PROBLEM :

- Dynamic Allocation of Linux systems for users
- Each user should have independent Linux System
- Specific training environment should be created in Container
- User should not allow to access other containers/images
- User should not allow to access docker command
- Monitor participants containers
- Debug/live demo for the participants if they have any doubts/bug in running applications.
- Automate container creation and deletion.

Creating the container image:-

1. A new container must be created from base image.
Command: -sudo docker create it name docker_list ubuntu /bin/bash

2. Start the container
command: -sudo docker start my_container

3. Attach to the container
command: -sudo docker attach my_container

4. Install the required applications using the following commands

```
 -apt update
- apt install vim
- apt install gcc
```

5.Create questions.txt, instructions.txt and save them.

command: touch questions.txt

touch instructions.txt

6. Commit the container

command: docker commit a "Soumya" 37f609ba3b38

Allocating Containers To Users:

The shell script create Containers.s will automatically create a docker container for every user.

1.users.txt

```
soumya

tharun

vani
```

2.createContainers.sh

```
echo -n "Enter name of file with usernames: "
read file
while read user
    do
        docker create -it --name $user
        docker_class_image_2018 /bin/bash
    done < $file
```

1.Fill the entries in users.txt with usernames and run the shell script sh createContainers.sh -x. This creates a docker container corresponding to each username from users.txt.
2.The user can then start using the allocated container by doing the following

- Method 1

```
docker start <name> # Starts the container
docker attach <name> # Attach the container
```

However attaching to a container may not give the desired behaviour, so it might be better to start a new shell

- Method 2

```
docker start <name> # Starts the container
docker exec -it <name> /bin/bash
```

Monitoring participants container

• To monitor the user containers create a bash file monitor_containors.sh

• monitor_Containers.sh

```
-echo n "Enter username of container to be monitored: "
-read name
-docker logs f $name
```

Deleting The Containers

Automate the deletion using the deleteContainers.sh script.

<u>deleteContainers.sh</u>

```
echo -n "Enter name of file containing usernames: "
read file
while read user
    do
```

```
                docker stop $user
                docker rm $user
        done < $file
```

You can either delete all users or user by name using sh deleteContainers.sh -x.