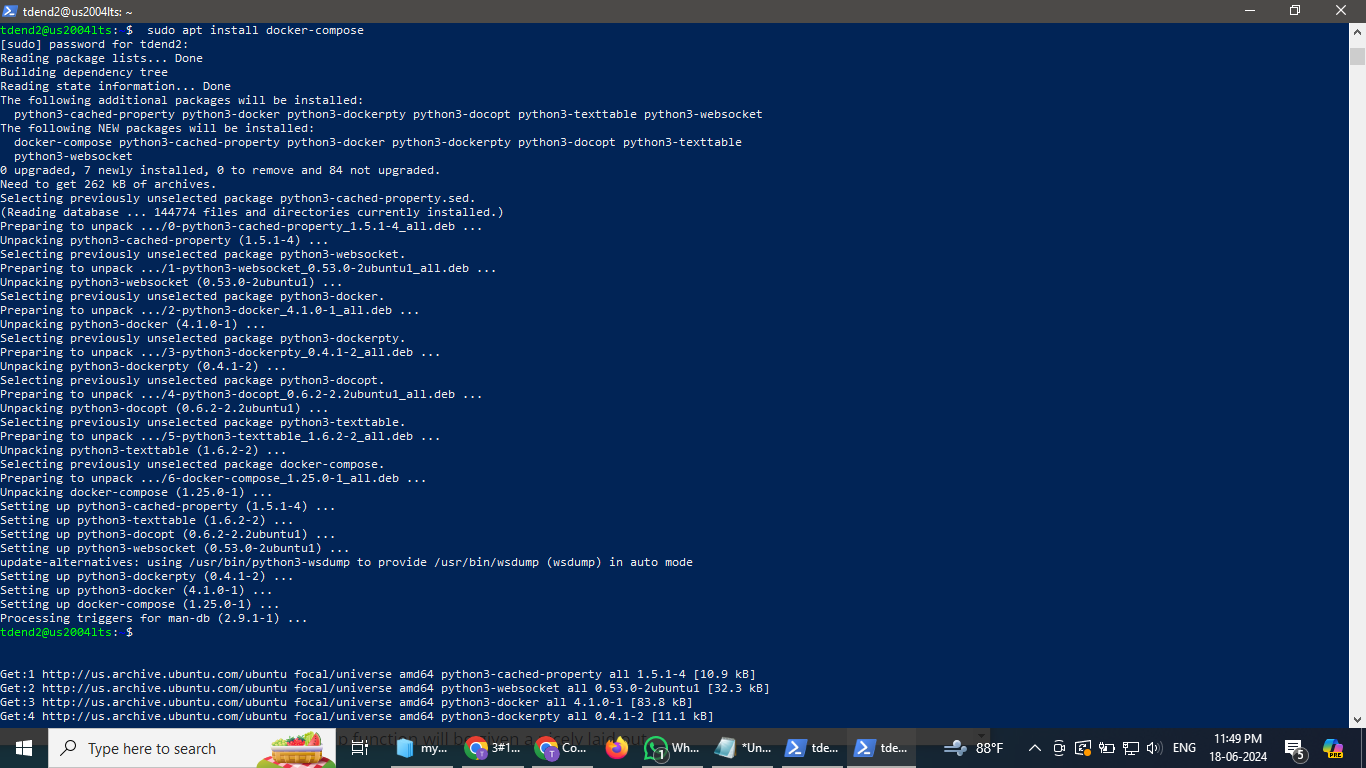
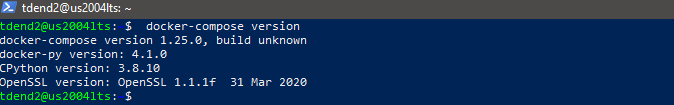
**Deploying Multi-Container Apps with Docker Compose**

• a. **Section 1.2** Installed Docker Composeas shown below.

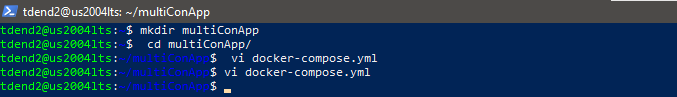




Verified the installation of docker compose as shown above.

b. **Section 1.3** Created Docker compose file for PostgreSQL and pgAdmin

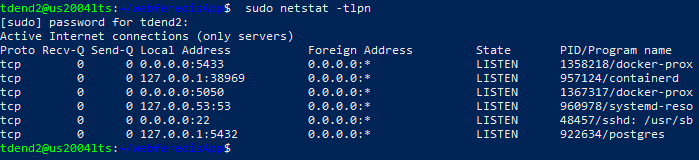
i. changed the e-mail and passwords in the file in the pgadmin service of yml file.



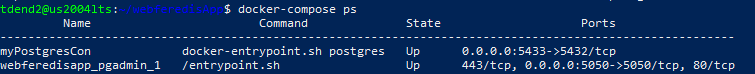
c. **Section 1.4** Running PostgreSQL and pgAdmin



Verification of ports on which these are running:



*ports 5050 and 5433(port mapping:5432) are opened by Docker.*



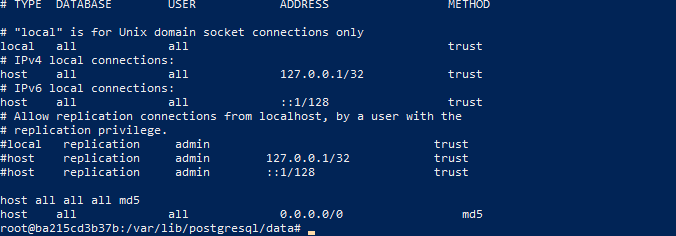
For the pgadmin service, Docker host port 5050 is mapped to the container TCP port 5050. For the postgresql service, Docker host port 5433 is mapped to the container TCP port 5432.

d. Section 1.5 Accessing PostgreSQL DB from pgAdmin

Added:

***echo "host all all 0.0.0.0/0 md5" >> pg\_hba.conf***

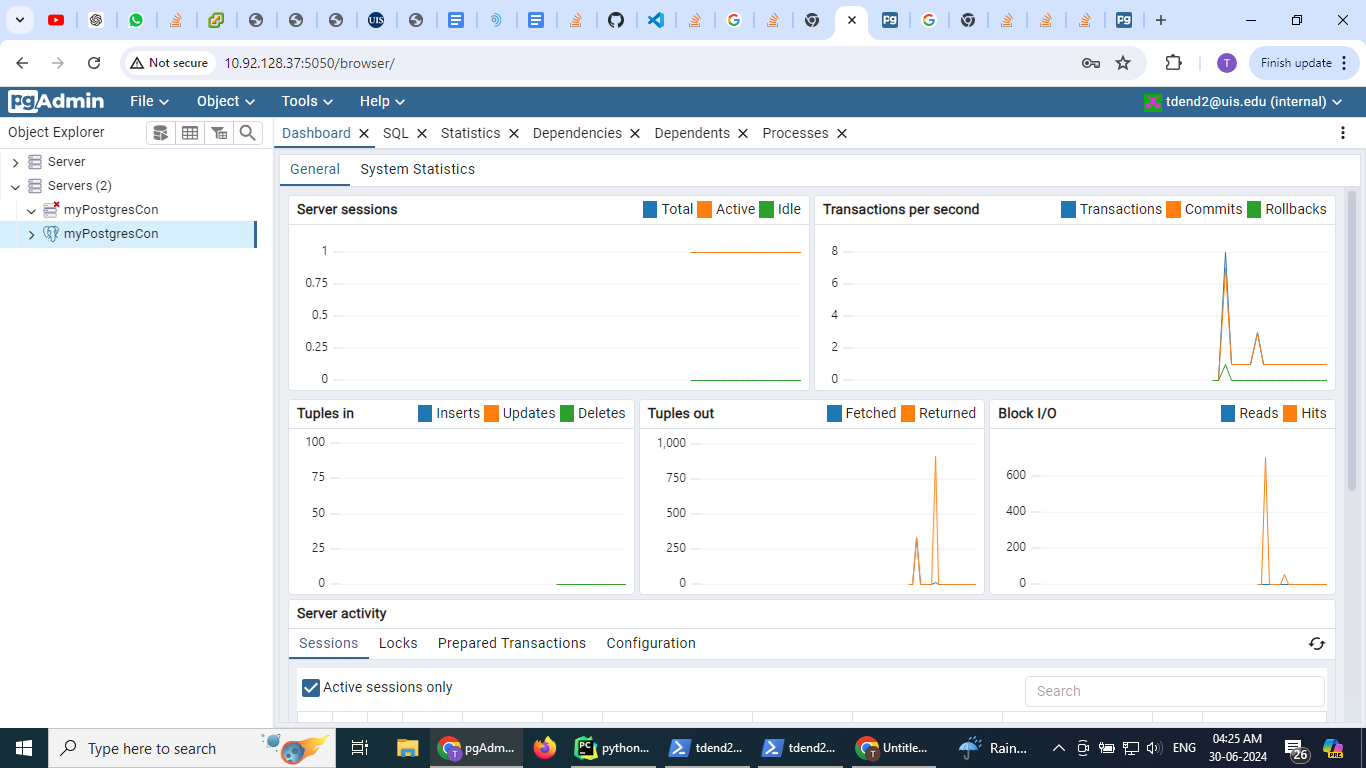
to pg\_hba.conf to ensure ensure that the pg\_hba.conf file of the server from which you are connecting allows connections from the host of the client.



In the environment of pgadmin in docker-compose.yml ,it was ensured to connect to IPV4 address by adding PGADMIN\_LISTEN\_ADDRESS:0.0.0.0

Also made sure postgresql is in active state.

At <http://10.92.128.37:5050>/browser/:accessed postgresql service from pgadmin service:



• Add Portainer in your app, multiConAppHW (PostgreSQL + pgAdmin +

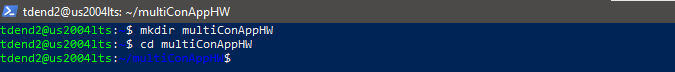
Portainer).

a. Portainer is a powerful web-based Docker management tool. See

https://www.portainer.io/

b. Created a new compose file as shown below using ***vi docker-compose.yml*** in the vi editor with name docker-compose following the synatx/ structural format of yml(Yet Another Markup Language). Formatting plays a major role in yml files.

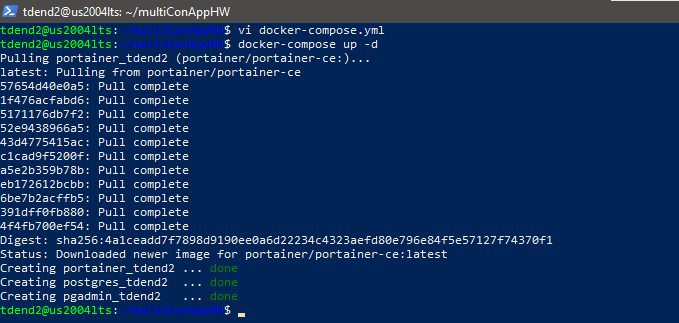
i. Create a new directory, multiConAppHW



ii. Under the directory, created a new compose file, docker-compose.yml,

based on the file for the multiConApp

iii. Named containers/services and added NetID as suffixes.



c. Add a service for Portainer in the compose file.

i. added container name, image, restart policy, environment,

network, volume, and/or etc. for the service as needed as follows:

*portainer\_tdend2:*

*image: portainer/portainer-ce*

*container\_name: portainer\_tdend2*

*restart: always*

*ports:*

*- "9000:9000"*

*volumes:*

*- /var/run/docker.sock:/var/run/docker.sock # Allows Portainer to connect to the Docker daemon*

*- portainer\_data:/data # Stores Portainer configuration data*

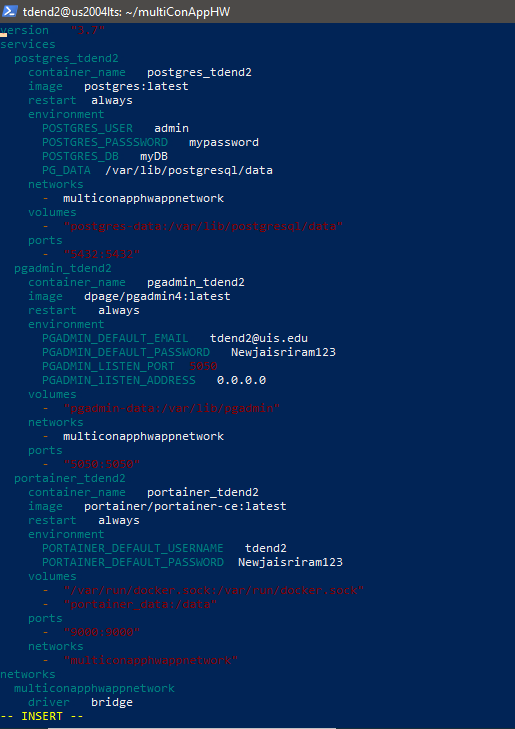
*networks:*

*- app-network*

d. Added top-level network(s) and/or volume(s) for the service as needed as shown in the compose yml file.

e. screenshot(s) of the

***compose file are shown below:***



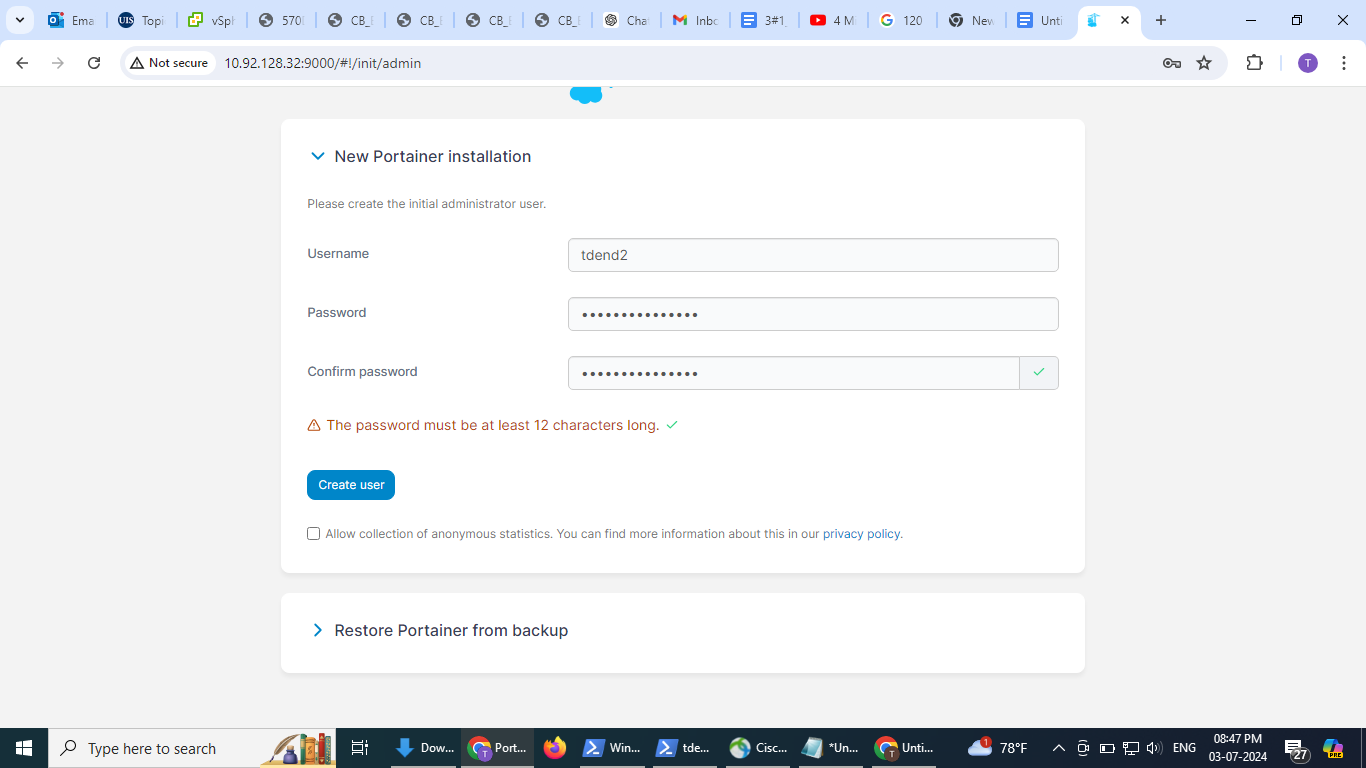
Continuation of ymlfile is below:



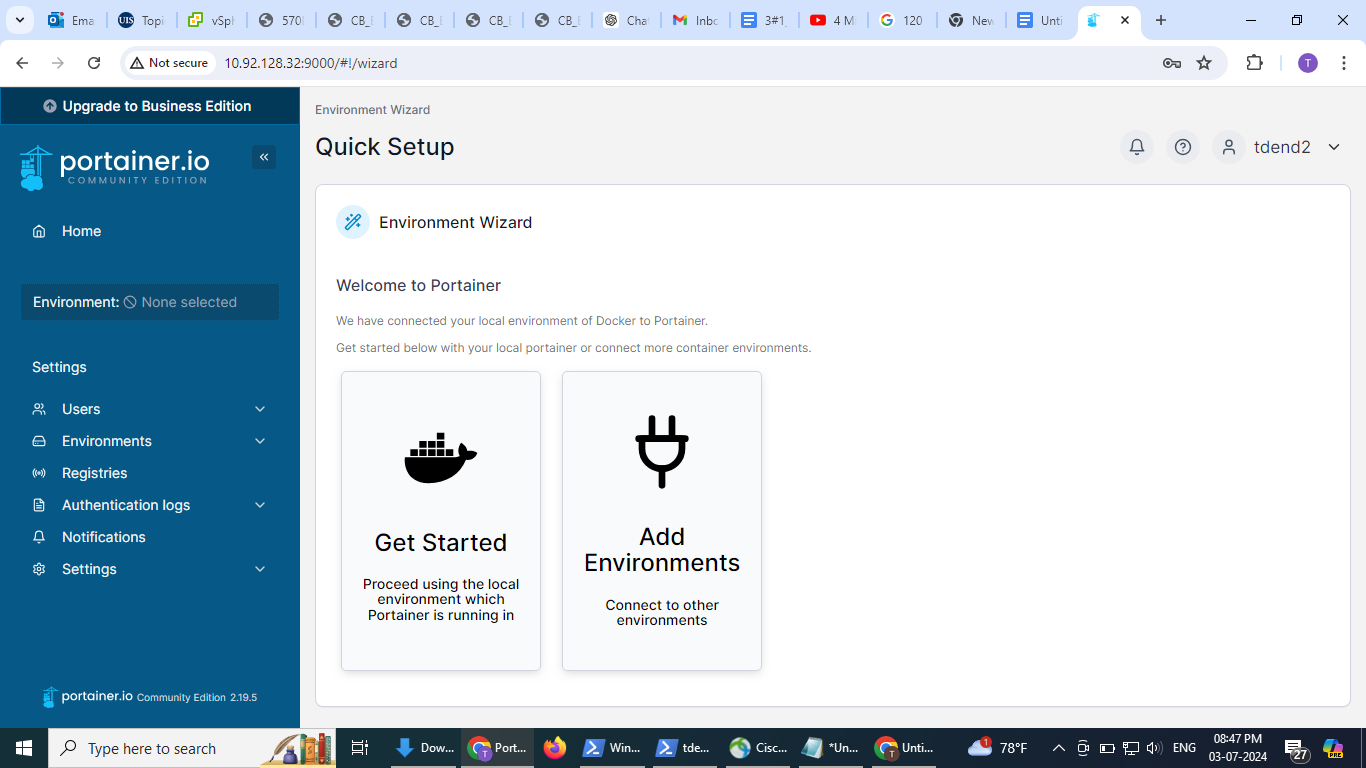
***running the app*** at [http://10.92.128.32:9000](http://10.92.128.32:9000/#!/2/docker/containers)

Where 10.92.128.32 is the IP address of VM and 9000 is the port on which portainer is running

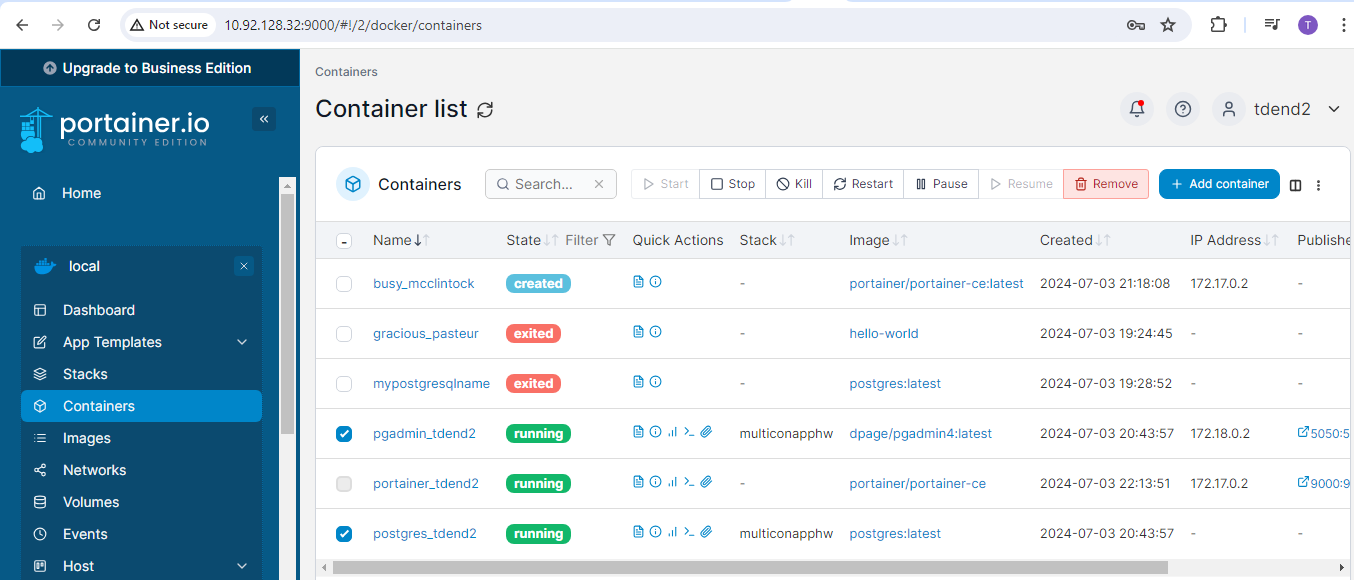
After the portainer is loaded on the above url, entered the username as tdend2 and password as given in the compose file as follows:



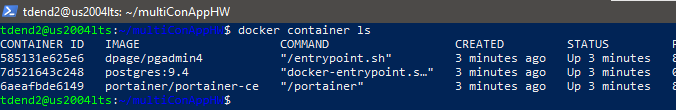
After logging at the portainer UI, it navigated to the next page as follows:

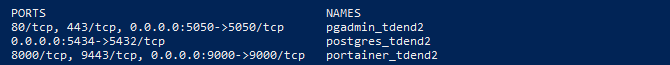


On clicking the home, the local dashboard navigation bar of the portainer popped up with containers as one of its menu. On clicking containers the list of containers are displayed as follows:



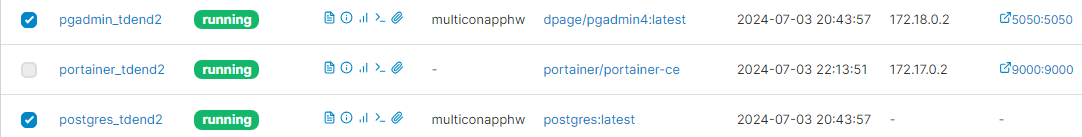
listing containers using ‘docker container ls’,





and Portainer showing three containers for the three services

(PostgreSQL + pgAdmin + Portainer); attached screenshot here below:



==================================THE END===============================