Docker Compose is a tool for defining and running multi-container Docker applications. It allows you to define your application's services, networks, and volumes in a single docker-compose.yml file, and then start and stop all the services from this file.

With Docker Compose, you can easily define and manage the configuration of your application's services, which can be containers, networks, and volumes. You can specify the dependencies between services, such as which services depend on which other services.

Docker Compose also provides a convenient way to manage your application's lifecycle, including starting and stopping services, scaling services up and down, and viewing logs.

In summary, Docker Compose simplifies the process of building, testing, and deploying multi-container applications by allowing you to manage the entire application stack in a single file.

Basic docker compose.yaml file:

version: 'X' # version of the Compose file format

services:

service\_name\_1:

image: image\_name\_1 # name of the Docker image

ports:

- "host\_port\_1:container\_port\_1" # mapping of host port to container port

environment:

- ENV\_VAR\_1=value\_1

- ENV\_VAR\_2=value\_2

service\_name\_2:

image: image\_name\_2

ports:

- "host\_port\_2:container\_port\_2"

environment:

- ENV\_VAR\_3=value\_3

- ENV\_VAR\_4=value\_4

networks:

network\_name\_1:

driver: driver\_name # name of the network driver to use

volumes:

volume\_name\_1:

driver: driver\_name # name of the volume driver to use

Each service defined in the services section represents a separate container that will run in your application. The image field specifies the name of the Docker image that should be used to create the container. The ports field is used to map a host port to a container port. The environment field is used to set environment variables for the container.

The networks section defines the custom networks that can be used by your application's services. The driver field specifies the network driver to use.

The volumes section defines the custom volumes that can be used by your application's services. The driver field specifies the volume driver to use.

Guna Vardhan

docker compose basic commands

Here are some basic commands you can use with Docker Compose:

docker-compose up: Start the services defined in the docker-compose.yml file. This command will create the containers, networks, and volumes defined in the file, and start the services. If the containers, networks, or volumes already exist, the up command will simply start the services.

docker-compose down: Stop the services and remove the containers, networks, and volumes defined in the docker-compose.yml file. This command will stop the services and remove all resources created by the docker-compose up command.

docker-compose ps: List the containers created by the docker-compose up command, along with their status, ports, and names.

docker-compose logs: View the logs for the containers created by the docker-compose up command.

docker-compose restart: Restart the services defined in the docker-compose.yml file. This command will stop the services and start them again.

docker-compose stop: Stop the services defined in the docker-compose.yml file without removing the containers, networks, and volumes.

docker-compose build: Build the services defined in the docker-compose.yml file. This command will rebuild the images for the services.

docker-compose exec: Execute a command in a running container. For example, docker-compose exec service\_name\_1 bash will open a bash shell in the container for service service\_name\_1.