**IT3037C**

**Lab#1 Docker for Virtualization**

In this lab, you will study Docker and how it can be used to virtualize not just servers, but also desktop and mobile device environments. Finally, you will learn how to deploy your own local cloud in your organization through Nextcloud.

**Tools and Materials**

1. Ubuntu Command line
2. Docker CLI
3. Docker Desktop

**Note: You need to replace words in italics with your information. Be sure to show your output.**

**Questions will be in red. *Please type all answers in Blue***

0. Installing Docker

1. In order to install docker on the VM you need to run the install script to simplify installation. To do so, open your Ubuntu command line and enter the following commands:  
  
curl -fsSL https://get.docker.com -o get-docker.sh

Then issue from the directory you installed Docker into:

sudo sh ./get-docker.sh

1. Building a Dockerfile from Scratch:
2. Open your Ubuntu command line and create a file called Dockerfile (NOTE: the “D” in Dockerfile must be capitalized!)

nano Dockerfile

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

FROM ubuntu:18.04

# Install dependencies

RUN apt-get update && \

apt-get -y install apache2

# Install apache and write hello world message

RUN echo 'Hello World, this is to test our apache container!' > /var/www/html/index.html

# Configure apache

RUN echo '. /etc/apache2/envvars' > /root/run\_apache.sh && \

echo 'mkdir -p /var/run/apache2' >> /root/run\_apache.sh && \

echo 'mkdir -p /var/lock/apache2' >> /root/run\_apache.sh && \

echo '/usr/sbin/apache2 -D FOREGROUND' >> /root/run\_apache.sh && \

chmod 755 /root/run\_apache.sh

EXPOSE 80

CMD /root/run\_apache.sh

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

1. After editing our Dockerfile, we can build it with the following command:

docker build -t class-test .

What did you see?

=> [internal] load build definition from Dockerfile 0.0s

=> => transferring dockerfile: 614B 0.0s

=> [internal] load metadata for docker.io/library/ubuntu:18.04 0.7s

=> [internal] load .dockerignore 0.0s

=> => transferring context: 2B 0.0s

=> [1/4] FROM docker.io/library/ubuntu:18.04@sha256:152dc042452c496007f07ca91275 1.7s

=> => resolve docker.io/library/ubuntu:18.04@sha256:152dc042452c496007f07ca91275 0.0s

=> => sha256:dca176c9663a7ba4c1f0e710986f5a25e672842963d95b960191e2d 424B / 424B 0.0s

=> => sha256:f9a80a55f492e823bf5d51f1bd5f87ea3eed1cb31788686aa99 2.30kB / 2.30kB 0.0s

=> => sha256:7c457f213c7634afb95a0fb2410a74b7b5bc0ba527033362c 25.69MB / 25.69MB 1.0s

=> => sha256:152dc042452c496007f07ca9127571cb9c29697f42acbfad723 1.33kB / 1.33kB 0.0s

=> => extracting sha256:7c457f213c7634afb95a0fb2410a74b7b5bc0ba527033362c240c7a1 0.7s

=> [2/4] RUN apt-get update && apt-get -y install apache2 9.5s

=> [3/4] RUN echo 'Hello World, this is to test our apache container!' > /var/ww 0.2s

=> [4/4] RUN echo '. /etc/apache2/envvars' > /root/run\_apache.sh && echo 'mkdir 0.1s

=> exporting to image 0.5s

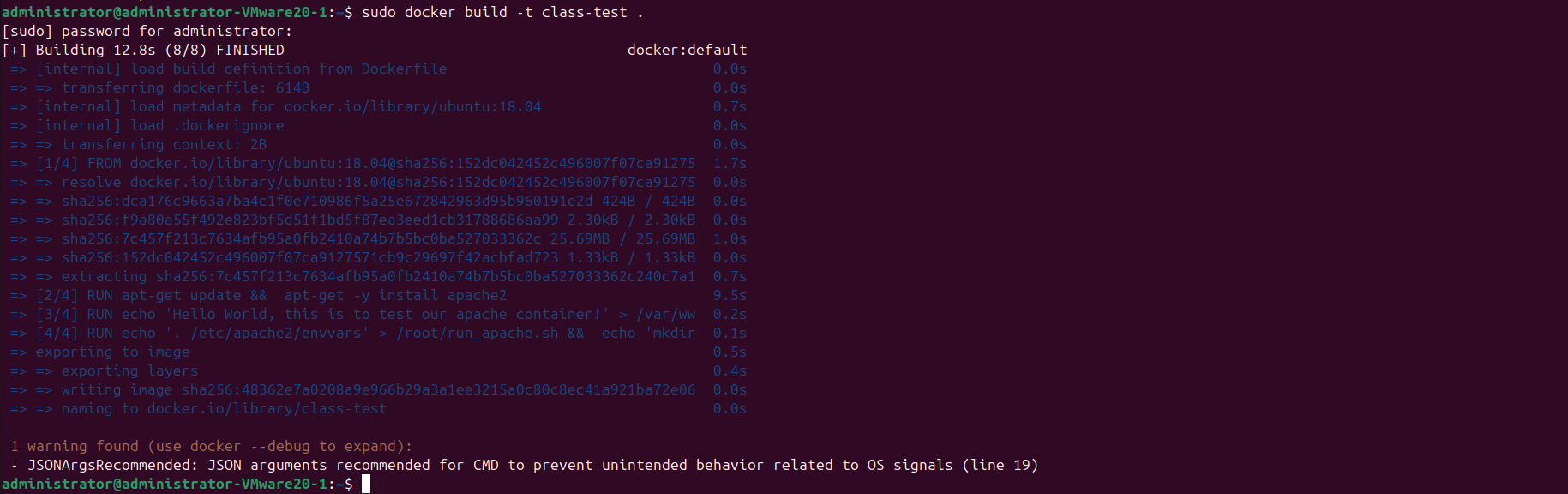
=> => exporting layers 0.4s

=> => writing image sha256:48362e7a0208a9e966b29a3a1ee3215a0c80c8ec41a921ba72e06 0.0s

=> => naming to docker.io/library/class-test 0.0s

1 warning found (use docker --debug to expand):

- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 19)



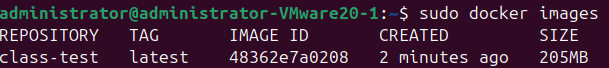
1. Enter the following to view your docker images:

docker images

What did you see?

REPOSITORY TAG IMAGE ID CREATED SIZE

class-test latest 48362e7a0208 2 minutes ago 205MB



1. After verifying the image is there we can run our container with the following command:

docker run -t -i -p 80:80 class-test

1. Navigate to your browser and enter the following :  
     
   localhost:80

or your devices ip like so:

<http://10.10.25.68/>

What did you see?

Hello World, this is to test our apache container!



1. This should then show you the web server index page you created

Press “ctrl + c” to stop the container from running

After creating your image, you can delete it by entering the following command

docker image rmi -f class-test   
  


~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

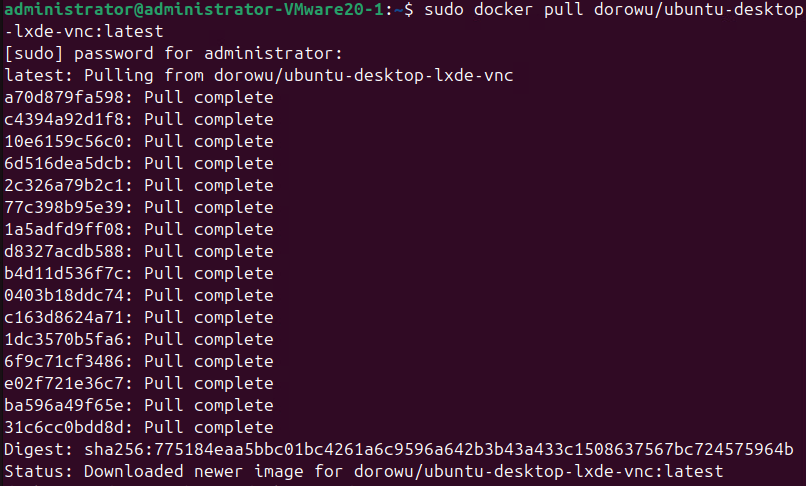
II. Test pulling a Docker Image and Running It

1. In order to pull a container, you can either use the docker pull command or the docker run command. We will start by pulling a Ubuntu Desktop Environment container.

docker pull dorowu/ubuntu-desktop-lxde-vnc:latest

Describe how this container could be used, why would you want a containerized Linux environment?

The reason for why you might want to have a container is a plethora of reasons. This container could be used for testing projects or software off of the local machine, it could



1. In order to run this container in the command line, we can use the docker run command below which will launch our ubuntu desktop container.

This command does as follows:

We set the hostname and mac-address of the container using the –hostname and –mac-address,

We set the environmental variables with ENV for the frontend as well as the system binaries by specifying the environment

We specifying how we want a network to handle data which is through a bridge network with –network=bridge

the working directory for root with --workdir=/root

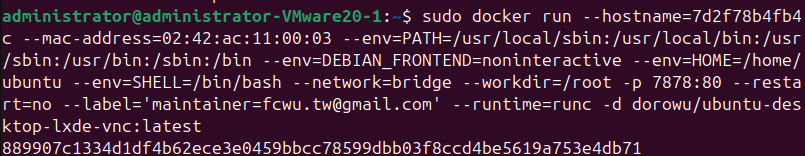
which port is being used with the -p option 7878 aliases to port 80

we then state no restart with --restart=no

and specify our runtime which is our container with the runtime=runc and -d options

docker run --hostname=7d2f78b4fb4c --mac-address=02:42:ac:11:00:03 --env=PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin --env=DEBIAN\_FRONTEND=noninteractive --env=HOME=/home/ubuntu --env=SHELL=/bin/bash --network=bridge --workdir=/root -p 7878:80 --restart=no --label='maintainer=fcwu.tw@gmail.com' --runtime=runc -d dorowu/ubuntu-desktop-lxde-vnc:latest

What did you see?  
889907c1334d1df4b62ece3e0459bbcc78599dbb03f8ccd4be5619a753e4db71

  
  
3. Navigate to the browser and enter

http://localhost:7878

What did you see?

noVNC



III. Pulling with Docker run:

3.1 Virtual Android Devices  
  
1. In your ubuntu command line enter the following command:

docker run -d -p 6080:6080 -e EMULATOR\_DEVICE="Samsung Galaxy S10" -e WEB\_VNC=true --device /dev/kvm --name android-container budtmo/docker-android:emulator\_11.0

This should run a container named budtmo, which is a docker image that can run several different android device emulators. In our cause we’re passing the -e variable for a Galaxy s10.

IF YOU HAVE ISSUES USE THE FOLLOWING FIX DUE TO A LACK OF KVM

For the docker lab there is an issue with section 3.1 which covers the virtual andriod device. This is due to the lack of KVM support. In order to run the container originally we issued " **docker run -d -p 6080:6080 -e EMULATOR\_DEVICE="Samsung Galaxy S10" -e WEB\_VNC=true --device /dev/kvm --name android-container budtmo/docker-android:emulator\_11.**0 "

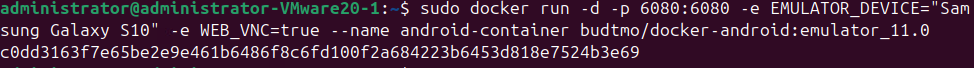
issue the following command instead:

**" docker run -d -p 6080:6080 -e EMULATOR\_DEVICE="Samsung Galaxy S10" -e WEB\_VNC=true --name android-container budtmo/docker-android:emulator\_11.0 "**

Once the container finishes running in the command line, what is the output? Describe how this container could be used, why would you want a containerized Android environment?

c0dd3163f7e65be2e9e461b6486f8c6fd100f2a684223b6453d818e7524b3e69

There is a few reasons for having a android container, if you were a developer for android apps or products you’d want to have a virtual environment. Pentester would also want to be able to test different infiltration methods on a VM.



2. Connect to your emulated android phone by going to the browser. Navigate to your browser and enter the address formatted as follows:  
  
http://youripaddress:6080

What do you see? Turn on the device and make sure it works with the sidebar power button. Explore the environment and share an image of you searching an image using the google chrome app in your device.  
I couldn’t get the emulator to show up no matter what I did.



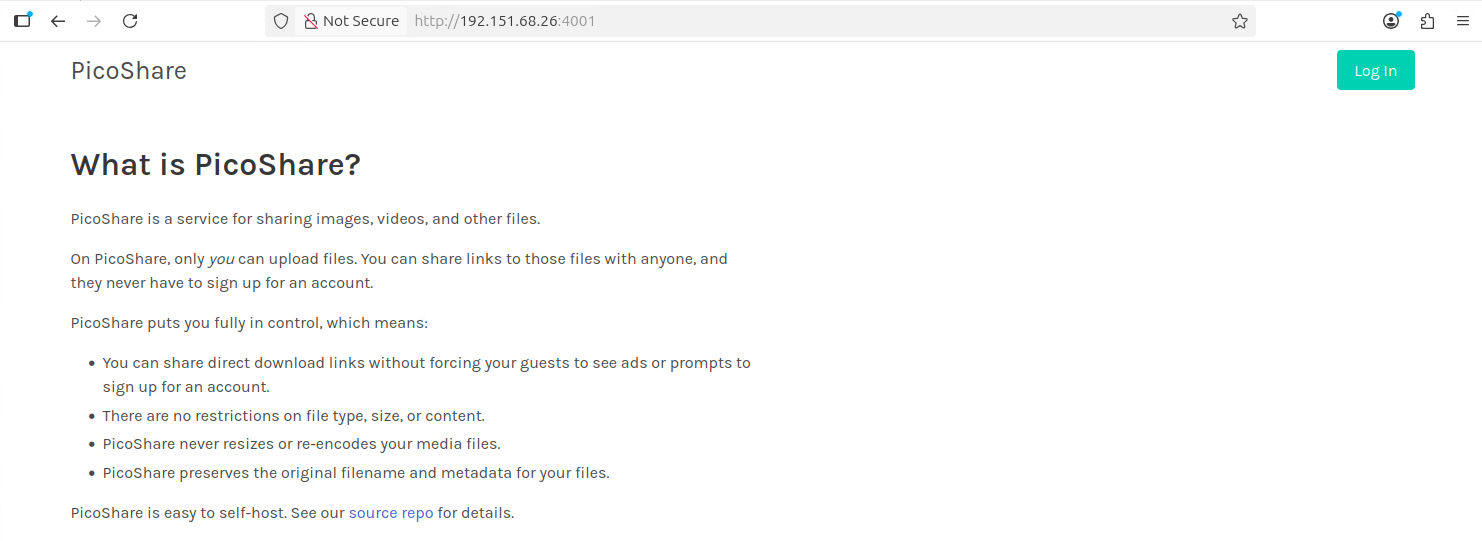
If you’re not doing this locally,   
  
3.2 Picoshare Admin (LAN FIleshare)   
**Local NAS Solution using PicoShare**

*1.* Picoshare is a service that allows you to easily deploy as a Network resources other users can use public links and access control for file sharing. To start launch Picoshare with the following command:

docker run --env "PORT=4001" --env "PS\_SHARED\_SECRET=somesecretpass" --publish 4001:4001/tcp --volume "${PWD}/data:/data" --name picoshare mtlynch/picoshare

What is the output?

7099c0ff105a1f74286ff60b3f3c0cc1f30628e01c8c699a406aff7741a88f57



2.Upload three different file types by clicking the upload file and by choosing a file or paste text options.

*A white background with black text

Description automatically generated*

What files did you upload?

Uploaded a png, jpeg, and a docx file.



3. Set the link for 30 days and try to access your files from a browser using incognito mode. You will enter the link it provides into the browser.

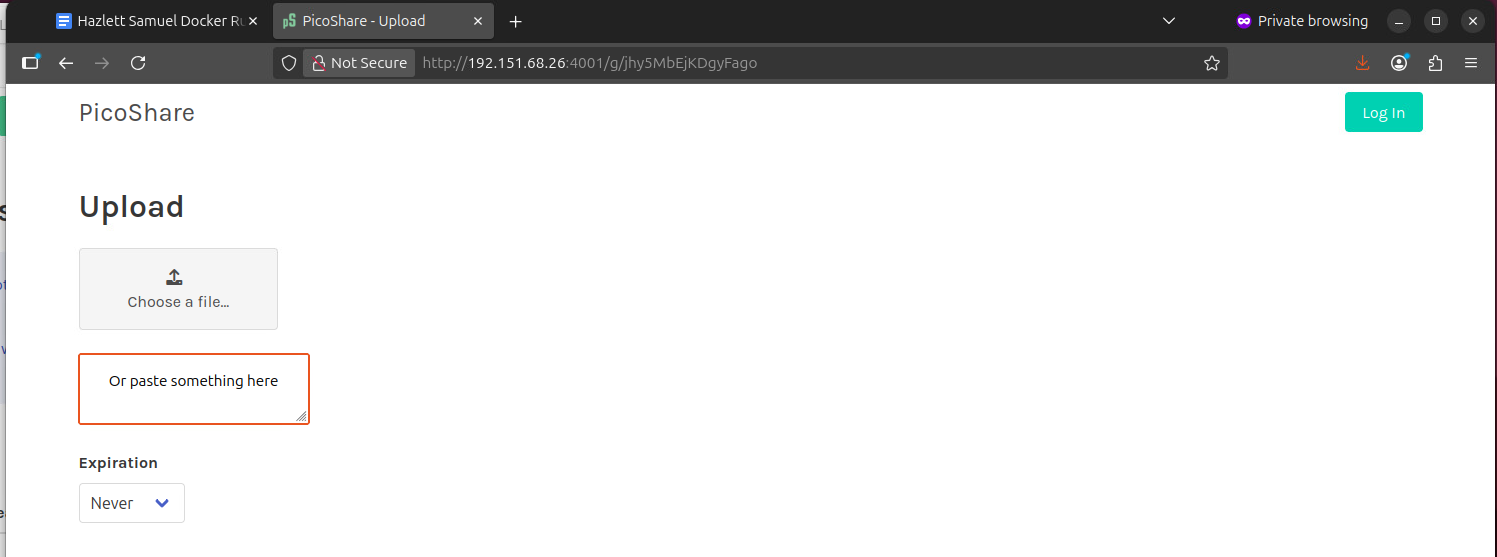
*A screenshot of a computer

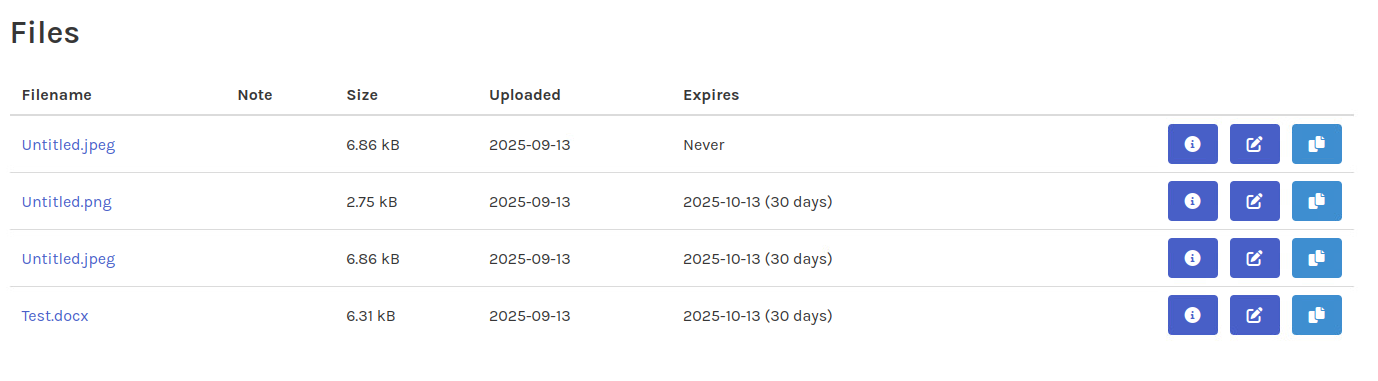
Description automatically generated*

Take a screenshot of your completed upload screen for a file .

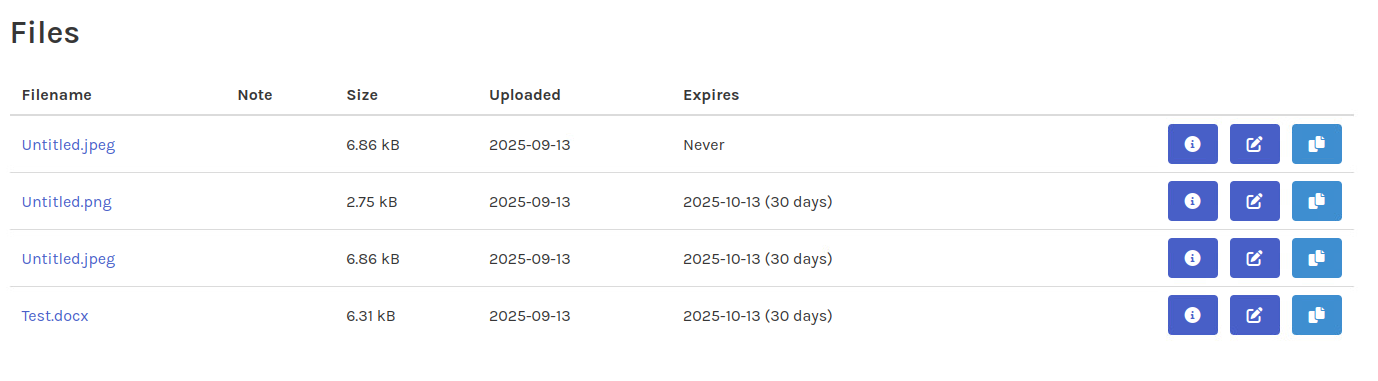
Are you able to see your files?

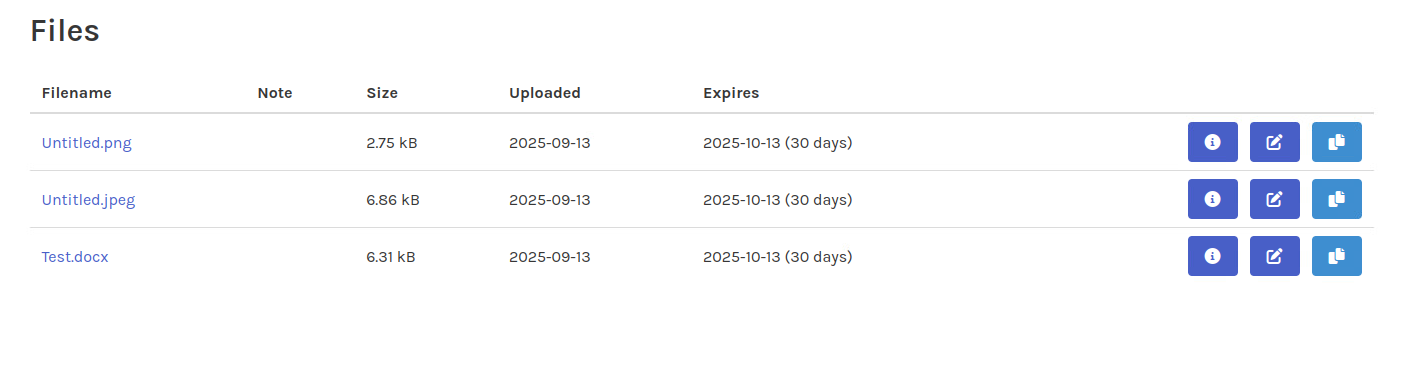
Yes in incognito I uploaded another jpeg





4. Navigate to files tab, in order to delete a file or modify it chose the pencil icon to edit the file or delete it. Delete a file and show a screenshot before showing all files and after you delete that file.

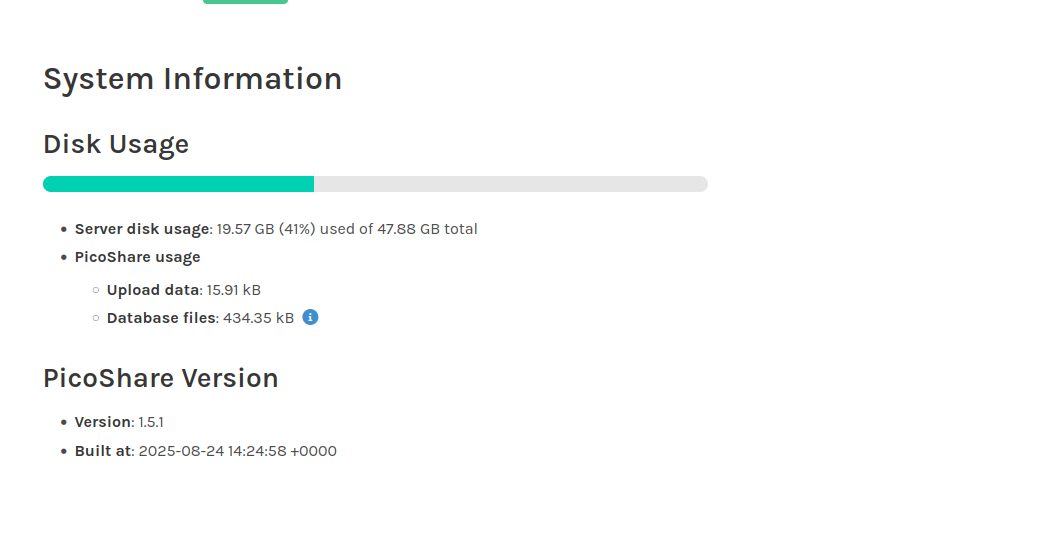




You can also check system settings and storage by clicking the system button: *A screenshot of a computer

Description automatically generated*

Provide a screenshot of this screen from your VM



3.3 Nextcloud Admin (Local Cloud Environment)

1. Start by running the following command to launch Nextcloud:

docker run -d --name=nextcloud -e PUID=1000 -e PGID=1000 -e TZ=Etc/UTC -e NEXTCLOUD\_DEBUG=true -p 443:443 -v /home/yourusername/config:/config -v nextcloud-data:/var/www/html --restart=unless-stopped lscr.io/linuxserver/nextcloud:latest

What is the output?

Unable to find image 'lscr.io/linuxserver/nextcloud:latest' locally

latest: Pulling from linuxserver/nextcloud

d4630f5991ad: Pull complete

e1cde46db0e1: Pull complete

79a1e9e6d785: Pull complete

116b5ca80343: Pull complete

4be280d3104d: Pull complete

427146d25286: Pull complete

3d2fb2be56b5: Pull complete

d2bd286058ef: Pull complete

966c7f99d68c: Pull complete

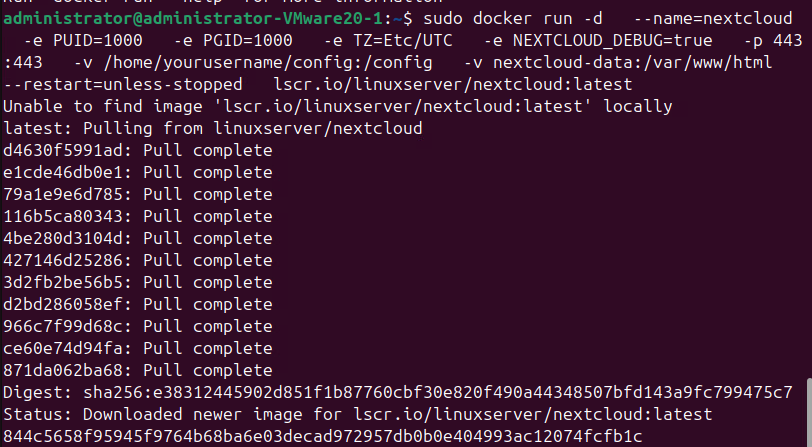
ce60e74d94fa: Pull complete

871da062ba68: Pull complete

Digest: sha256:e38312445902d851f1b87760cbf30e820f490a44348507bfd143a9fc799475c7

Status: Downloaded newer image for lscr.io/linuxserver/nextcloud:latest

844c5658f95945f9764b68ba6e03decad972957db0b0e404993ac12074fcfb1c



*You will be prompted to create an admin account:*

*Admin*

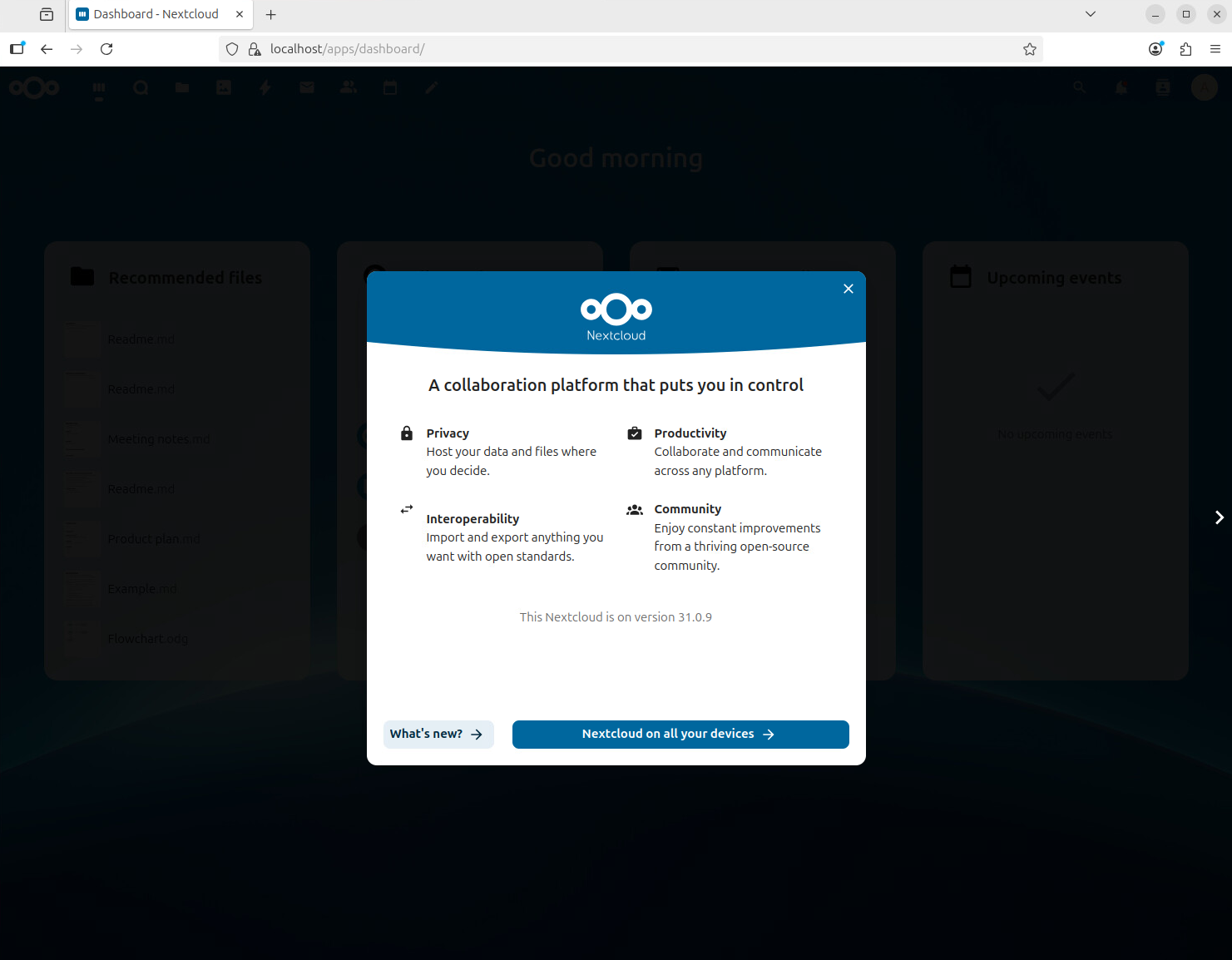
*Pa$$w0rd  
  
A screenshot of a computer

Description automatically generated*

2. After this you will be asked which apps you want to install, leave the defaults and then you can click install recommended apps:  
  
A screenshot of a computer

Description automatically generated

3. After this finishes installing, then you will be brought to the Nextcloud Dashboard, skip installing on other devices

What do you see, include the url in your submission?  


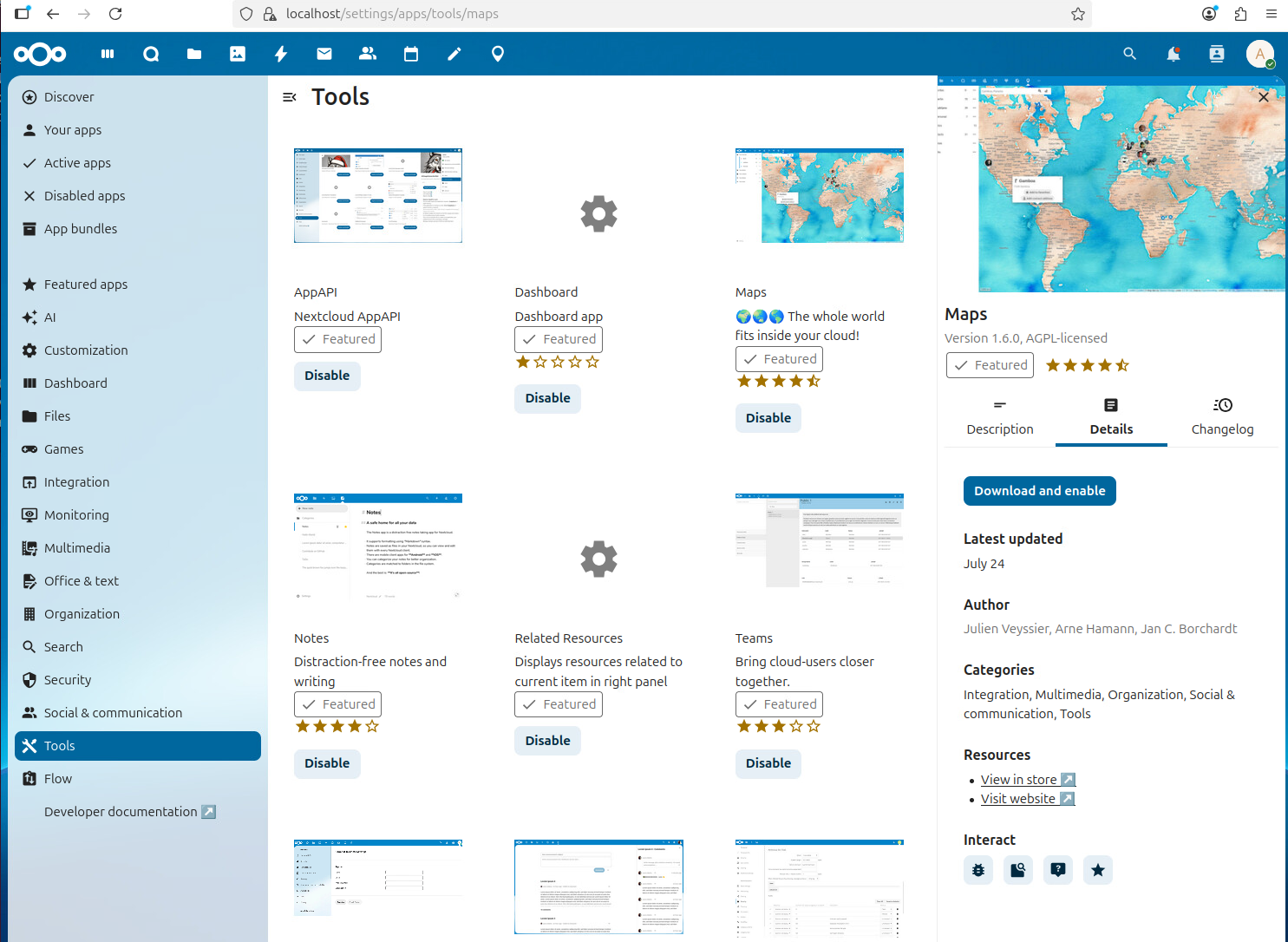
4. Navigate to the apps menu by selecting the user in the top right corner of the window:  
  
A screenshot of a computer

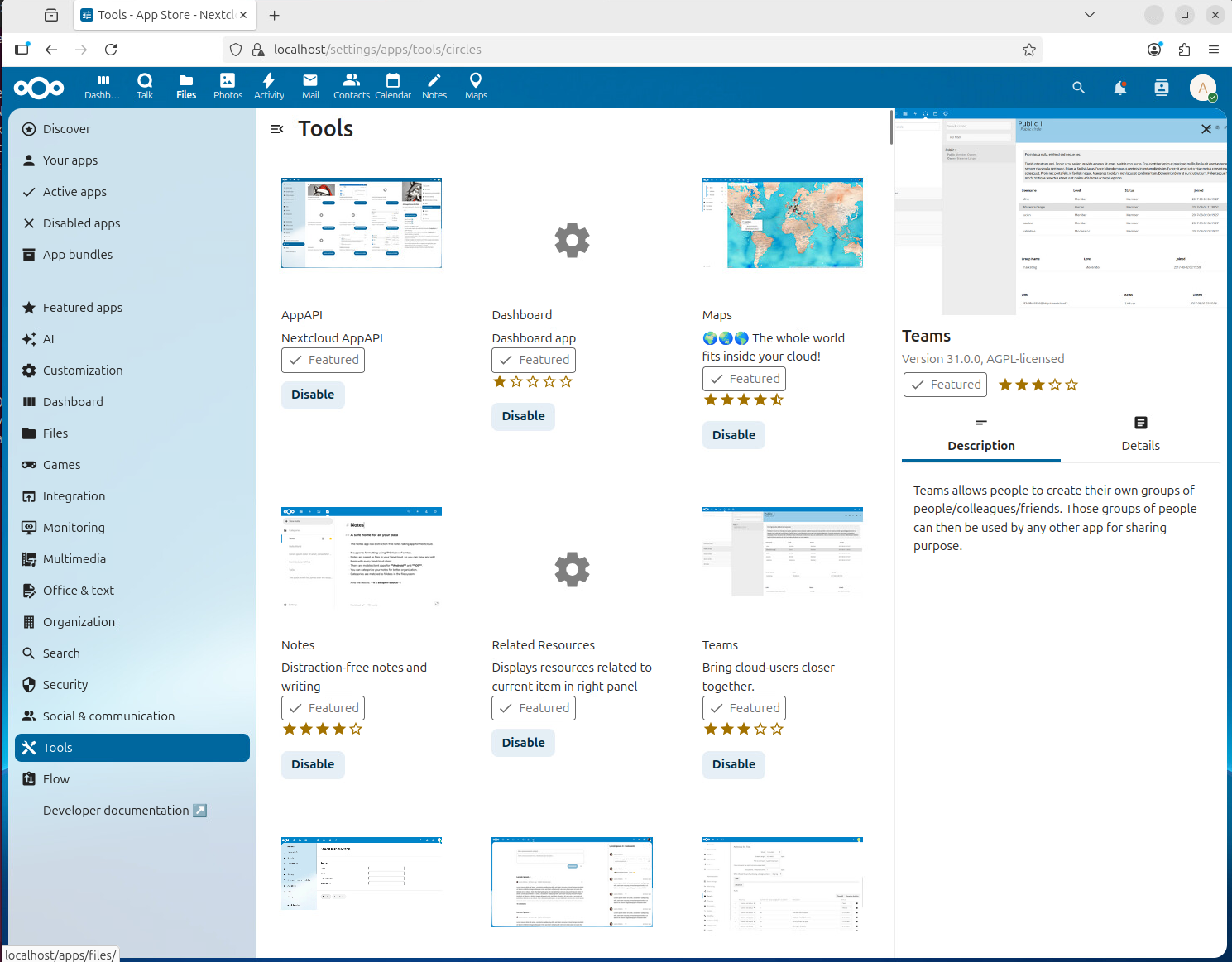
Description automatically generated

5. Once in the apps menu, click tools and then click install on maps:  
A screenshot of a computer

Description automatically generated

6. Go to the maps tab that was created at the top. What do you see after clicking this? Explore several of the tabs and describe what you see or post screenshots.





**If you’re not doing this locally, you can skip this next section below. If you do complete it, it will count as extra credit**

**~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~**

**Bonus : Complete the below questions for extra credit. If you do not complete this section, skip to the last page of this document to complete the assignment.**

7. Go to Docker Desktop then click containers, after this , click the container that is created for Nextcloud, from here you should see the EXEC tab to execute remote commands into your container:  
  
A screenshot of a computer

Description automatically generated  
  
This will open a command line for our container the same as if we did

*docker exec -it <container\_name> /bin/sh*

**Allow Access for Others in Nextcloud**

*1. navigate to the Config file for nextcloud by going to the /config/www/nextcloud/config*

*Edit the config.php file to have your IP address as 1 => ‘xxx.xxx.xxx.xxx’ (NOTE: if you do not have a static Ip then you will have to change this each time your DHCP lease is up) Then save the edited file.*

*A computer screen with white text

Description automatically generated*

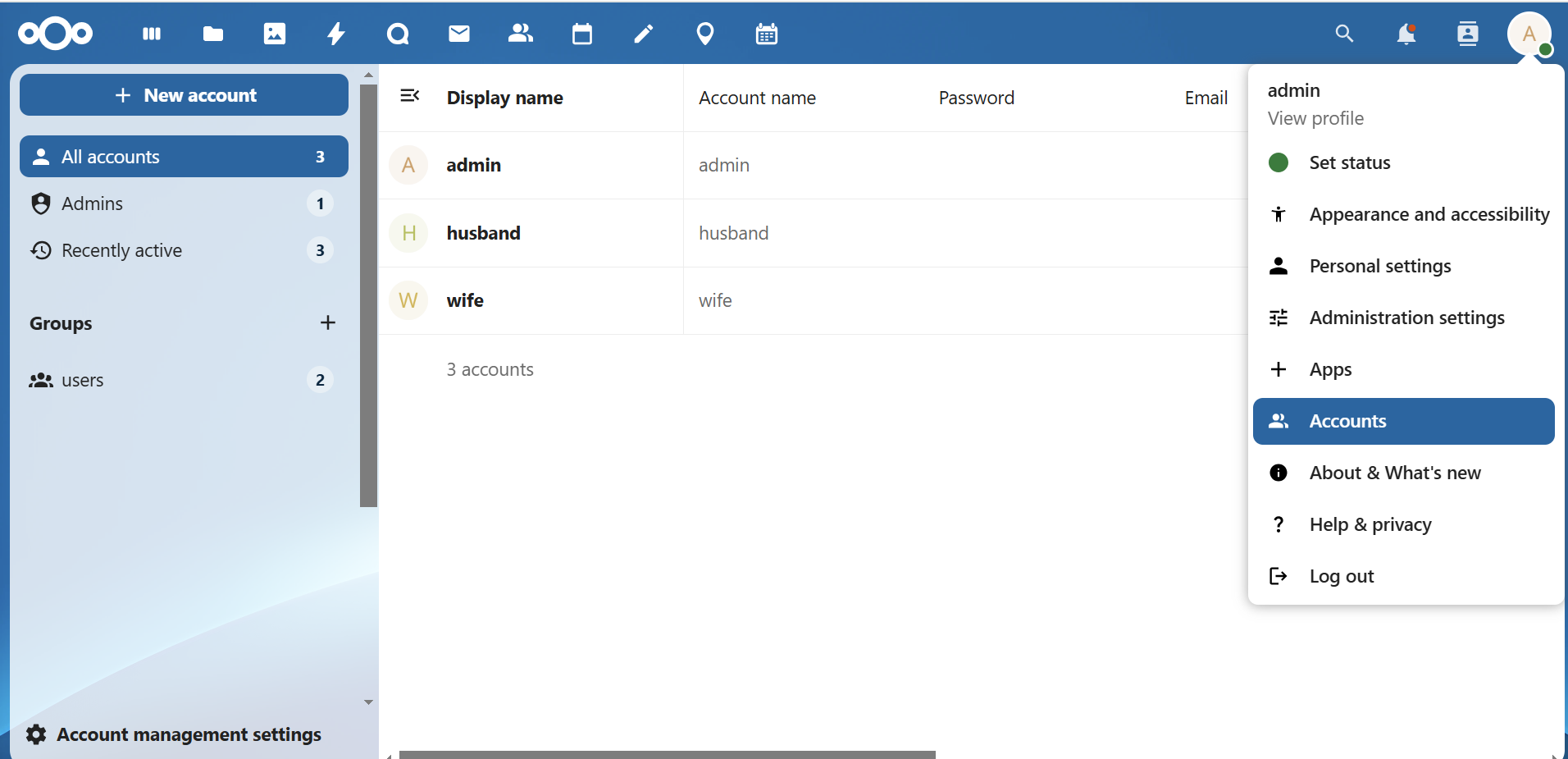
What is your config file?

2. Restart your container by clicking the restart button in Docker Desktop:  
  
A screenshot of a computer

Description automatically generated

This will make it so that you can access your Nextcloud through your IP address as well as the port you assigned it  
  
example: <https://192.168.86.106:443> (NOTE: you must use HTTPS to access this)

Provide a screenshot of the Nextcloud Dashboard once you have signed in using your IP address:

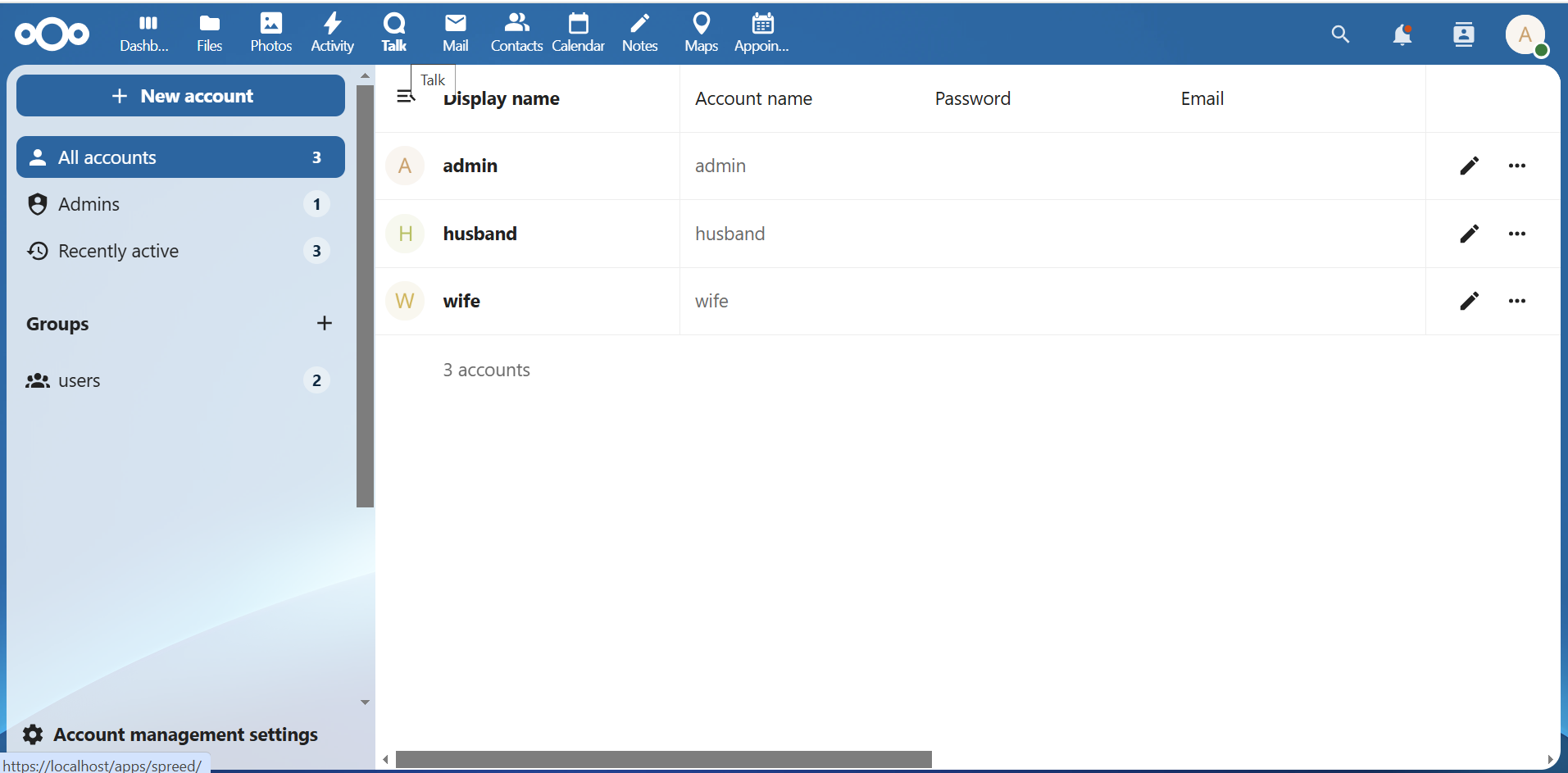
3. Create a User by going to accounts in the top right corner:  


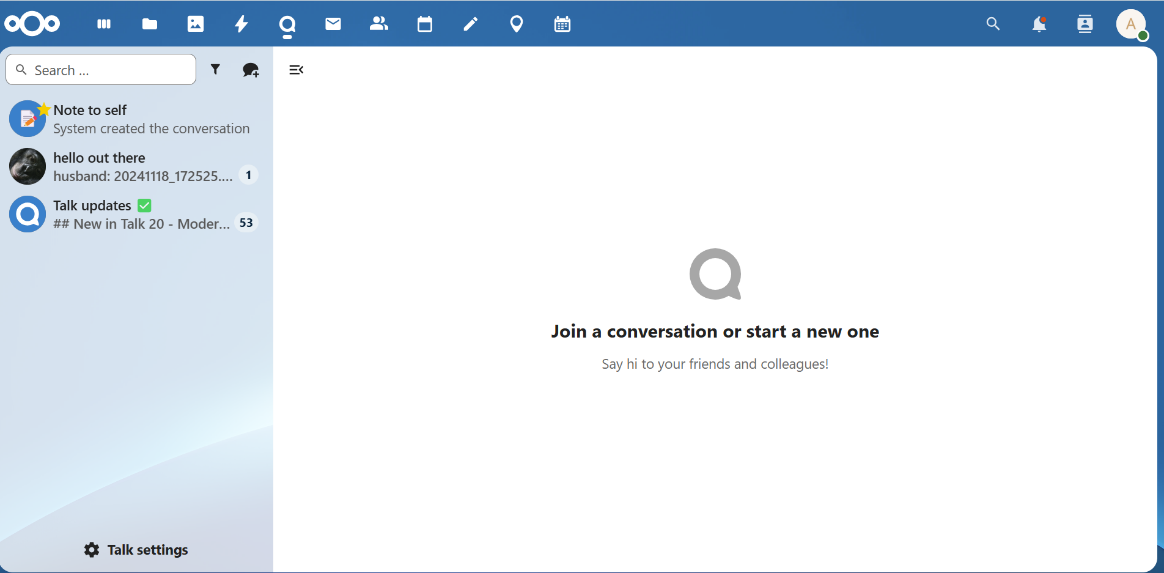
4. Create a group named admin and assign your user to it

A screenshot of a computer

Description automatically generated

5. Add a new account by clicking the new account button. It will ask you to confirm with the admin password you set at the beginning.  
  
What are your users after creating this, please show the updated window with your new user?

6. Navigate to the Talk Application:  
  




7. SWITCH TO YOUR PHONE Connect your phone browser to the nextcloud ip address with HTTPS:// in front of your ip above and sign in as your new user

https://your\_ip:443

What did you see? Who are you logged in as?

6. Go to the chat app

A screenshot of a computer

Description automatically generated

7. Create a new chat by clicking the chat bubble with the “+”

A screenshot of a computer

Description automatically generated

Name your chat in the pop-up then click “add participates”

A screenshot of a chat

Description automatically generated

Then add your participants like so:  
  
A screenshot of a chat

Description automatically generated

8. Enter the new chat and start a call

Between your two devices.  
  
Did this work? You should be able to communicate between both devices.

IV. Clean-up:  
  
Go to Docker Desktop and click the containers tab:  
A screenshot of a computer

Description automatically generated

Select your running containers and delete them.

What does your container dashboard look like after this?

**Submission instruction**

After completing this work, submit this word document with your answers to Canvas. In your word document, your answers should be in blue font.