A Report on Docker Compose Assignment: Setting Up a Composed Server

By,
Neil Duraiswami.

Contents

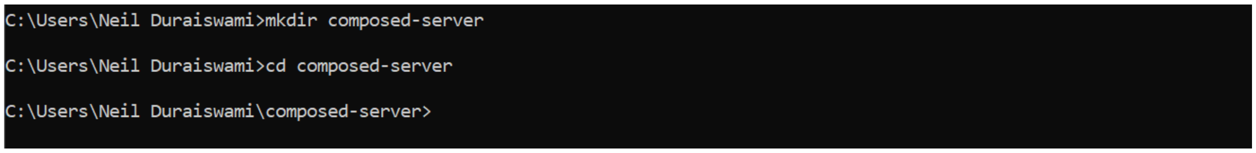
Objective

The objective of this assignment is to familiarize participants with Docker Compose, a tool for defining and managing multi-container Docker applications. Participants will learn to create a composed server environment comprising a web server (Nginx), a database (PostgreSQL), and a data visualization tool (Adminer). By completing the assigned tasks, participants will develop practical skills in defining services, configuring networking, managing volumes and ports, and effectively running and testing Docker Compose projects. This exercise aims to equip participants with a foundational understanding of containerized application deployment and management using Docker Compose.

Task 1: Docker Compose File Creation

Step 1: Create Directory

- Open your terminal or command prompt.
- Execute the command **mkdir composed-server** to create a new directory named composed-server.
- Execute **cd composed-server** to navigate into the newly created directory.



```
C:\Users\Neil Duraiswami>mkdir composed-server

C:\Users\Neil Duraiswami>cd composed-server

C:\Users\Neil Duraiswami\composed-server>
```

Step 2: Create Docker Compose File

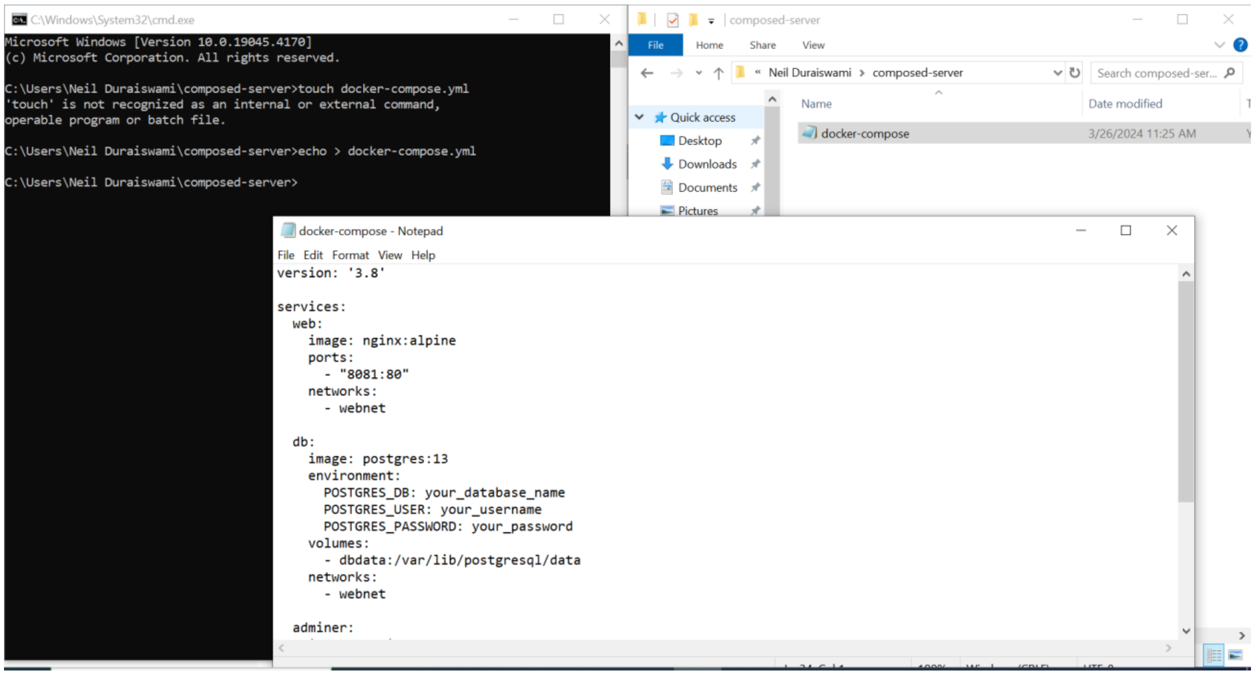
- In the composed-server directory, execute **echo > docker-compose.yml** to create an empty file named docker-compose.yml.

Step 3: Edit Docker Compose File

- Open the docker-compose.yml file using a text editor of your choice (e.g., Notepad, VSCode).
- Copy and paste the following content into the file:

```
version: '3.8'

services:
 web:
  image: nginx:alpine
  ports:
   - "8081:80"
```

```
    networks:
      - webnet

  db:
    image: postgres:13
    environment:
      POSTGRES_DB: your_database_name
      POSTGRES_USER: your_username
      POSTGRES_PASSWORD: your_password
    volumes:
      - dbdata:/var/lib/postgresql/data
    networks:
      - webnet

  adminer:
    image: adminer
    ports:
      - "8082:8080"
    networks:
      - webnet

networks:
  webnet:

volumes:
  dbdata:
```



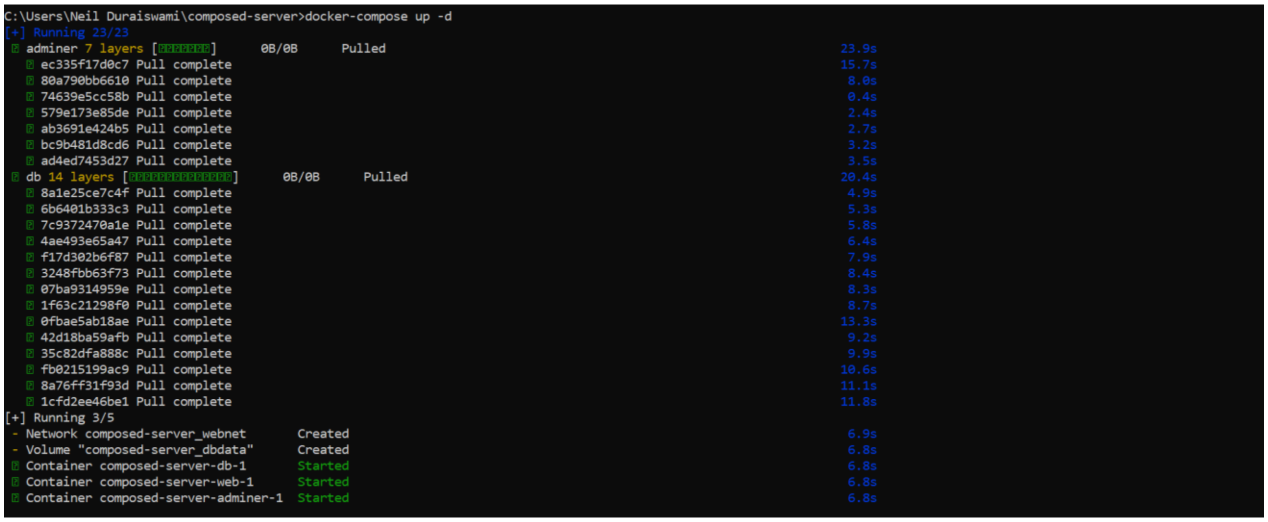
```
Microsoft Windows [Version 10.0.19045.4170]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Neil Duraiswami\composed-server>touch docker-compose.yml
'touch' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\Neil Duraiswami\composed-server>echo > docker-compose.yml

C:\Users\Neil Duraiswami\composed-server>
```

```
version: '3.8'

services:
  web:
    image: nginx:alpine
    ports:
      - "8081:80"
    networks:
      - webnet

  db:
    image: postgres:13
    environment:
      POSTGRES_DB: your_database_name
      POSTGRES_USER: your_username
      POSTGRES_PASSWORD: your_password
    volumes:
      - dbdata:/var/lib/postgresql/data
    networks:
      - webnet

  adminer:
```

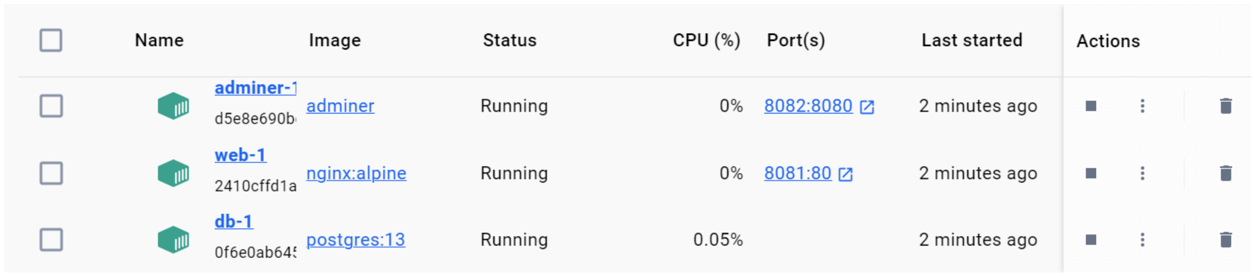
Task 2: Running and Testing Your Composed Environment

Step 1: Start Services

- In your terminal or command prompt, navigate to the composed-server directory if you're not already there.
- Execute the command **docker-compose up -d**. This command starts the services defined in the docker-compose.yml file in detached mode (in the background).



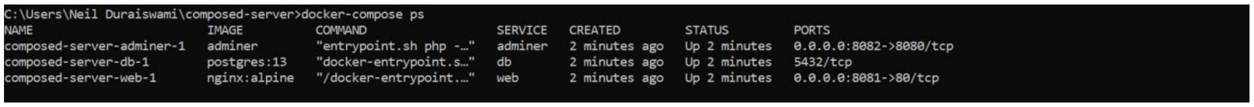
```
C:\Users\Neil Duraiswami\composed-server>docker-compose up -d
[+] Running 23/23
 ▯ adminer 7 layers [▯▯▯▯▯▯▯]      0B/0B      Pulled                                   23.9s
   ▯ ec335f17d0c7 Pull complete                                                        15.7s
   ▯ 80a790bb6610 Pull complete                                                         8.0s
   ▯ 74639e5cc58b Pull complete                                                         0.4s
   ▯ 579e173e85de Pull complete                                                         2.4s
   ▯ ab3691e424b5 Pull complete                                                         2.7s
   ▯ bc9b481d8cd6 Pull complete                                                         3.2s
   ▯ ad4ed7453d27 Pull complete                                                         3.5s
 ▯ db 14 layers [▯▯▯▯▯▯▯▯▯▯▯▯▯▯]      0B/0B      Pulled                                20.4s
   ▯ 8a1e25ce7c4f Pull complete                                                         4.9s
   ▯ 6b6401b333c3 Pull complete                                                         5.3s
   ▯ 7c9372470a1e Pull complete                                                         5.8s
   ▯ 4ae493e65a47 Pull complete                                                         6.4s
   ▯ f17d302b6f87 Pull complete                                                         7.9s
   ▯ 3248fbb63f73 Pull complete                                                         8.4s
   ▯ 07ba9314959e Pull complete                                                         8.3s
   ▯ 1f63c21298f0 Pull complete                                                         8.7s
   ▯ 0fbae5ab18ae Pull complete                                                        13.3s
   ▯ 42d18ba59afb Pull complete                                                         9.2s
   ▯ 35c82dfa888c Pull complete                                                         9.9s
   ▯ fb0215199ac9 Pull complete                                                        10.6s
   ▯ 8a76ff31f93d Pull complete                                                        11.1s
   ▯ 1cfd2ee46be1 Pull complete                                                        11.8s
[+] Running 3/5
 - Network composed-server_webnet       Created                                         6.9s
 - Volume "composed-server_dbdata"      Created                                         6.8s
 ▯ Container composed-server-db-1       Started                                         6.8s
 ▯ Container composed-server-web-1      Started                                         6.8s
 ▯ Container composed-server-adminer-1  Started                                         6.8s
```



☐	Name	Image	Status	CPU (%)	Port(s)	Last started	Actions
☐	adminer-1 d5e8e690b	adminer	Running	0%	8082:8080	2 minutes ago	■ ⋮ 🗑
☐	web-1 2410cffd1a	nginx:alpine	Running	0%	8081:80	2 minutes ago	■ ⋮ 🗑
☐	db-1 0f6e0ab64	postgres:13	Running	0.05%		2 minutes ago	■ ⋮ 🗑

Step 2: Check Status

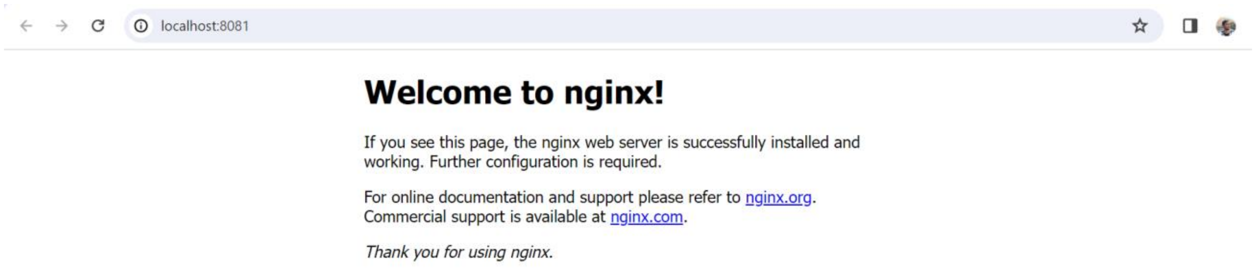
- Execute **docker-compose ps** to check the status of the running containers. This command will show you if the containers are running or not.



```
C:\Users\Neil Duraiswami\composed-server>docker-compose ps
NAME                        IMAGE          COMMAND                   SERVICE    CREATED          STATUS          PORTS
composed-server-adminer-1   adminer        "entrypoint.sh php -…"   adminer    2 minutes ago    Up 2 minutes    0.0.0.0:8082->8080/tcp
composed-server-db-1        postgres:13    "docker-entrypoint.s…"   db         2 minutes ago    Up 2 minutes    5432/tcp
composed-server-web-1       nginx:alpine   "/docker-entrypoint.…"   web        2 minutes ago    Up 2 minutes    0.0.0.0:8081->80/tcp
```

Step 3: Testing the Web Server

- Open your web browser.
- Navigate to http://localhost:8081. You should see the Nginx welcome page.



localhost:8081

Welcome to nginx!

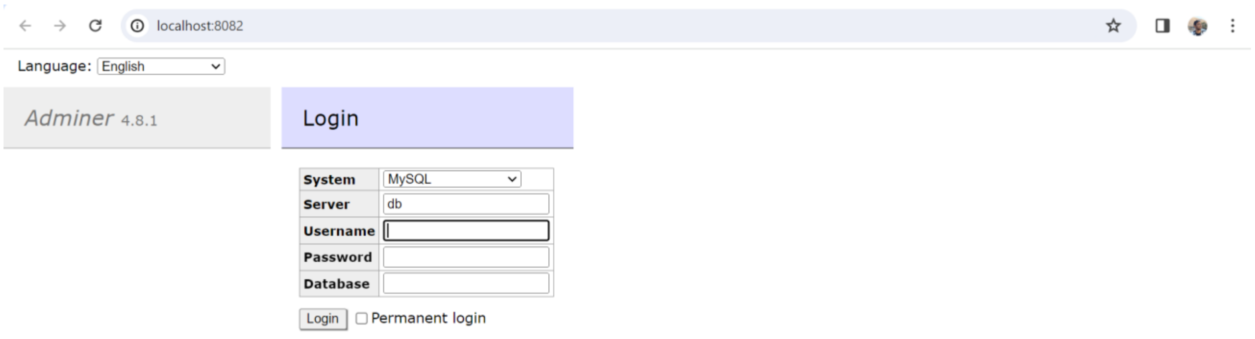
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Step 4: Testing the Database and Adminer

- In your web browser, navigate to http://localhost:8082. This should bring up the Adminer login page.



localhost:8082

Language: English

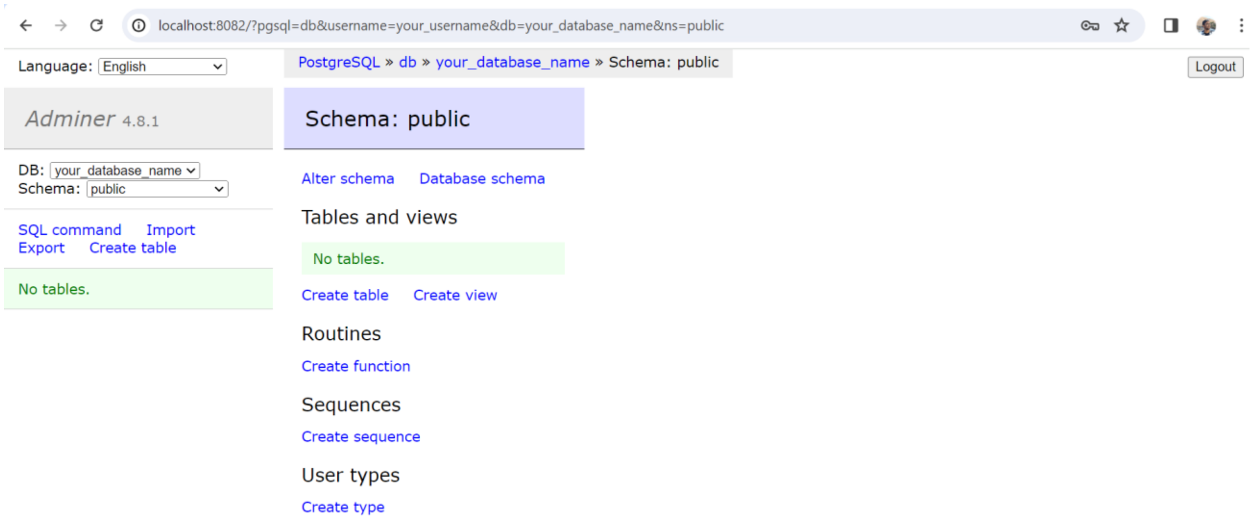
Adminer 4.8.1

Login

System	MySQL
Server	db
Username	
Password	
Database	

Login ☐ Permanent login

- Log in using the PostgreSQL credentials you defined in the Docker Compose file.



localhost:8082/?pgsql=db&username=your_username&db=your_database_name&ns=public

Language: English

PostgreSQL » db » your_database_name » Schema: public

Logout

Adminer 4.8.1

DB: your_database_name
Schema: public

SQL command Import
Export Create table

No tables.

Schema: public

Alter schema Database schema

Tables and views

No tables.

Create table Create view

Routines

Create function

Sequences

Create sequence

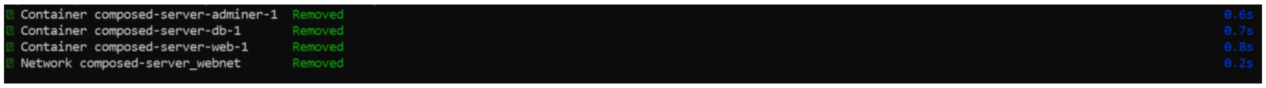
User types

Create type

Task 3: Docker Compose Management

Step 1: Stop Services

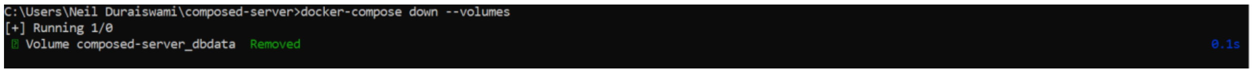
- In your terminal or command prompt, make sure you're still in the composed-server directory.
- Execute **docker-compose down** to stop and remove the containers created by Docker Compose.



```
 Container composed-server-adminer-1  Removed      0.6s
 Container composed-server-db-1       Removed      0.7s
 Container composed-server-web-1      Removed      0.8s
 Network composed-server_webnet       Removed      0.2s
```

Step 2: Cleanup

- While still in the composed-server directory, execute **docker-compose down --volumes**. This command removes all stopped containers, networks, and volumes created by Docker Compose.



```
C:\Users\Neil Duraiswami\composed-server>docker-compose down --volumes
[+] Running 1/0
 Volume composed-server_dbdata  Removed      0.1s
```

Conclusion

In conclusion, this assignment has provided participants with valuable hands-on experience in utilizing Docker Compose to orchestrate multi-container Docker applications. Through step-by-step tasks, participants successfully created a composed server environment and gained proficiency in defining services, configuring networking, managing volumes and ports, and executing Docker Compose projects. This practical exercise has enhanced participants' comprehension of containerized application deployment and management, fostering essential skills applicable to various Docker-based development scenarios. By completing this assignment, participants have achieved a significant milestone in their journey toward mastering Docker Compose, empowering them to build and deploy complex containerized applications in diverse environments efficiently.