Q. What is Docker?

Docker is one of the most popular tools for application containerization. Docker enables efficiency and reduces operational overheads so that any developer, in any dev environment, can build stable and reliable applications.

Q. Why do we use Docker?

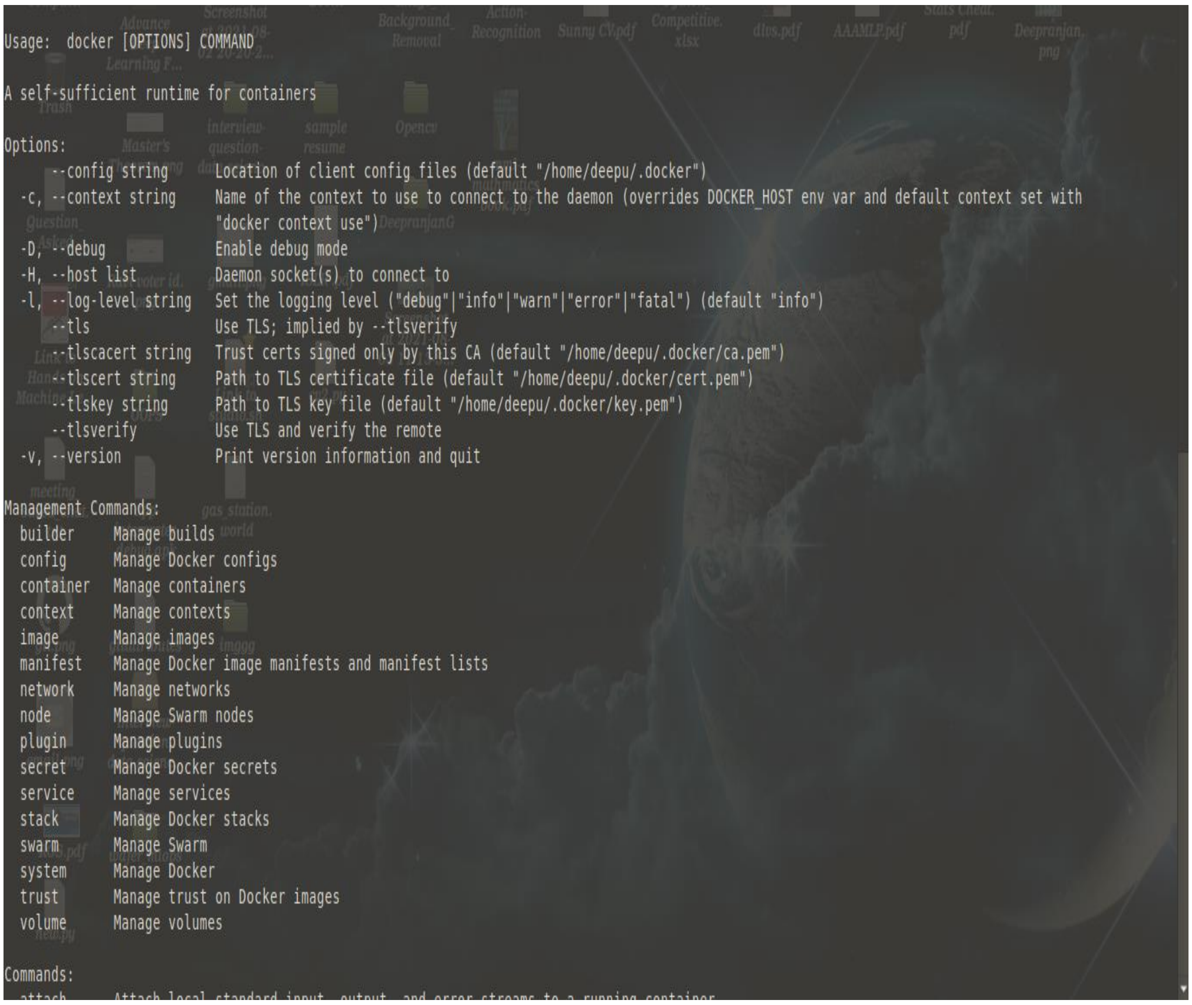
Developers can create containers without Docker, but the platform makes it easier, simpler, and safer to build, deploy and manage containers. Docker is essentially a toolkit that enables developers to build, deploy, run, update, and stop containers using simple commands and work-saving automation through a single API.

Docker Installation

[Docker Installation Video Ubuntu]

Steps:

1. Open terminal and type ***docker*** if it is installed will get output like that



```
Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Options:
      --config string      Location of client config files (default "/home/deepu/.docker")
  -c, --context string     Name of the context to use to connect to the daemon (overrides DOCKER_HOST env var and default context set with
                           "docker context use")
  -D, --debug              Enable debug mode
  -H, --host list          Daemon socket(s) to connect to
  -l, --log-level string   Set the logging level ("debug"|"info"|"warn"|"error"|"fatal") (default "info")
      --tls                Use TLS; implied by --tlsverify
      --tlscacert string   Trust certs signed only by this CA (default "/home/deepu/.docker/ca.pem")
      --tlscert string     Path to TLS certificate file (default "/home/deepu/.docker/cert.pem")
      --tlskey string      Path to TLS key file (default "/home/deepu/.docker/key.pem")
      --tlsverify          Use TLS and verify the remote
  -v, --version            Print version information and quit

Management Commands:
  builder     Manage builds
  config      Manage Docker configs
  container   Manage containers
  context     Manage contexts
  image       Manage images
  manifest    Manage Docker image manifests and manifest lists
  network     Manage networks
  node        Manage Swarm nodes
  plugin      Manage plugins
  secret      Manage Docker secrets
  service     Manage services
  stack       Manage Docker stacks
  swarm       Manage Swarm
  system      Manage Docker
  trust       Manage trust on Docker images
  volume      Manage volumes

Commands:
  attach      Attach local standard input, output, and error streams to a running container
```

2. Check is your project running properly.
3. Go Inside project location, Create ***Dockerfile*** (a file with name Dockerfile without extension)
4. Inside that Dockerfile write these lines:

```
FROM python:3.7.5-slim

RUN apt-get update -y && \

   apt-get install -y python-pip python-dev && \

   apt-get install -y build-essential cmake && \

   apt-get install -y libopenblas-dev liblapack-dev && \

   apt-get install -y libx11-dev libgtk-3-dev

COPY ./requirements.txt /requirements.txt

WORKDIR /

RUN pip3 install -r requirements.txt

COPY . /

ENTRYPOINT [ "python3" ]

CMD [ "clientApp.py" ]
```

5. Do changes in CMD according to your project
6. After that open terminal inside that folder.
7. Type ***sudo docker build -t fash:latest .*** (do changes in fash:latest according to your project these are just name. Keep it according to your project name .
8. Hit enter and wait for the installation it will take time depends on requirements file and internet connection as well.
9. Once it is done type ***sudo docker images*** in the terminal it will show all docker images present.
10. Now run that docker images, type ***sudo docker run --name fashiontest -p 8000:5000 -rm fash:latest*** (8000 is the port number which we want to run, 5000 is the port number that we have mention during project implementation)
11. To Delete any image, type ***sudo docker rmi --force 5733500520cf*** (5733500520cf is the id of image)
12.