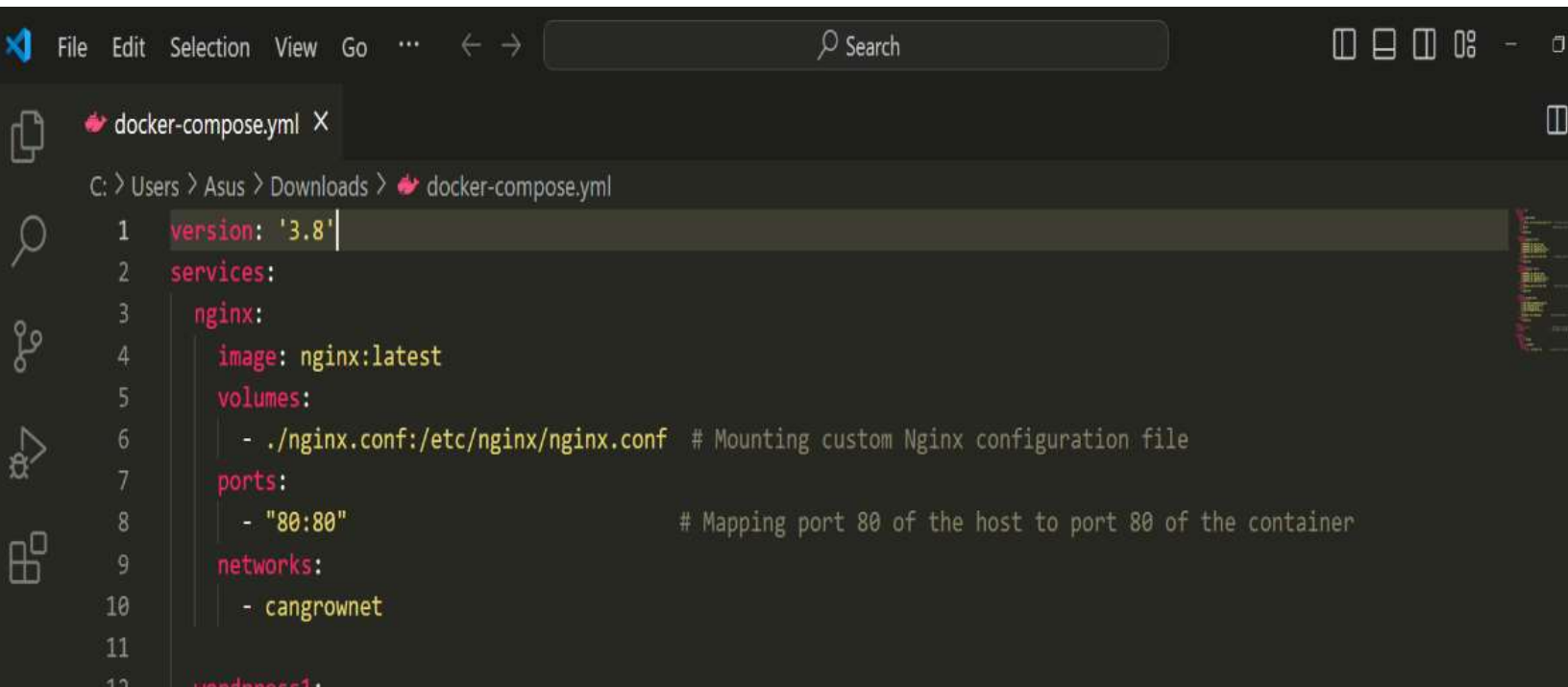
سرویس Nginx:

این سرویس از نسخه Docker nginx:latest استفاده می‌کند که آخرین نسخه موجود از Nginx را فراهم می‌کند.

فایل پیکربندی nginx.conf که در مسیر /etc/nginx/nginx.conf قرار دارد، به مسیر /etc/nginx/nginx.conf در داخل کانتینر متصل می‌شود.

این سرویس بر روی پورت ۸۰ از میزبان به طور قابل دسترس است و به این صورت کانفیگ شده است: "۸۰:۸۰".

این سرویس به شبکه cangrownnet با استفاده از درایور bridge و زیرشبکه ۲۴/۱۷۲.۲۸.۰.۰ متصل می‌شود.



```
1 version: '3.8'
2 services:
3   nginx:
4     image: nginx:latest
5     volumes:
6       - ./nginx.conf:/etc/nginx/nginx.conf # Mounting custom Nginx configuration file
7     ports:
8       - "80:80" # Mapping port 80 of the host to port 80 of the container
9     networks:
10      - cangrownnet
11
12 wordpress1:
```

### سرویس WordPress1 :

این سرویس از نسخه Docker wordpress:latest استفاده می‌کند که آخرین نسخه موجود از وردپرس را فراهم می‌کند.

تنظیمات مربوط به دیتابیس MySQL که باید از آن استفاده کند، به عنوان متغیرهای محیطی ارائه شده‌اند.

داده‌های وردپرس به مسیر /var/www/html داخل کانتینر متصل می‌شود.

این سرویس نیز به شبکه cangrownet متصل می‌شود.

```
11
12 wordpress1:
13   image: wordpress:latest
14   environment:
15     - WORDPRESS_DB_HOST=mariadb
16     - WORDPRESS_DB_USER=wordpress
17     - WORDPRESS_DB_PASSWORD=wordpress
18     - WORDPRESS_DB_NAME=wordpress
19   volumes:
20     - wordpress_data:/var/www/html      # Mounting volume for persistent WordPress data
21   networks:
22     - cangrownet
```

سرویس WordPress2 : مشابه سرویس WordPress1 عمل می‌کند و تنها تفاوت آن این است که یک نمونه دیگر از وردپرس را فراهم می‌کند.

```
2 services:
24 wordpress2:
25   image: wordpress:latest
26   environment:
27     - WORDPRESS_DB_HOST=mariadb
28     - WORDPRESS_DB_USER=wordpress
29     - WORDPRESS_DB_PASSWORD=wordpress
30     - WORDPRESS_DB_NAME=wordpress
31   volumes:
32     - wordpress_data:/var/www/html      # Mounting volume for persistent WordPress data
33   networks:
34     - cangrownet
```

سرویس MariaDB :

این سرویس از نسخه Docker mariadb:latest استفاده می‌کند که آخرین نسخه موجود از MariaDB را فراهم می‌کند.

تنظیمات مربوط به رمز عبور ریشه و نام کاربری و دیتابیس MariaDB به عنوان متغیرهای محیطی تنظیم شده‌اند.

داده‌های MariaDB به مسیر /var/lib/mysql داخل کانتینر متصل می‌شود.

این سرویس نیز به شبکه cangrownnet متصل می‌شود.

```
36 mariadb:
37   image: mariadb:latest
38   environment:
39     - MYSQL_ROOT_PASSWORD=wordpress
40     - MYSQL_DATABASE=wordpress
41     - MYSQL_USER=wordpress
42     - MYSQL_PASSWORD=wordpress
43   volumes:
44     - db_data:/var/lib/mysql      # Mounting volume for persistent MariaDB data
45   networks:
46     - cangrownnet
47
```

همچنین، در بخش volumes، دو بخش تعریف شده‌اند:

wordpress\_data: برای ذخیره دائمی فایل‌های وردپرس

db\_data: برای ذخیره دائمی داده‌های پایگاه داده MariaDB

و در نهایت، در بخش networks، یک شبکه محیطی با نام cangrownnet تعریف شده است که از درایور bridge برای اتصالات شبکه استفاده می‌کند و با زیرشبکه ۲۴/۱۷۲.۲۸.۰.۰ پیکربندی شده است.

```
File Edit Selection View Go ... Search
C:\> Users > Asus > Downloads > docker-compose.yml
48 volumes:
51
52 networks:
53   cangrownnet:
54     driver: bridge
55     ipam:
56       driver: default
57       config:
58         - subnet: 172.28.0.0/24      # Configuring custom subnet for the network
59
```

تنظیمات مربوط به فایل `nginx.conf` برای سرویس `Nginx` در پروژه است. این فایل تنظیمات مربوط به وب سرور `Nginx` را تعیین می کند. `worker_processes 1`: این دستور تعداد فرآیندهای کارگر را مشخص می کند. در اینجا تنها یک فرآیند کارگر تعریف شده است که به این معناست که `Nginx` با یک فرآیند کار می کند.

`events { ... }`: در این بخش، تنظیمات مربوط به رویدادها تعریف می شود. مثلاً تعداد اتصال های همزمان که هر فرآیند کارگر می تواند پذیرفته و مدیریت کند.

```
;worker_processes 1

} events

;worker_connections 1024

{
```

این بخش شامل تنظیمات مختلف برای پروتکل `HTTP` است. برای مثال، `sendfile on` این قابلیت را فعال می کند که فایل های استاتیک به صورت بهینه از سمت سرور به مشتری ارسال شوند. `tcp_nopush` و `tcp_nodelay` نیز تنظیماتی هستند که برای بهبود عملکرد `TCP` استفاده می شوند. `keepalive_timeout` زمانی را که یک اتصال `Keep-Alive` باید باز بماند، تعیین می کند.

`types_hash_max_size` نیز اندازه حداکثر جدول `hash` برای شناسایی نوع `MIME` را تنظیم می کند. در انتها، با استفاده از `include`، فایل `MIME types` برای برخورد با فایل ها شناسایی می شود.

تنظیمات: `HTTP`

`sendfile on; tcp_nopush on; tcp_nodelay on; keepalive_timeout 65;` این تنظیمات مربوط به بهینه سازی عملکرد `HTTP` هستند.

`types_hash_max_size 2048`: تنظیماتی برای مدیریت نوع `MIME` هستند.

`include /etc/nginx/mime.types`: این دستور فایل `MIME types` را برای برخورد با فایل ها اضافه می کند.

`default_type application/octet-stream`: تعیین نوع پیش فرض برای فایل ها.

```

    } http
    ;sendfile on
    ;tcp_nopush on
    ;tcp_nodelay on
    ;keepalive_timeout 65
    ;types_hash_max_size 2048

    ;include /etc/nginx/mime.types
    ;default_type application/octet-stream

    ;access_log /var/log/nginx/access.log
    ;error_log /var/log/nginx/error.log

```

تنظیمات Log: ها

دسترسی به لاگ‌های دسترسی و خطا  
 دسترسی به لاگ‌های خطا و دسترسی  
 دسترسی به لاگ‌های دسترسی و خطا

تنظیمات فشرده‌سازی:

gzip on: فشرده‌سازی فایل‌های ارسالی فعال است.

gzip\_disable "msie6": غیرفعال کردن فشرده‌سازی برای MSIE 6

Upstream و Proxy:

wordpress\_cluster { ... } upstream: ایجاد یک گروه از سرورها (upstream) با نام wordpress\_cluster

server { ... } : تنظیمات ویژه برای سرور، مانند گوش دادن به درخواست‌های HTTP و انتقال آنها به کانتینرهای WordPress از طریق گروه wordpress\_cluster.

```

gzip on; # Enables or disables gzip compression.
gzip_disable "msie6"; # Disables gzip compression for MSIE 6.

upstream wordpress_cluster {
    least_conn; # Specifies that the least-connected method should be used for load balancing.
    server wordpress1:80 weight=10 max_fails=3 fail_timeout=30s; # Specifies the address of the first server in the group.
    server wordpress2:80 weight=10 max_fails=3 fail_timeout=30s; # Specifies the address of the second server in the group.
}

server {
    listen 80; # Sets the port on which the server listens for requests.
    server_name localhost; # Defines the server name.

    location / {
        proxy_pass http://wordpress_cluster; # Sets the backend server for handling requests.
        proxy_redirect off; # Disables proxy redirects.
        proxy_set_header Host $host; # Sets the value of the "Host" header to the value of the $host variable.
        proxy_set_header X-Real-IP $remote_addr; # Sets the value of the "X-Real-IP" header to the value of the $remote_addr variable.
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for; # Appends the value of the $proxy_add_x_forwarded_for variable to the value of the X-Forwarded-For header.
        proxy_set_header X-Forwarded-Proto $scheme; # Sets the value of the "X-Forwarded-Proto" header to the value of the $scheme variable.
    }
}

```

## چالش های فاز ۱

پس از نوشتن ساختار فایل ها و انتقال آن به سرور برای اجرا کردن کانتینر به یک مشکل خوردم که ارور تعریف می کرد کانتینر nginx نمی تواند اجرا شود به این دلیل که پورت ۸۰ مربوط به آن توسط یک سرویس دیگر اشغال است

```

root@cangrow:/home/debian# docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS        NAMES
root@cangrow:/home/debian# docker-compose up -d
Recreating debian_nginx_1 ...
Recreating debian_nginx_1 ... error
Starting debian_wordpress1_1 ...
Starting debian_mariadb_1 ...
: Host is already in use by another container

Starting debian_wordpress2_1 ... done
Starting debian_wordpress1_1 ... done
Starting debian_mariadb_1 ... done

ERROR: for nginx Cannot start service nginx: driver failed programming external connectivity on endpoint de
bian_nginx_1 (40271f8964dd92bdd024fd543fb8212bac94652302933eaa0042be5e8e153c47): Error starting userland pro
xy: listen tcp4 0.0.0.0:80: bind: address already in use
ERROR: Encountered errors while bringing up the project.
root@cangrow:/home/debian#

```

بعد از سرچ متوجه شدم با استفاده از دستور `netstat -tuln | grep ':80'` می توانیم سرویس هایی که از پورت ۸۰ در حال حاضر استفاده می کنند را ببینیم

بعد از آن با دستور `sudo systemctl stop nginx` تمام سرویس های مورد استفاده را متوقف و دوباره فرآیند `up` گرفتن از کانتینر `nginx` آغاز کردم و با موفقیت `up` شد

```

CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS        NAMES
4a14bef5b109   wordpress:latest "docker-entrypoint.s..." 8 hours ago   Up 15 minutes  80/tcp      ordpress2_1
6ee23eaf0dd8   wordpress:latest "docker-entrypoint.s..." 8 hours ago   Up 15 minutes  80/tcp      ordpress1_1
cc60c8977f63   mariadb:latest  "docker-entrypoint.s..." 8 hours ago   Up 15 minutes  3306/tcp    mariadb_1

root@cangrow:/home/debian# sudo systemctl stop tcp6
Failed to stop tcp6.service: Unit tcp6.service not loaded.
root@cangrow:/home/debian# sudo systemctl stop nginx
root@cangrow:/home/debian# sudo netstat -tuln | grep ':80'
root@cangrow:/home/debian# sudo netstat -tuln | grep ':80'
root@cangrow:/home/debian# sudo netstat -tuln | grep ':80'
root@cangrow:/home/debian# docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS        NAMES
4a14bef5b109   wordpress:latest "docker-entrypoint.s..." 8 hours ago   Up 24 minutes  80/tcp      ordpress2_1
6ee23eaf0dd8   wordpress:latest "docker-entrypoint.s..." 8 hours ago   Up 24 minutes  80/tcp      ordpress1_1
cc60c8977f63   mariadb:latest  "docker-entrypoint.s..." 8 hours ago   Up 24 minutes  3306/tcp    mariadb_1

root@cangrow:/home/debian# ls
docker-compose.yml  nginx.conf
root@cangrow:/home/debian# docker-compose up -d
Removing debian_nginx_1
debian_wordpress1_1 is up-to-date

```



```

CONTAINER ID   NAMES          COMMAND                  CREATED        STATUS        PORTS
bf3cad5188b7   nginx:latest   "/docker-entrypoint..." 7 seconds ago  Up 6 seconds  0.0.0.0:80->80/tcp
4a14bef5b109   wordpress:latest "docker-entrypoint.s..." 8 hours ago   Up 24 minutes  80/tcp
6ee23eaf0dd8   wordpress:latest "docker-entrypoint.s..." 8 hours ago   Up 24 minutes  80/tcp
cc60c8977f63   mariadb:latest "docker-entrypoint.s..." 8 hours ago   Up 24 minutes  3306/tcp

root@cangrow:/home/debian# ls
docker-compose.yml  nginx.conf
root@cangrow:/home/debian# docker ps
CONTAINER ID   IMAGE          NAMES          COMMAND                  CREATED        STATUS
bf3cad5188b7   nginx:latest   debian_nginx_1 "/docker-entrypoint..." About a minute ago  Up About a minute
4a14bef5b109   wordpress:latest "docker-entrypoint.s..." 8 hours ago     Up 25 minutes
6ee23eaf0dd8   wordpress:latest "docker-entrypoint.s..." 8 hours ago     Up 25 minutes
cc60c8977f63   mariadb:latest "docker-entrypoint.s..." 8 hours ago     Up 25 minutes
root@cangrow:/home/debian#
```