

NIPUN SINGAL
500069052
R171218069

Application Containerization Lab

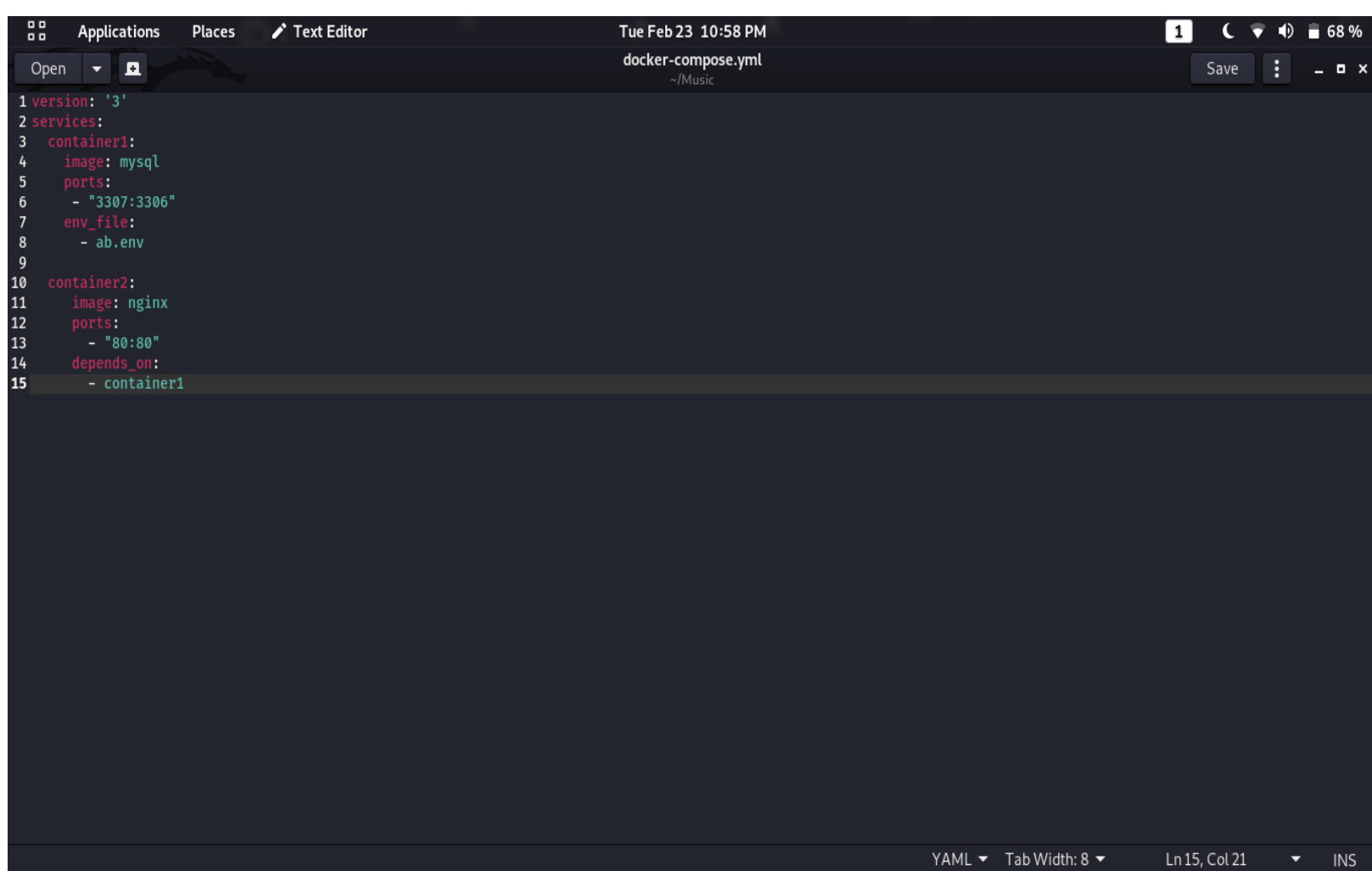
EXPERIMENT-6

Creating and using docker-compose file

Step1: Create a docker-compose file

syntax: **touch docker-compose.yml**

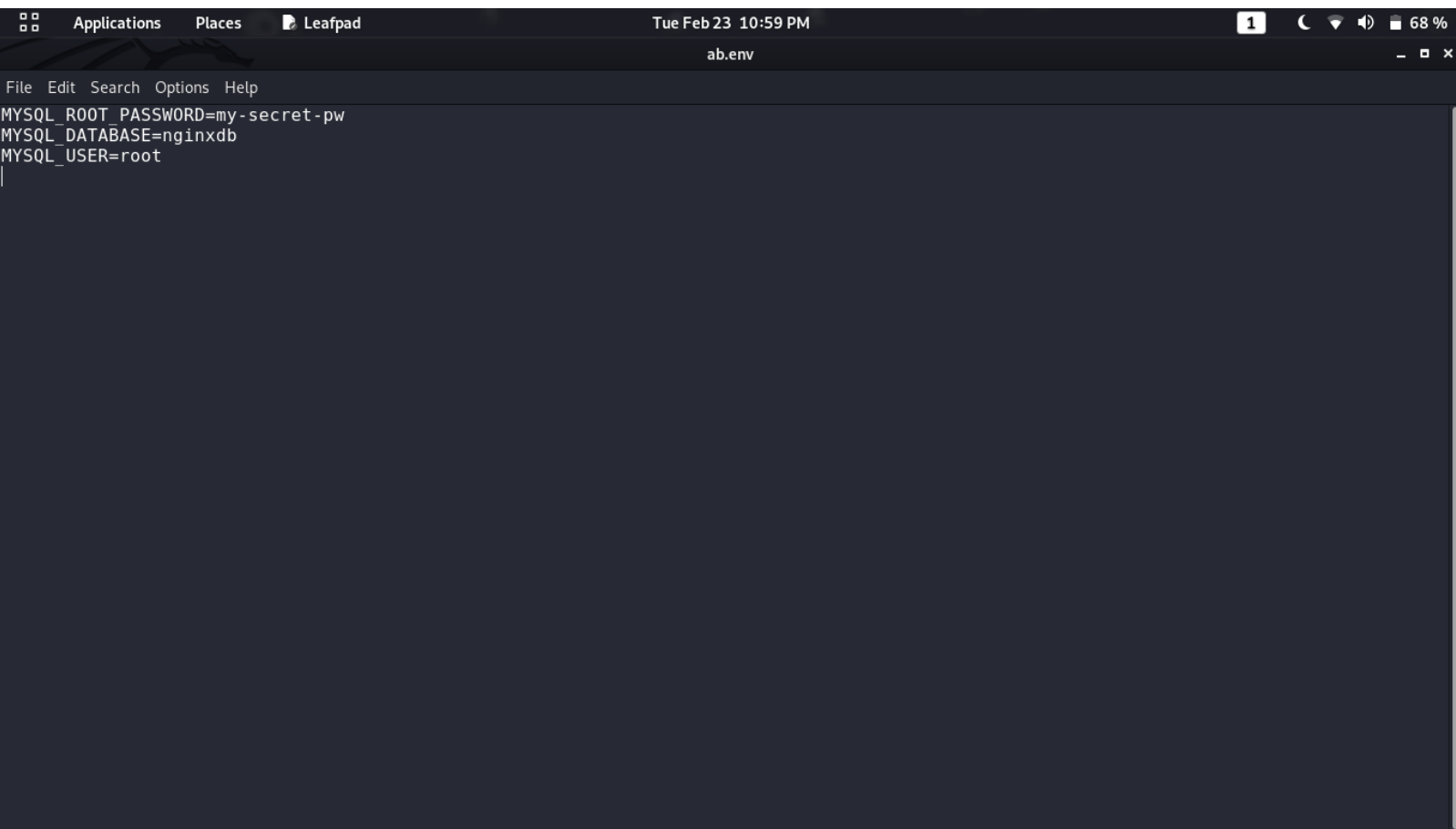
Step2: Write the instructions in it.

A screenshot of a text editor window with a dark theme. The window title bar shows 'Applications', 'Places', and 'Text Editor'. The file name is 'docker-compose.yml' located at '~/Music'. The editor contains a YAML configuration for two services: 'container1' (mysql) and 'container2' (nginx). The status bar at the bottom indicates 'YAML', 'Tab Width: 8', 'Ln 15, Col 21', and 'INS' mode.

```
1 version: '3'
2 services:
3   container1:
4     image: mysql
5     ports:
6     - "3307:3306"
7     env_file:
8     - ab.env
9
10  container2:
11    image: nginx
12    ports:
13    - "80:80"
14    depends_on:
15    - container1
```

Step3: As we are using sql image so, we need to provide it the username, password and database name. Therefore we are creating an env file

syntax: **touch abc.env**

A screenshot of a Linux desktop environment showing a text editor window titled 'ab.env'. The window has a menu bar with 'File', 'Edit', 'Search', 'Options', and 'Help'. The content of the file is as follows:

```
MYSQL_ROOT_PASSWORD=my-secret-pw
MYSQL_DATABASE=nginxdb
MYSQL_USER=root
```

The desktop background is dark, and the top panel shows the date and time as 'Tue Feb 23 10:59 PM' along with system icons for network, volume, and battery (68%).

Step4: Now to run the compose file type the commands

docker-compose up -d/-it

Step5: Do `docker ps` to check the running containers

```
root@anonymous: ~/Music
root@anonymous:~/Music# touch docker-compose.yml
root@anonymous:~/Music# leafpad docker-compose.yml
root@anonymous:~/Music# touch ab.env
root@anonymous:~/Music# leafpad ab.env
root@anonymous:~/Music# docker-compose up -d
Building with native build. Learn about native build in Compose here: https://docs.docker.com/go/compose-native-build/
Pulling container1 (mysql:...)
latest: Pulling from library/mysql
45b42c59be33: Pull complete
b4f790bd91da: Pull complete
325ae51788e9: Pull complete
adcb9439d751: Pull complete
174c7fe16c78: Pull complete
698058ef136c: Pull complete
4690143a669e: Pull complete
f7599a246fd6: Pull complete
35a55bf0c196: Pull complete
790ac54f4c47: Pull complete
18602acc97e1: Pull complete
365caa3500d0: Pull complete
Digest: sha256:b1cc887ed32cc6c2f217b12703bd05f503f2037892c8bb226047fe5dff85a109
Status: Downloaded newer image for mysql:latest
Pulling container2 (nginx:...)
latest: Pulling from library/nginx
45b42c59be33: Already exists
8acc495fd91: Pull complete
ec3bd7de90d7: Pull complete
19e2441aeeab: Pull complete
f5a38c5f8d4e: Pull complete
83500d851118: Pull complete
Digest: sha256:f3693fe50d5b1df1ecd315d54813a77afd56b0245a404055a946574deb6b34fc
Status: Downloaded newer image for nginx:latest
Creating music_container1_1 ... done
Creating music_container2_1 ... done
root@anonymous:~/Music# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
d7590d0e6f17	nginx	"/docker-entrypoint..."	2 minutes ago	Up About a minute	0.0.0.0:80->80/tcp	music_container2_1
c52c026e6ecc	mysql	"docker-entrypoint.s..."	2 minutes ago	Up 2 minutes	33060/tcp, 0.0.0.0:3307->3306/tcp	music_container1_1

```
root@anonymous:~/Music#
```