Chapter 3.1 : Docker-Compose

==========

What is docker-compose ?

- Docker Compose is a tool that allows you to define and run multi-container Docker applications.
- It uses a YAML file to configure the application's services, networks, and volumes
- uses a single command to create and start all of the services
- It is a useful tool for setting up and managing complex containerized applications, as it allows you to easily define the different components of your application and how they should be configured.

Benefits of docker-compose ?

- **Simplicity**: Docker Compose allows you to define and start all of the services for your application with a single command. This makes it easy to set up and manage complex containerized applications.
- **Reusability**: You can create a single Compose file that can be used to start your application on any environment that supports Docker. This can help improve consistency across development, testing, and production environments.
- **Isolation**: Each service runs in its own container, providing a high degree of isolation between the different components of your application.
- **Scalability**: Services can be scaled individually, you can bring new instances of a service up or take them down without affecting the rest of the application.
- **Version control**: since the configuration is defined in a file, it is easy to store the file in version control and rollback/update easily.
- **Portability**: Docker Compose works on Windows, Linux, and macOS, which allows for greater portability of applications across different operating systems.
- **Networking** : With Docker Compose, you can create networks and attach services to those networks, allowing you to define how the different components of your application communicate with each other.

Docker-compose file format and naming ?

- The Compose file is a YAML file that defines your application's services, networks, and volumes.
- It must be named `docker-compose.yml` or `docker-compose.yaml`, and should be placed in the root of the project.

Chapter 3.1 : Docker-Compose

==========

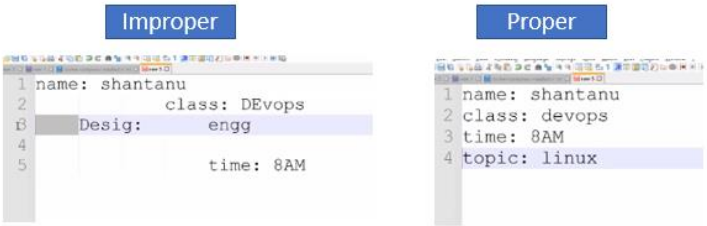
- docker compose for running multiple containers in yaml language
- consist of key=value

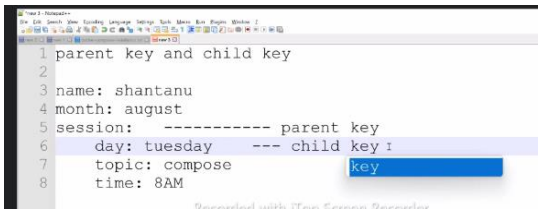
  name : vishal

  class : devops

  time : 9:00 AM

- Indentation is very important





- only spaces allowed no tabs.
- Sections are represented by hyphen.
- Service:- A service definition contains configuration that is applied to each container started for that service, much like passing command-line parameters to docker run.
- In simple words collection of everything that i want to tell docker-compose to create a container or multiple containers i.e what is the image you want how many containers etc.
- Here in compose we are not going to refer as a container but as a service.
- Docker compose will not create container it will create service, service will include image, containers, ports etc.
- Compose handles everything as a service

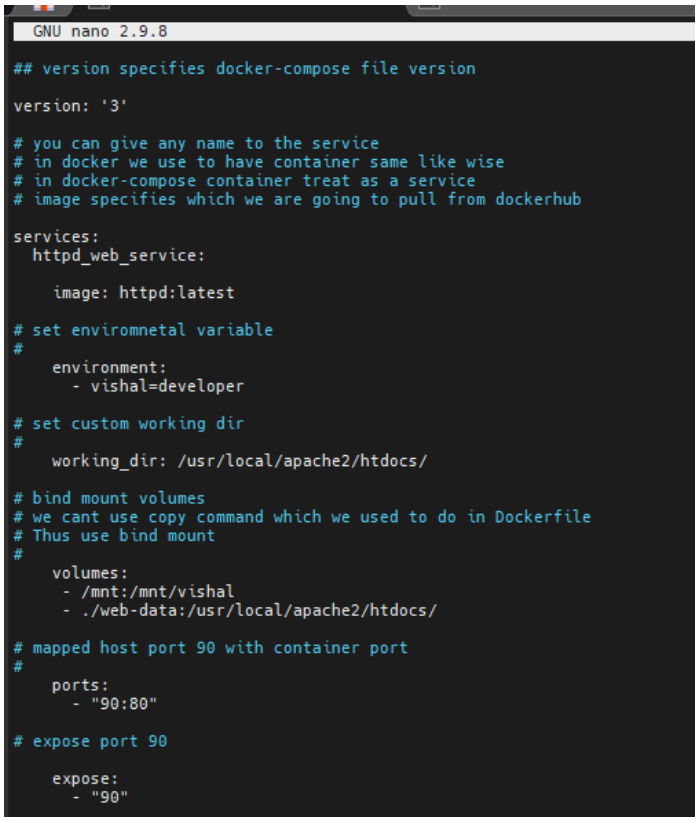
Chapter 3.1 : Docker-Compose

==========

Sample docker-compose.yml with explanation ?

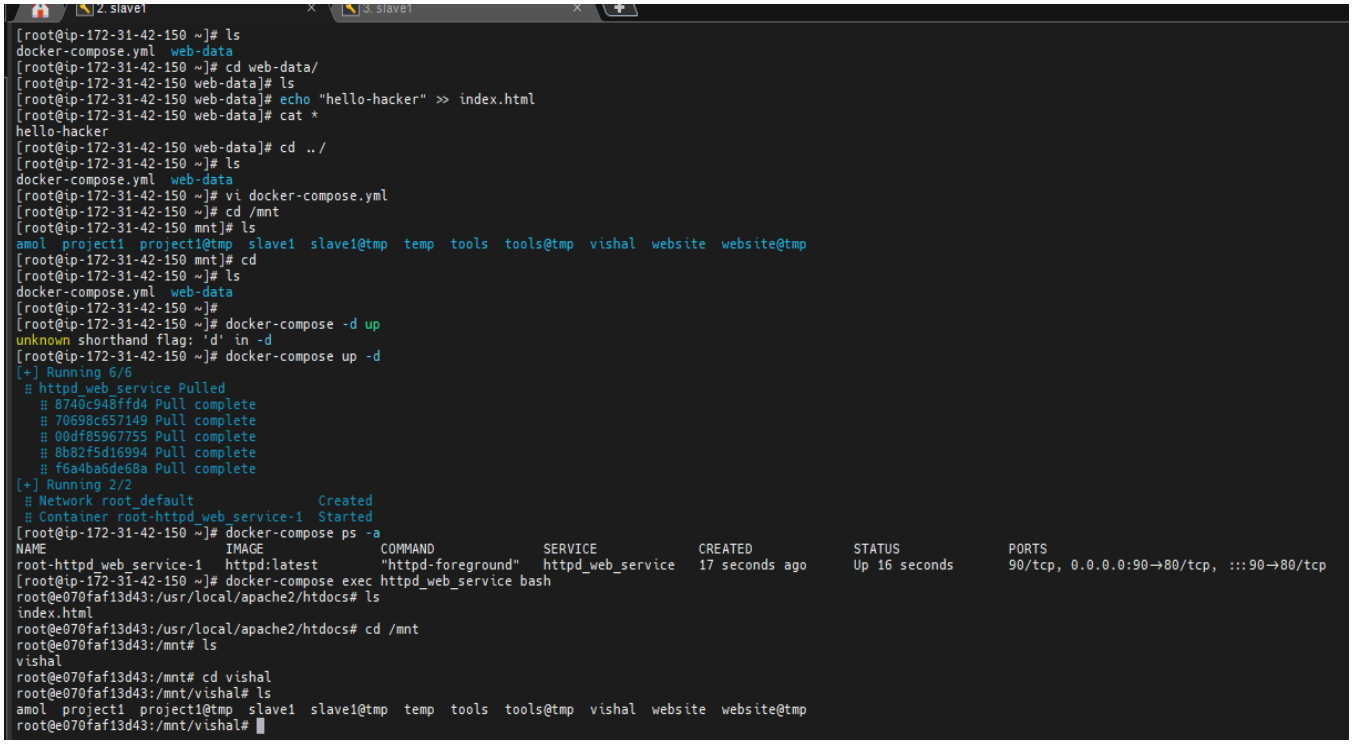
Here I m just adding all things in format in order to understand further

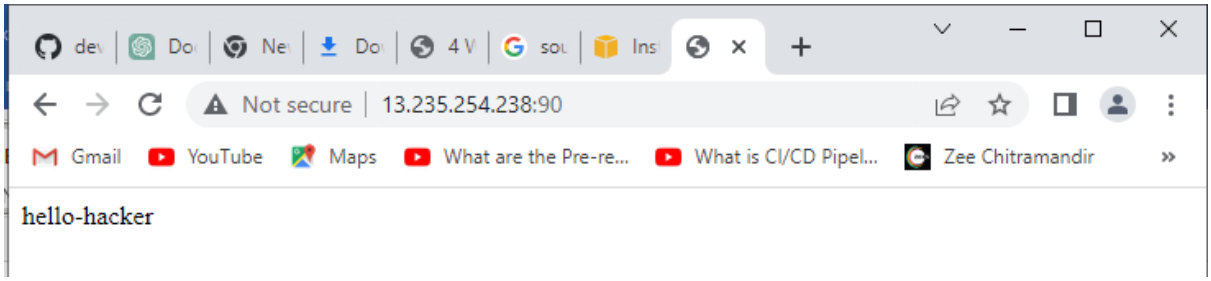
```
version: '3'
services:
  httpd_web_service:
    image: httpd:latest
    environment:
      - vishal=developer

    working_dir: /usr/local/apache2/htdocs/
    volumes:
     - /mnt:/mnt/vishal
     - ./web-data:/usr/local/apache2/htdocs/

    ports:
      - "90:80"

    expose:
      - "90"
```



Chapter 3.1 : Docker-Compose

==========





Ok Lets come the point ,

Docker-compose commands ??

- `docker-compose up`: starts the services defined in the `docker-compose.yml` file. You can add the `-d` option to run the services in detached mode.
- `docker-compose down`: stops the services defined in the `docker-compose.yml` file and removes the associated containers, networks, and volumes.
- `docker-compose start service-name`: starts the stopped service.
- `docker-compose stop service-name`: stops the running service.
- `docker-compose restart service-name`: restarts the running service.
- `docker-compose ps`: shows the status of the services defined in the `docker-compose.yml` file.
- `docker-compose logs`: shows the logs of the running services.
- `docker-compose exec`: runs a command in a running container. For example, `docker-compose exec service-name bash` runs the `bash` command

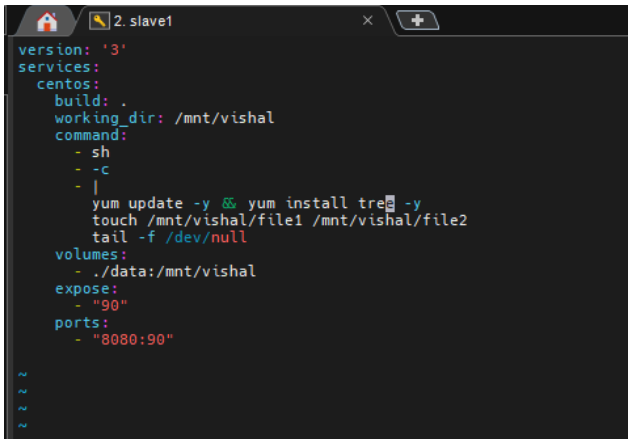
Chapter 3.1 : Docker-Compose

==========

Q. docker-compose.yml and Dockerfile combination.

vi Dockerfile

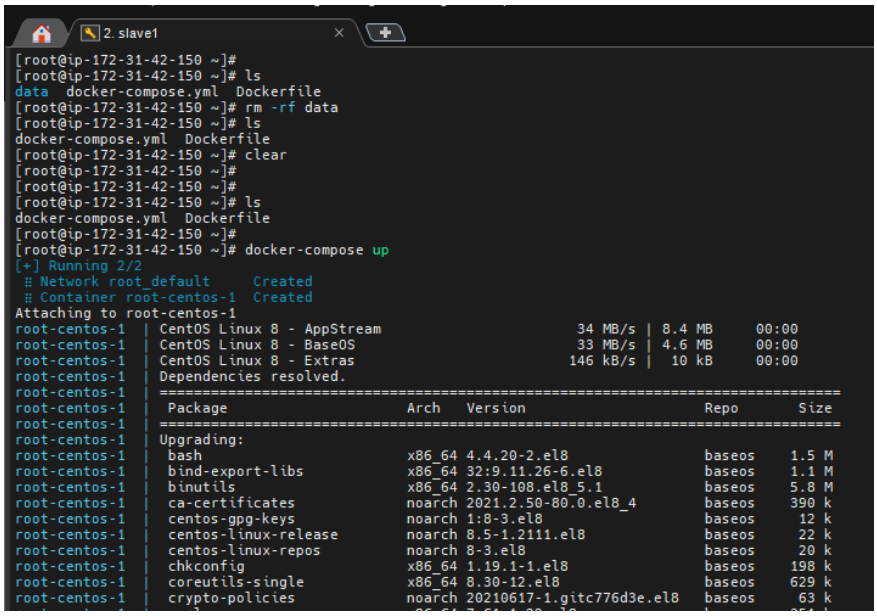
Vi docker-compose.yml



In docker file , base image = centos , update pakages repos

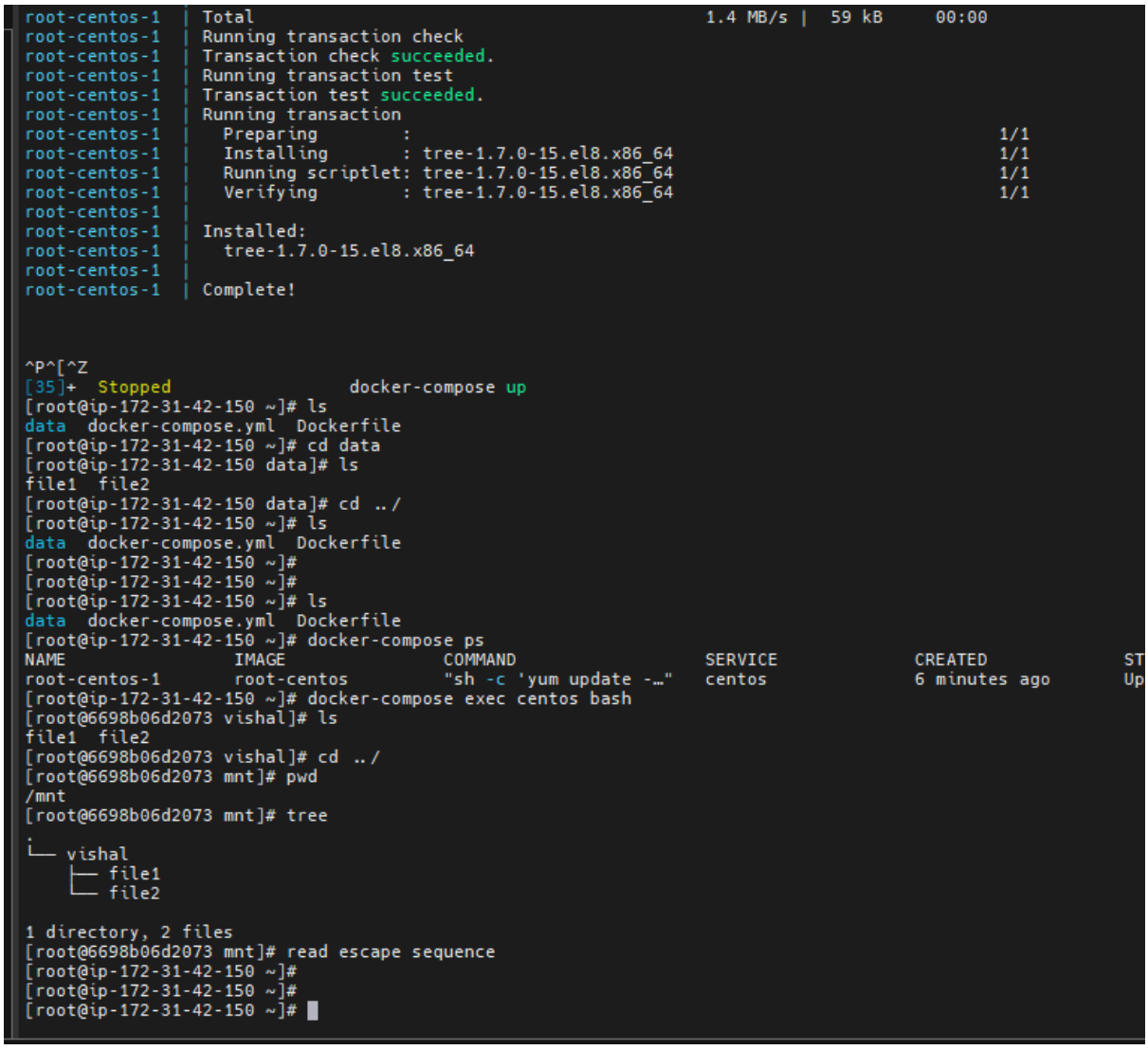
In docker-compose.yml , set working dir , create file1 file2 , create and bind mount volume with host /data, expose 90 , mapped host port 8080 with container port 90

Alternatively why tail -f /dev/null : it is because to keep container running continuously by keep executing this one , if I don't do that then container will exit after completing all the task



Chapter 3.1 : Docker-Compose

==========

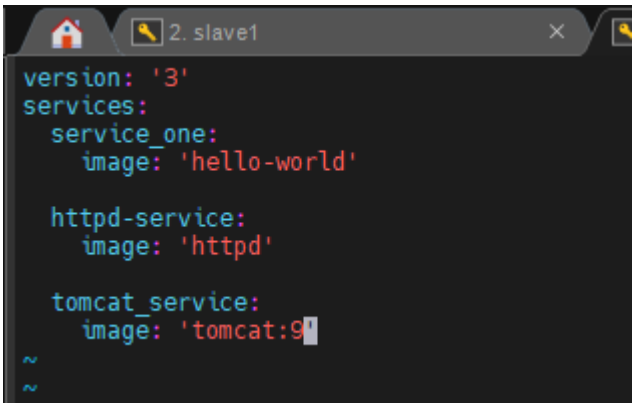


Chapter 3.1 : Docker-Compose

==========

Q. running multiple services/containers

vi docker-compose.yml




```
[root@ip-172-31-42-150 ~]# ls
combination-docker-compose-file
[root@ip-172-31-42-150 ~]# vi docker-compose.yml

[1]+  Stopped                 vi docker-compose.yml
[root@ip-172-31-42-150 ~]# ls
combination-docker-compose-file
[root@ip-172-31-42-150 ~]#
[root@ip-172-31-42-150 ~]#
[root@ip-172-31-42-150 ~]# vi docker-compose.yml
[root@ip-172-31-42-150 ~]# ls
combination-docker-compose-file  docker-compose.yml
[root@ip-172-31-42-150 ~]# docker-compose up
[+] Running 8/8
 # service_one Pulled
   # 2db29710123e Pull complete
 # httpd-service Pulled
   # 8740c948ffd4 Pull complete
   # 70698c657149 Pull complete
   # 00df85967755 Pull complete
   # 8b82f5d16994 Pull complete
   # f6a4ba6de68a Pull complete
WARN[0008] Found orphan containers ([root-centos-1]) for this project. If you removed or renamed
--remove-orphans flag to clean it up.
[+] Running 3/3
 # Container root-service_one-1     Created
 # Container root-httpd-service-1   Created
 # Container root-tomcat_service-1  Created
Attaching to root-httpd-service-1, root-service_one-1, root-tomcat_service-1
root-tomcat_service-1  | NOTE: Picked up JDK_JAVA_OPTIONS:  --add-opens=java.base/java.lang=ALL-
--add-opens=java.base/java.util.concurrent=ALL-UNNAMED --add-opens=java.rmi/sun.rmi.transport=AL
root-service_one-1     |
root-service_one-1     | Hello from Docker!
root-service_one-1     | This message shows that your installation appears to be working correct
root-service_one-1     |
root-service_one-1     | To generate this message, Docker took the following steps:
root-service_one-1     |  1. The Docker client contacted the Docker daemon.
root-service_one-1     |  2. The Docker daemon pulled the "hello-world" image from the Docker Hu
root-service_one-1     |     (amd64)
root-service_one-1     |  3. The Docker daemon created a new container from that image which run
root-service_one-1     |     executable that produces the output you are currently reading.
root-service_one-1     |  4. The Docker daemon streamed that output to the Docker client, which
root-service_one-1     |     to your terminal.
```

Chapter 3.1 : Docker-Compose

==========