

# Docker Composed

## Setting up a composed webserver

### Introduction:

This report outlines the methodology for establishing a Docker Compose environment tailored for web application development. This environment will encompass three core components: a web server, a database, and an administrative interface. The deployment will leverage Docker and Docker Compose, ensuring seamless communication and data persistence among the services. A solid understanding of YAML syntax and web server configurations is a prerequisite for this endeavor.

### Pre-requisites:

- Docker and Docker Compose are installed on your machine. Verify installation by running `docker --version` and `docker-compose --version` in your terminal.
- Basic understanding of YAML syntax and web server configurations.

### Task 1: Docker Compose File Creation

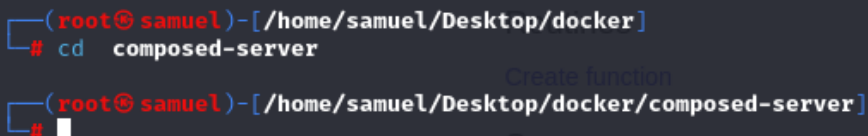
1. Create a directory for your project:

- Create a new directory named `composed-server`.
- Navigate into the directory.



```
(root@samuel)-[/home/samuel/Desktop/docker]
# mkdir composed-server
```

- 



```
(root@samuel)-[/home/samuel/Desktop/docker]
# cd composed-server

(root@samuel)-[/home/samuel/Desktop/docker/composed-server]
#
```

- 

2. Create a Docker Compose file:

- Inside the `composed-server` directory, create a file named `docker-compose.yml`.
- Open the file in a text editor.

```
(root@samuel)-[/home/samuel/Desktop/docker/composed-server]
# touch docker-compose.yml

(root@samuel)-[/home/samuel/Desktop/docker/composed-server]
# gedit docker-compose.yml
```

3. Define Services:

- In the docker-compose.yml file, define three services: web, db, and adminer.
- Use the nginx:alpine image for the web service.
- Use the postgres:13 image for the db service. Define environment variables for POSTGRES\_DB, POSTGRES\_USER, and POSTGRES\_PASSWORD.
- Use the adminer image for the adminer service.

4. Configure Networking:

- Create a custom network called webnet for your services to communicate.
- Assign all three services to this network.

5. Volumes and Ports:

- Mount a volume for the PostgreSQL data to persist data outside the container lifecycle.
- Map port 8080 to the web service's port 80.
- Map port 8081 to the adminer service's port 8080.

```
docker-compose.yml
/home/samuel/Desktop/docker/composed-server

1 version: "3.1"
2
3 services:
4   web:
5     image: nginx:alpine
6     ports:
7       - "8080:80"
8     networks:
9       - webnet
10  db:
11    image: postgres:13
12    environment:
13      POSTGRES_DB: exampledb
14      POSTGRES_USER: exampleuser
15      POSTGRES_PASSWORD: examplepass # Corrected the typo here
16    volumes:
17      - db-data:/var/lib/postgresql/data # Corrected the path and removed the 'type' property
18    networks:
19      - webnet
20  adminer:
21    image: adminer
22    ports:
23      - "8081:8080"
24    networks:
25      - webnet
26
27 networks:
28   webnet:
29
30 volumes: |
31 db-data:
```

## Task 2: Running and Testing Your Composed Environment

### 1. Starting Services:

- Run your composed environment with the following command:  
`docker-compose up -d`

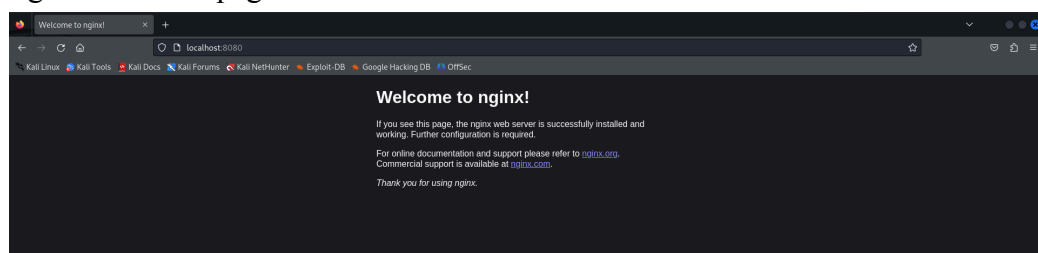
```
root@samuel:~# docker-compose up -d
Creating network "compromised-server_webnet" with the default driver
Creating volume "compromised-server_db-data" with default driver
Pulling db (postgres:13)...
13: Pulling from library/postgres
8a1e25ce7c4f: Already exists
6b6401b333c3: Pull complete
7c9372470a1e: Pull complete
4ae493e65a47: Pull complete
f17d302b6f87: Pull complete
3248fbb63f73: Pull complete
07ba9314959e: Pull complete
1f63c21298f0: Pull complete
0fbae5ab18ae: Pull complete
42d18ba59afb: Pull complete
35c82dfa888c: Pull complete
fb0215199ac9: Pull complete
8a76ff31f93d: Pull complete
1cfd2ee46be1: Pull complete
Digest: sha256:3840140f04b26432cd66de0e3a9a63dfc9fed70bb27e161c846b50fe754a9893
Status: Downloaded newer image for postgres:13
Pulling adminer (adminer)...
latest: Pulling from library/adminer
ec335f17d0c7: Pull complete
80a790bb6610: Pull complete
74639e5cc58b: Pull complete
579e173e85de: Pull complete
ab3691e424b5: Pull complete
bc9b481d8cd6: Pull complete
ad4ed7453d27: Pull complete
Digest: sha256:b75eae89431e8469613b844e76382a26efc8601c17f446bcd81665bc87ca9a1f
Status: Downloaded newer image for adminer:latest
Creating compromised-server_adminer_1 ... done
Creating compromised-server_db_1 ... done
Creating compromised-server_web_1 ... done
```

- Ensure all containers are running by using:  
`docker-compose ps`

```
root@samuel:~# docker-compose ps
Name                                Command                                State      Ports
-----                                -
compromised-server_adminer_1        entrypoint.sh php -S [::]: ...        Up         0.0.0.0:8081->8080/tcp, ::: 8081->8080/tcp
compromised-server_db_1             docker-entrypoint.sh postgres         Up         5432/tcp
compromised-server_web_1            /docker-entrypoint.sh nginx ...       Up         0.0.0.0:8080->80/tcp, ::: 8080->80/tcp
```

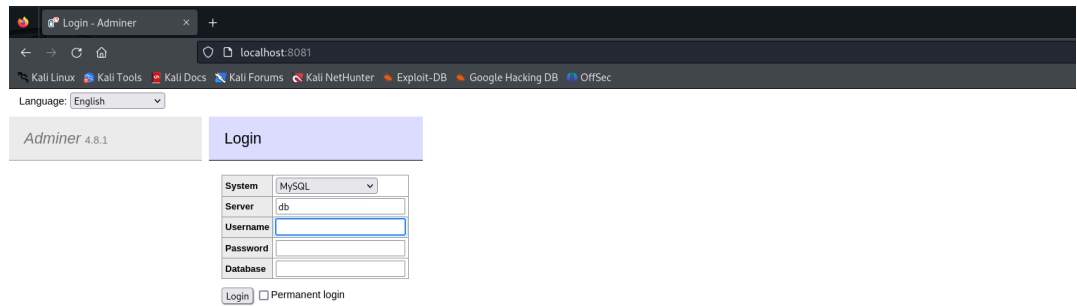
### 2. Testing the Web Server:

- Open your browser and navigate to `http://localhost:8080`. You should see the Nginx welcome page.

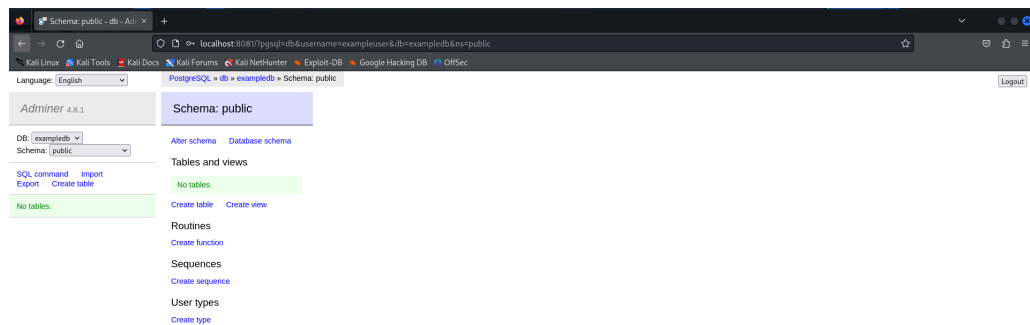


### 3. Testing the Database and Adminer:

- Open your browser and navigate to `http://localhost:8081`. You should see the Adminer login page.
- Log in using the PostgreSQL credentials you defined in your Docker Compose file.



- 
- Click on the MySQL dropdown to switch to PostgreSQL and then log into the database using the credentials mentioned in the docker-compose.yml file.



○

## Task 3: Docker Compose Management

### 1. Stopping Services:

- Stop your composed environment using:  
docker-compose down

### 2. Cleanup:

- Remove all stopped containers, networks, and volumes created by Docker Compose using:  
docker-compose down --volumes

```
(root@samuel)-[/home/samuel/Desktop/docker/composed-server]
# docker-compose ps

```

Name	Command	State	Ports
composed-server_adminer_1	entrypoint.sh php -S [::]: ...	Up	0.0.0.0:8081→8080/tcp, :::8081→8080/tcp
composed-server_db_1	docker-entrypoint.sh postgres	Up	5432/tcp
composed-server_web_1	/docker-entrypoint.sh nginx ...	Up	0.0.0.0:8080→80/tcp, :::8080→80/tcp

```
(root@samuel)-[/home/samuel/Desktop/docker/composed-server]
# docker-compose down
Stopping composed-server_db_1 ... done
Stopping composed-server_adminer_1 ... done
Stopping composed-server_web_1 ... done
Removing composed-server_db_1 ... done
Removing composed-server_adminer_1 ... done
Removing composed-server_web_1 ... done
Removing network composed-server_webnet

(root@samuel)-[/home/samuel/Desktop/docker/composed-server]
# docker-compose down --volumes
Removing network composed-server_webnet
WARNING: Network composed-server_webnet not found.
Removing volume composed-server_db-data
```

○