DOCKER --LINK

Running a WordPress Blog Using Two Linked Containers

AIM: You want to run a WordPress site with containers, but you do not want to run the MySQL database in the same container as WordPress. You want to keep the concept of separation of concerns in mind and decouple the various components of an application as much as possible.

Solution

Note: must be installed docker in your machine**.**

You start two containers: one running WordPress using the official image from the Docker Hub, and one running the MySQL database. The two containers are linked using the --link option of the Docker CLI. Start by pulling the latest images for WordPress and MySQL:

$ docker pull wordpress

$ docker pull mysql

$ docker images

```
[ec2-user@ip-172-31-29-246 ~]$  docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
wordpress           latest              7539ce0f28d0        5 days ago
420MB
mysql               latest              81f094a7e4cc        8 days ago
477MB
[ec2-user@ip-172-31-29-246 ~]$
```

Start a MySQL container, give it a name via the --name CLI option, and set the MYSQL_ROOT_PASSWORD via an environment variable:

//$ docker run --name mysqlwp -d mysql → it won't run success because need to be set password

//$ docker run --name mysqlwp -e MYSQL_ROOT_PASSWORD=wordpressdocker -d mysql

$docker run -d --name mysqlwp -e MYSQL_ROOT_PASSWORD=wordpressdocker mysql mysqld --default-authentication-plugin=mysql_native_password

```
[ec2-user@ip-172-31-29-246 ~]$ docker run --name mysqlwp -e MYSQL_ROOT_PASSWORD=
wordpressdocker -d mysql
c23137877c55f59faa632483ff89ddb024db271d5038cf3376d38a3e7ece944e
```

You can now run a WordPress container based on the wordpress:latest image. It will be linked to the MySQL container using the --link option, which means that Docker will automatically set up the networking so that the ports exposed by the MySQL container are reachable inside the WordPress container:

$ docker run --name wordpress --link mysqlwp:mysql -p 80:80 -d wordpress

```
[ec2-user@ip-172-31-29-246 ~]$ docker run --name wordpress --link mysqlwp:mysql
-p 80:80 -d wordpress
8660c55983bc8edb1222be3fb7fa32995b372ef3260b8c04195706b38765c20f
```

Both containers should be running in the background, with port 80 of the WordPress container mapped to port 80 of the host:

$ docker ps

```
[ec2-user@ip-172-31-29-246 ~]$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
b5e9d7550db4        wordpress           "docker-entrypoint.s…"   14 seconds ago      Up 13 seconds
4b6999995430        mysql               "docker-entrypoint.s…"   42 seconds ago      Up 41 seconds
```

Open a browser at http://<ipaddress> and it should show the WordPress installation screen with the language selection window. If you go through the WordPress setup, you will then have a fully functional WordPress site running with two linked containers.

If you see wordpress container not running and see the errors in word press logs do below steps

MySQL Connection Error: (2054) The server requested authentication method unknown to the client

Warning: mysqli::__construct(): The server requested authentication method unknown to the client [caching_sha2_password] in Standard input code on line 22

Warning: mysqli::__construct(): (HY000/2054): The server requested authentication method unknown to the client in Standard input code on line 22

First login sql container using docker exec and

$ mysql -u root -p

`You should be able to login as 'root' or your other password.

Execute the following commands:

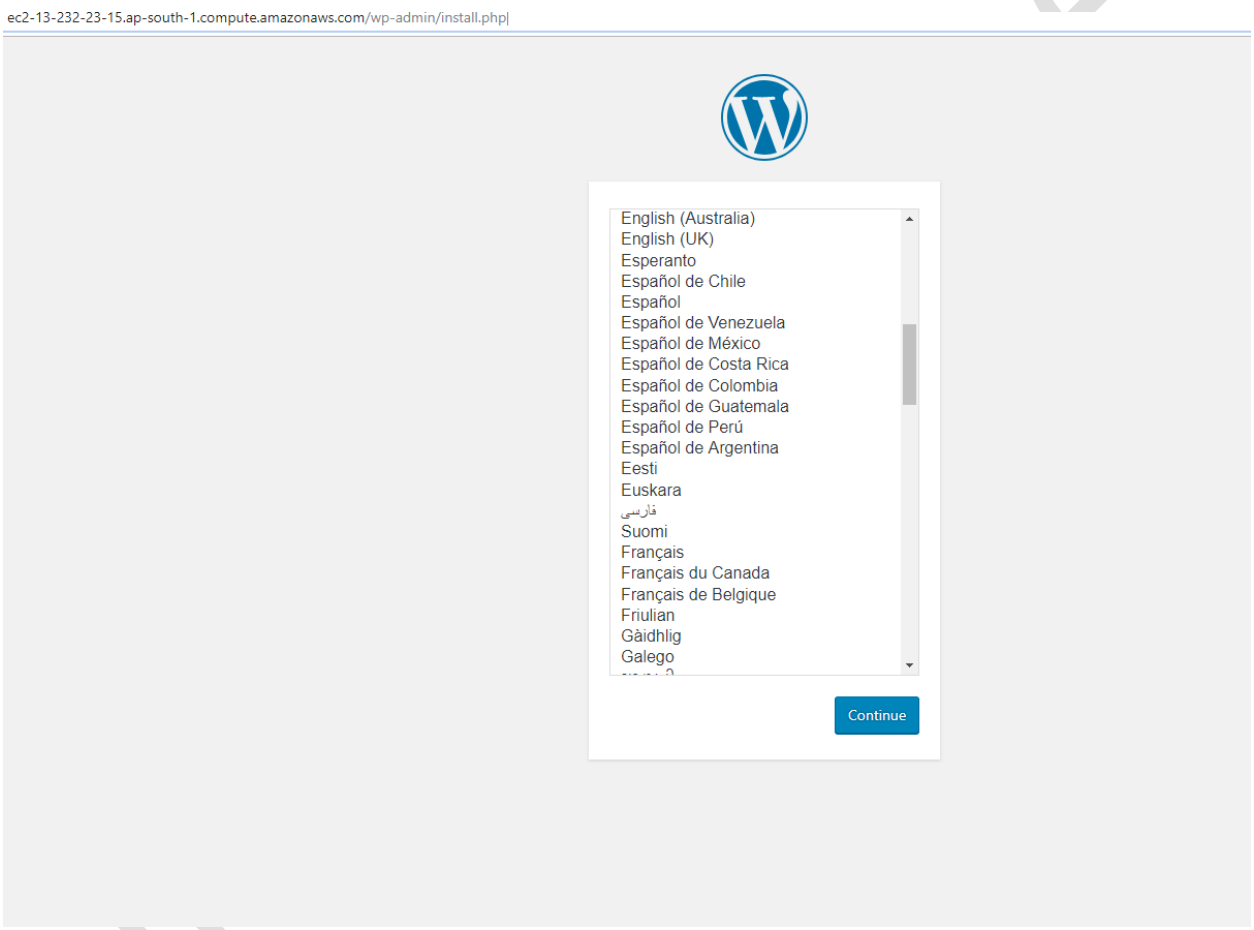
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'wordpressdocker';

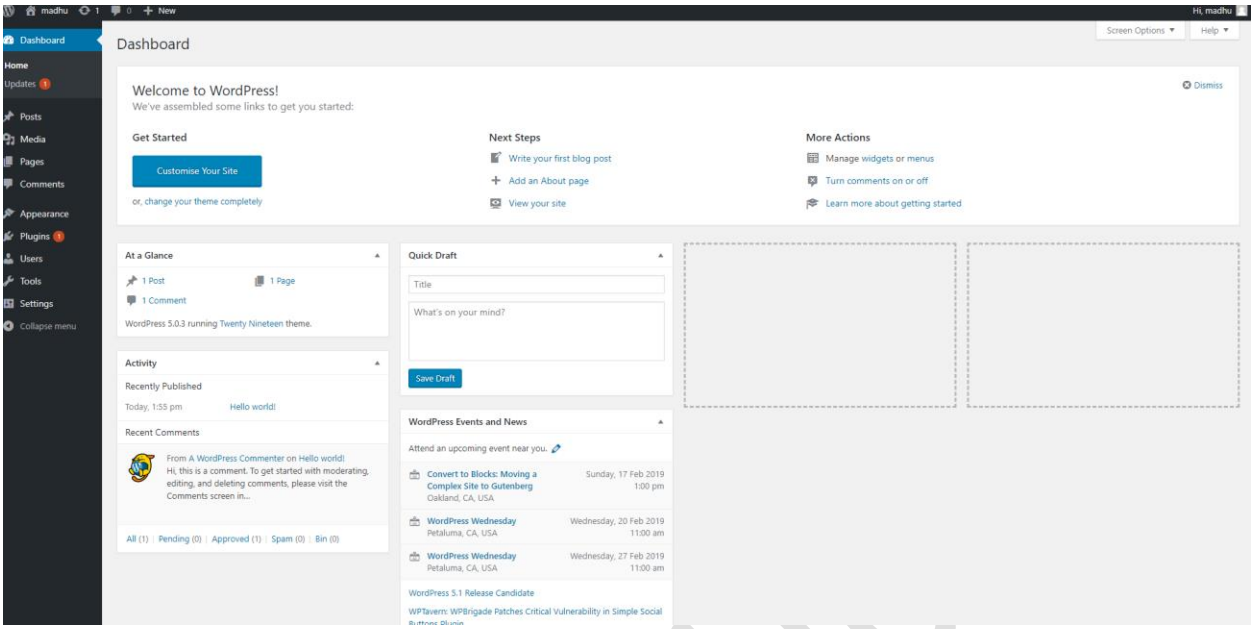
ALTER USER 'root'@'%' IDENTIFIED WITH mysql_native_password BY 'wordpressdocker';

FLUSH PRIVILEGES;

Come out form the container ctrl+p+q

Open browser and use http://<serverip>





The two images for WordPress and MySQL are official images maintained by the WordPress and MySQL communities. Each page on the Docker Hub provides addi- tional documentation for configuration of containers started with those images.

Of interest is that you can create a database and a user with appropriate privileges to manipulate that database by using a few environment variables: MYSQL_DATABASE, MYSQL_USER, and MYSQL_PASSWORD. In the preceding example, WordPress is run as the root MySQL user and this is far from best practice. It would be better to create a word press database and a user for it, like so:

```
$ docker run --name mysqlwp -e MYSQL_ROOT_PASSWORD=wordpressdocker \

-e MYSQL_DATABASE=wordpress \

-e MYSQL_USER=wordpress \

-e MYSQL_PASSWORD=wordpresspwd \

-d mysql
```

Once the database container is running, you run the WordPress container and specify the database tables you defined:

```
$ docker run --name wordpress --link mysqlwp:mysql -p 80:80 \

-e WORDPRESS_DB_NAME=wordpress \
```

-e WORDPRESS_DB_USER=wordpress \

-e WORDPRESS_DB_PASSWORD=wordpresspwd \

-d wordpress

For learn more about wordpress and mysql docker images and usage refer below links

https://hub.docker.com/_/wordpress/

https://hub.docker.com/_/mysql

Note : Reach me to learn devops hands-on practical in different scenarios