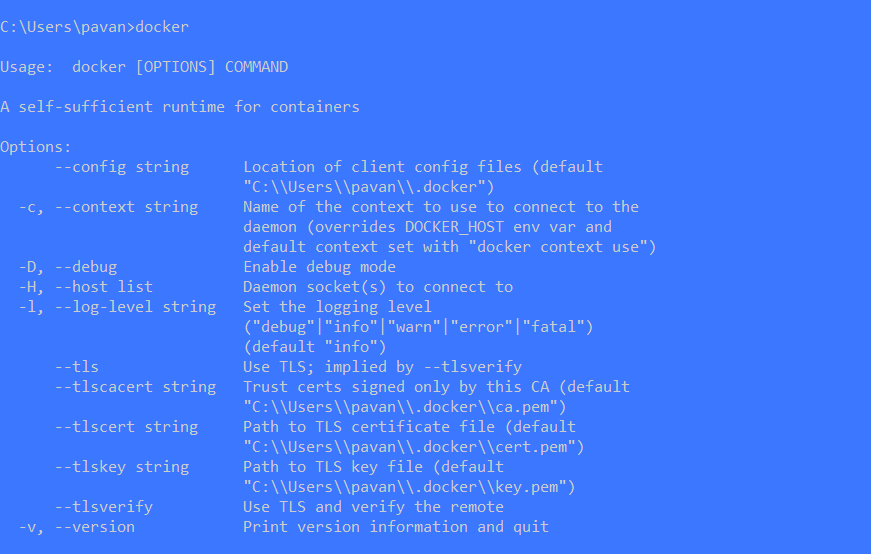
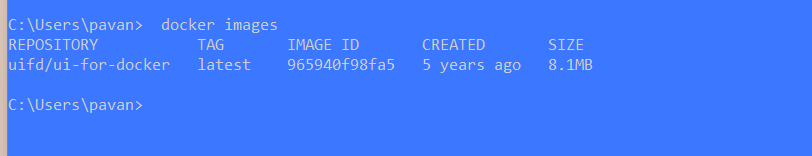
Docker

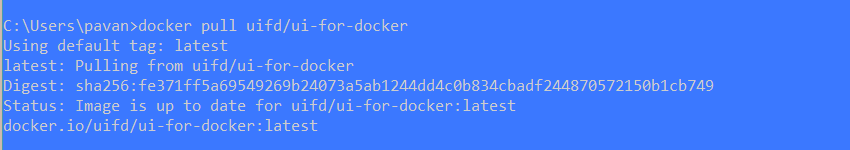


docker images // shows all the images present in the docker

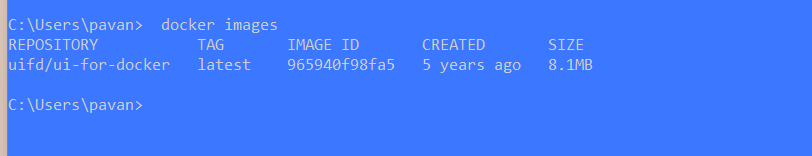


docker pull uifd/ui-for-docker //Command to pull images from the Docker hub

docker pull (image\_name)



docker images //list of images in the docker engine

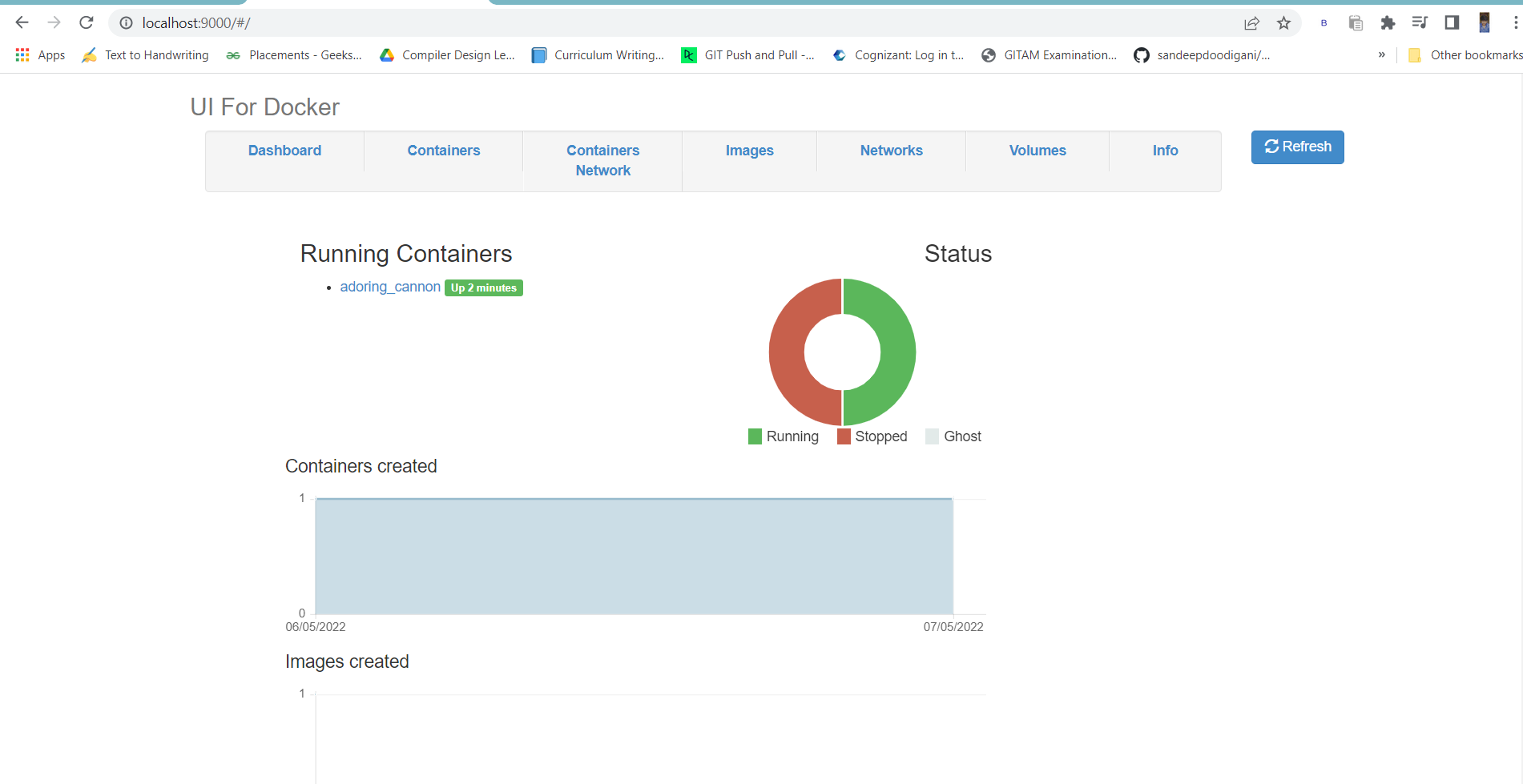


docker run -d -p 9000:9000 --privileged -v /var/run/docker.sock:/var/run/docker.sock uifd/ui-for-docker

docker ps //docker process state, can see what applications

are running in the docker engine

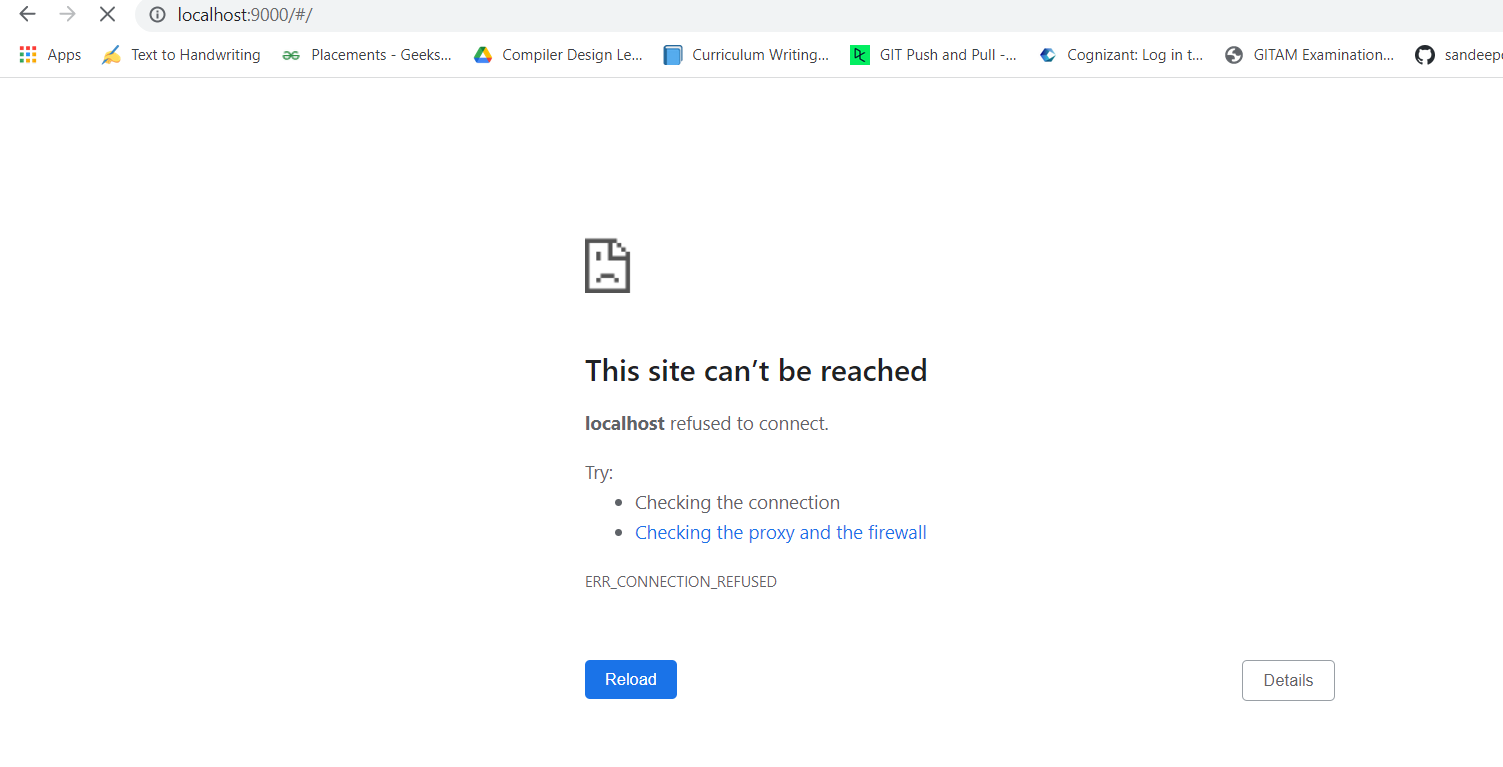
docker run -p 9000:9000 imagename //running a docker image in local host 9000

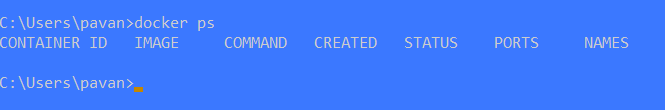


\*\*\*Once the application is running, it is called as a container\*\*\*

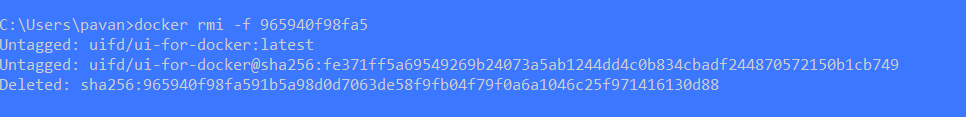
\*\*\* before that it is called as an image\*\*\*

docker stop 660488b34d69 // Stopping the container(conatiner id is used to stop)





docker rmi -f 965940f98fa5 //Deleting the image (rmi is remove image,-f is force)





Now process to convert a application into Docker Image and uplod to Docker HUB

1. We need to navigate the folder location

cd ..

cd ..

c:

cd DevOps

cd Plasma-Docker-main



FROM python:3.6.5-alpine //installing python in the OS Alpine

WORKDIR /app //Creates an file in Alphine and copy all the files

ADD . /app

COPY requirements.txt /app

RUN pip install -r requirements.txt //Libraries required for installing flask

CMD ["python","app.py"]

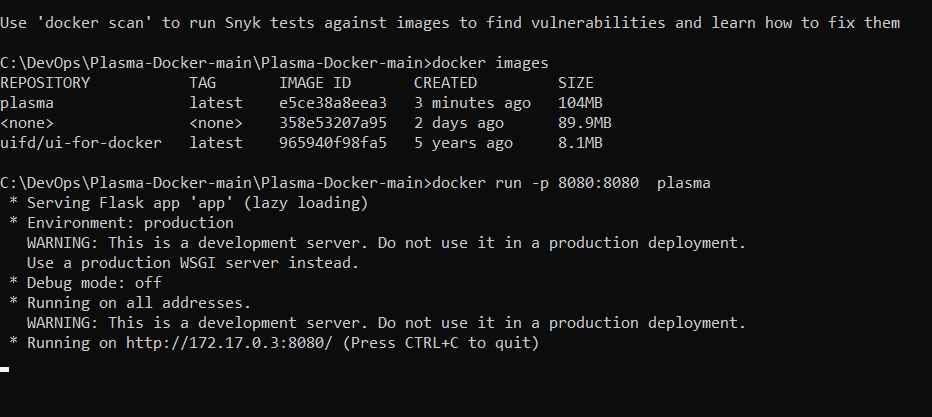
\*\*\*Building the image\*\*\*

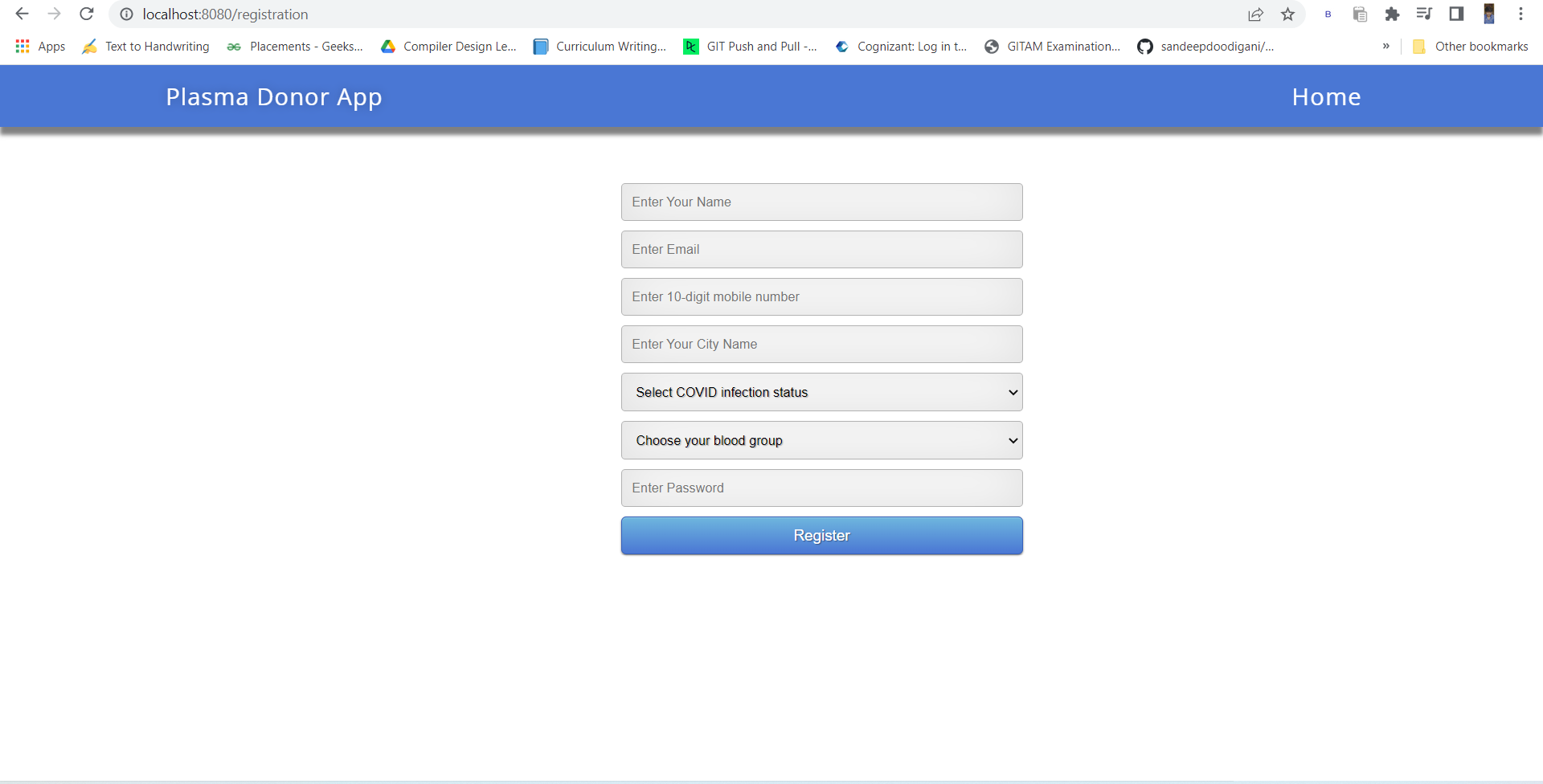
docker build . -f Dockerfile.txt //Navigating to the file

docker build -t plasma .

docker images

docker run -p 8080:8080 plasma //running the file

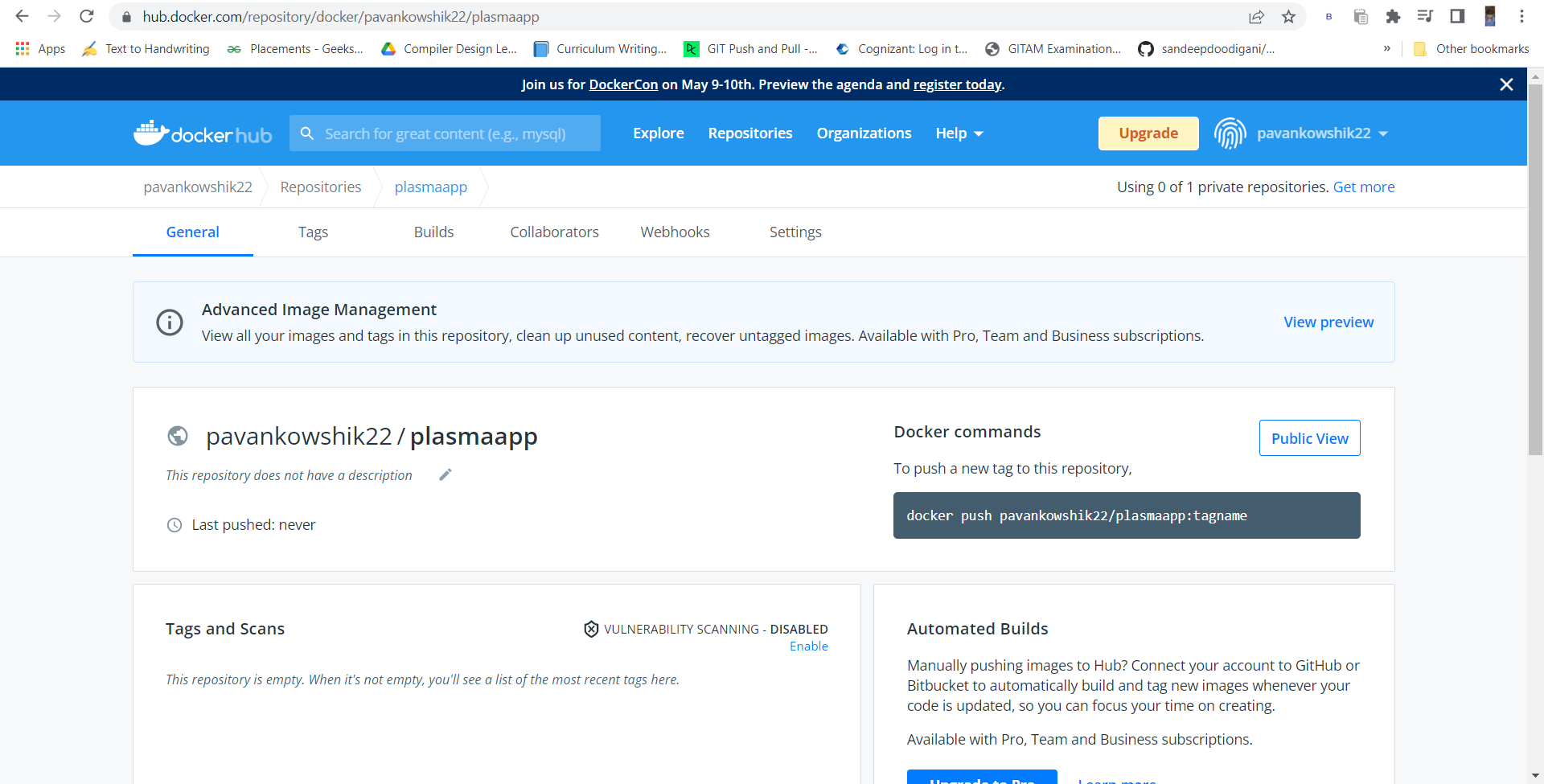




\*\*\*Making the Application live\*\*\*

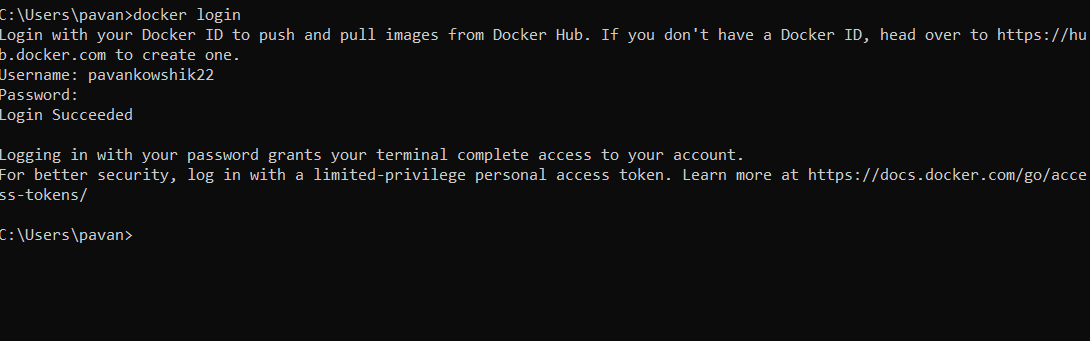
\*\*\*Now we are pushing the plasma App to Docker hub\*\*\*

1. Create a new repository in docker hub



1. Open command Prompt

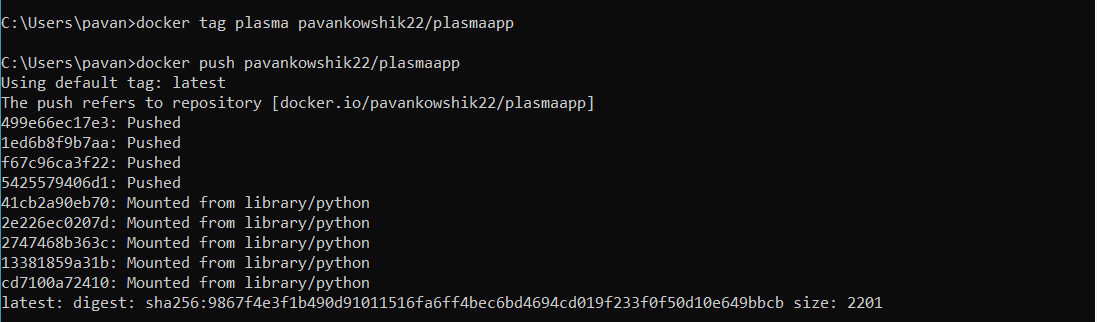
docker login



docker tag plasma pavankowshik22/plasmaapp //Making the local repository name with the

created ones in docker hub

docker push pavankowshik22/plasmaapp // Push all the files to Docker Hub



Now the images get uploded in the docker Hub

