

Personal Cloud using Raspberry Pi

Abstract:

Our project explains the personal edge cloud storage using Raspberry Pi gives us where we create Cloud storage for our personal use rather than relying on other cloud storage services like Dropbox, Google Drive, iCloud etc. In this dynamic environment and ever-changing technologies, the security of our data is of utmost importance as well as the storage we need to store data and have control over our data. We focus on the above-mentioned problems. As third-party Cloud services are open to other users too, so this creates an issue of security. As well as these Cloud service providers provide a limited amount of storage, and they also have some control over our data too and we have to pay a hefty amount of money to use these Cloud services. Using Raspberry Pi, we can use our external hard drive as Cloud storage for our personal use only. We can decide the amount of memory by ourselves and can have full control over our data. Using Nextcloud we can access our data in our external hard drive through any device which has internet, treating our external hard drive as a Cloud storage device.

Introduction:

Personal cloud storage using Raspberry pi services allows for synchronisation of local folders with raspberry that act as servers in the cloud. Personal cloud storage offers free services, synchronizing devices and sharing content. Personal cloud storage also can be referred to as the way of accessing software and storing data in the cloud representation of the internet. It is also an excellent way to make sure all your files are accessible anywhere you go. Raspberry pi is also the best alternative to make personal cloud storage because it confirms the security. The reason is that, without installing any additional software, Raspberry pi can use Secure Shell (SSH).

Secure Shell (SSH) is known as a UNIX-based command interface and protocol for securely getting access to a remote computer. As well as SSH allows you to connect to your server securely and perform Linux command-line operations. This can easily connect our Raspberry pi which can act as a server from another computer such as a Linux computer, Mac, Android, etc. Furthermore, SSH is the alternative way we can make sure all data is safe between two computers when it is accessed through the internet.

Our project Nextcloud using Raspberry Pi provides a solution for the lacking qualities of the cloud. Firstly, your data will be available to you all the time i.e., your personal hard drive will act as your own cloud server and secondly, your data will get maximum security so that it won't get theft or crashing of the server won't be a problem anymore.

Objectives:

- i. To develop personal cloud storage using Raspberry pi that only uses password and username with added security features involving encryption provided by Nextcloud.
- ii. To add on some features in cloud storage where users can expand storage space using their own secondary storage devices without any charge

Requirements:

- Raspberry Pi 400
- SD Card (16 GB+)
- Ethernet Cable or Wi-Fi

- USB Keyboard
- USB Mouse

Implementation:

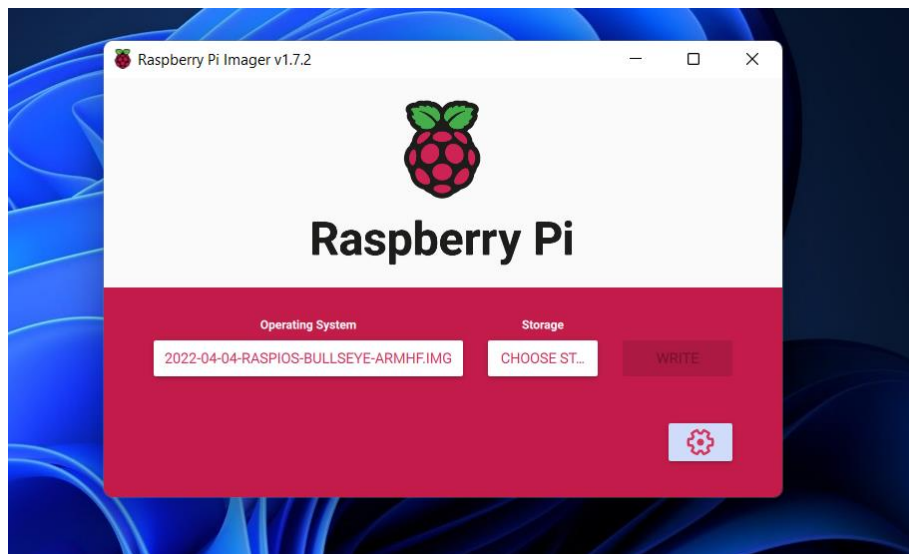
Part 1 - Self-Hosting Password Manager Bitwarden on Raspberry Pi

Step 1: Installing Raspberry Pi Os on SD Card

Download the Raspberry Pi Imager from the website [raspberrypi.org](https://www.raspberrypi.org)

In the software dropdown choose – Raspberry Pi OS (32-bit)

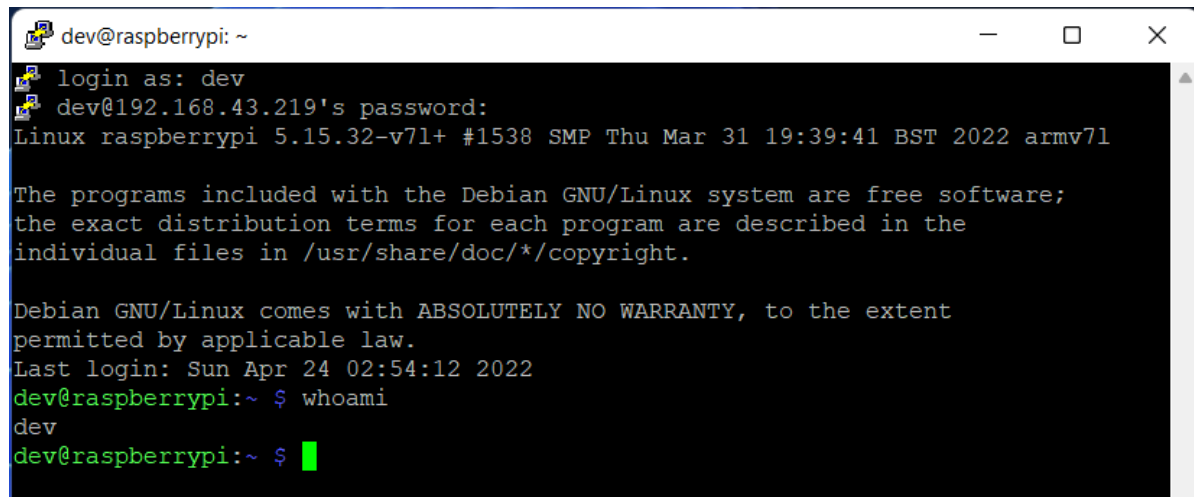
Select your inserted SD card and wait for the imager to finish



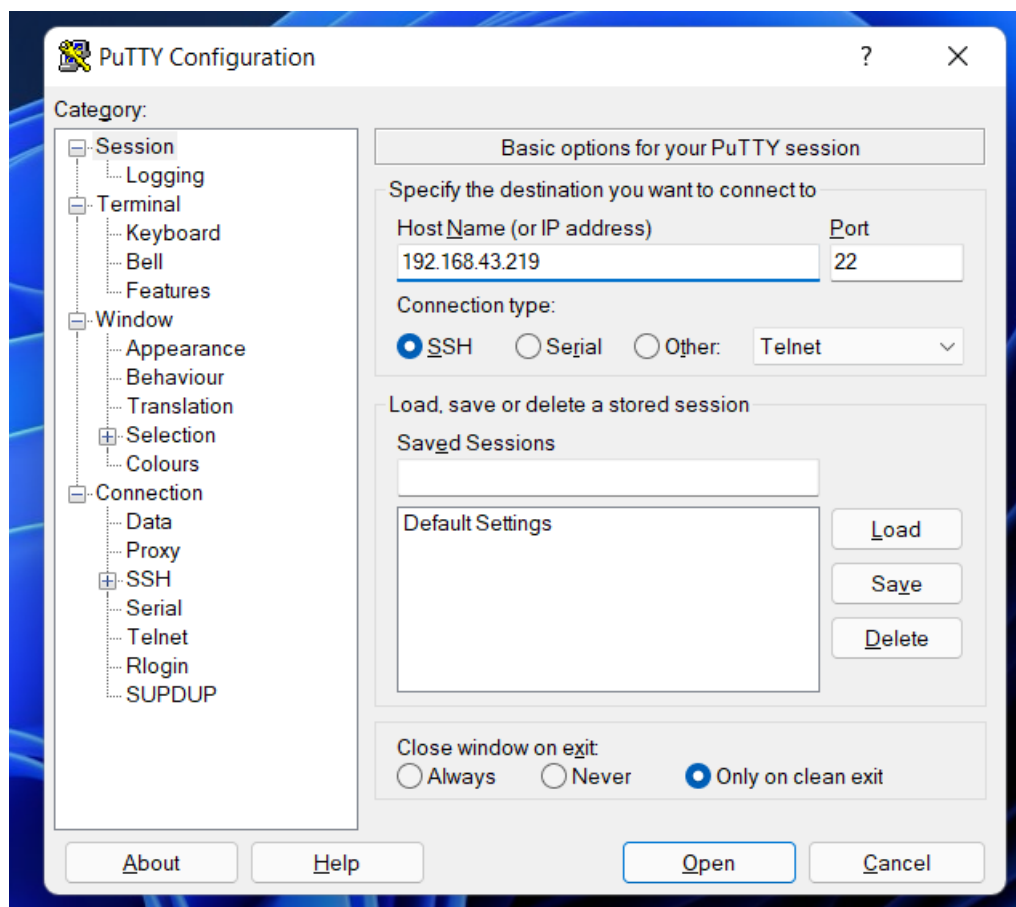
Step 2: Connecting with SSH

Once Raspian is installed reboot and log in again.

ssh dev@192.168.43.219:8080



```
dev@raspberrypi: ~  
login as: dev  
dev@192.168.43.219's password:  
Linux raspberrypi 5.15.32-v7l+ #1538 SMP Thu Mar 31 19:39:41 BST 2022 armv7l  
  
The programs included with the Debian GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
Last login: Sun Apr 24 02:54:12 2022  
dev@raspberrypi:~ $ whoami  
dev  
dev@raspberrypi:~ $
```



Once we are connected to SSH we will then install Docker

Step 3: Installing Docker

Docker is a platform that delivers software in easy to use packages called containers.

```
curl -fsSL https://get.docker.com -o get-docker.sh
```

```
sudo sh get-docker.sh
```

Step 4: Setting Docker non-root privileges

Adding user 'dev' to the Docker group so we run Docker as non-root

```
sudo usermod -aG docker dev
```

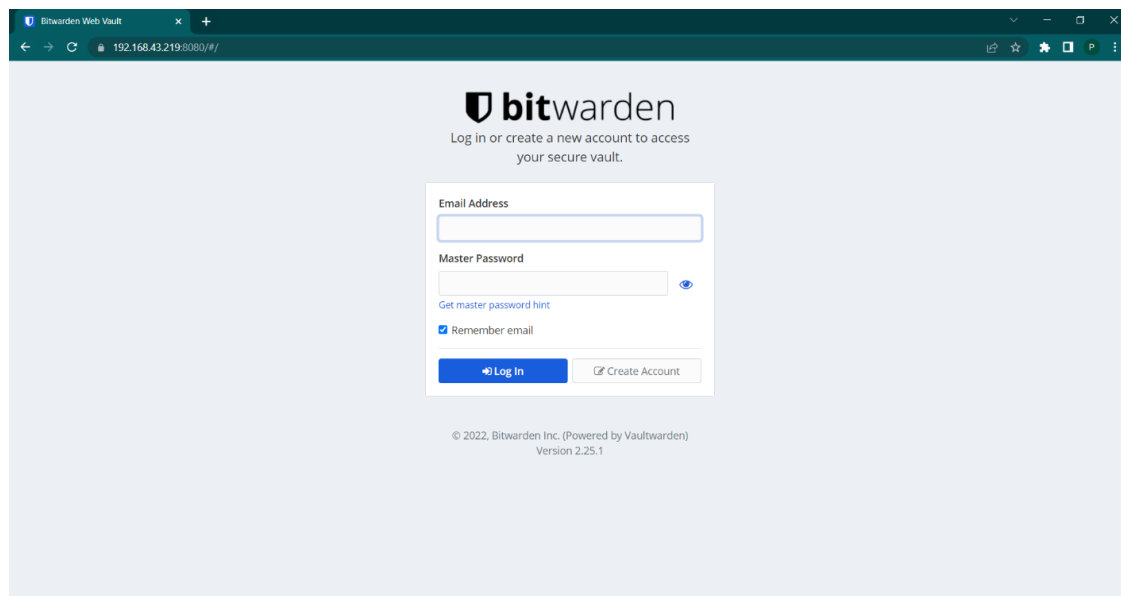
Restart Docker so recognizes this group change:

```
sudo reboot now
```

Step 5: Install and Run Bitwarden for Raspberry Pi

```
docker run -d --name bitwarden -v /bw-data:/data/ -p 8080:80 bitwardenrs/server:latest-arm32v6
```

Check the Raspberry Pi IP address



The Bitwarden server won't work entirely until we have a working HTTPS certificate for it.

Step 6: Creating a self-signed HTTPS certificate (SSL)

- Create a CA key (your own little on-premises Certificate Authority):

```
openssl genpkey -algorithm RSA -aes128 -out private-ca.key -outform PEM -pkeyopt  
rsa_keygen_bits:2048
```

- Create a CA certificate:

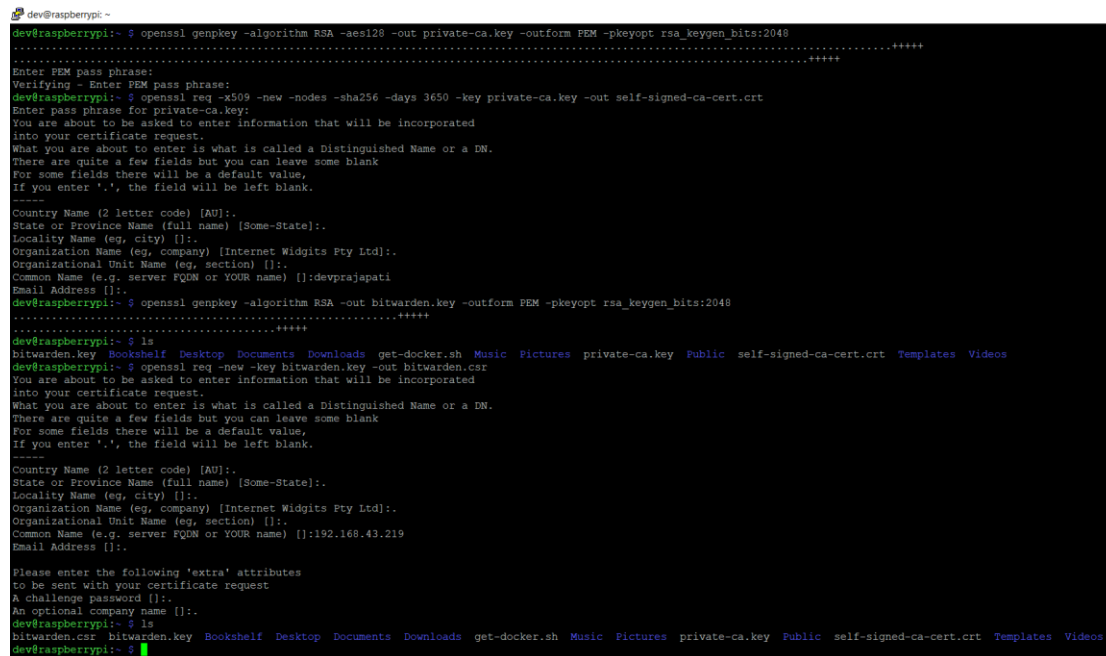
```
openssl req -x509 -new -nodes -sha256 -days 3650 -key private-ca.key -out self-signed-ca-  
cert.crt
```

- Create a bitwarden key:

```
openssl genpkey -algorithm RSA -out bitwarden.key -outform PEM -pkeyopt  
rsa_keygen_bits:2048
```

- Create the bitwarden certificate request file:

```
openssl req -new -key bitwarden.key -out bitwarden.csr
```



```
dev@raspberrypi:~$ openssl genpkey -algorithm RSA -aes128 -out private-ca.key -outform PEM -pkeyopt rsa_keygen_bits:2048  
.....+++++  
Enter PEM pass phrase:  
Verifying - Enter PEM pass phrase:  
dev@raspberrypi:~$ openssl req -x509 -new -nodes -sha256 -days 3650 -key private-ca.key -out self-signed-ca-cert.crt  
Enter pass phrase for private-ca.key:  
You are about to be asked to enter information that will be incorporated  
into your certificate request.  
What you are about to enter is what is called a Distinguished Name or a DN.  
There are quite a few fields but you can leave some blank  
for some fields there will be a default value,  
If you enter '.', the field will be left blank.  
-----  
Country Name (2 letter code) [AU]:.  
State or Province Name (full name) [Some-State]:.  
Locality Name (eg, city) []:.  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:.  
Organizational Unit Name (eg, section) []:.  
Common Name (e.g. server FQDN or YOUR name) []:devprajapati  
Email Address []:.  
dev@raspberrypi:~$ openssl genpkey -algorithm RSA -out bitwarden.key -outform PEM -pkeyopt rsa_keygen_bits:2048  
.....+++++  
dev@raspberrypi:~$ ls  
bitwarden.key  Bookshelf  Desktop  Documents  Downloads  get-docker.sh  Music  Pictures  private-ca.key  Public  self-signed-ca-cert.crt  Templates  Videos  
dev@raspberrypi:~$ openssl req -new -key bitwarden.key -out bitwarden.csr  
You are about to be asked to enter information that will be incorporated  
into your certificate request.  
What you are about to enter is what is called a Distinguished Name or a DN.  
There are quite a few fields but you can leave some blank  
for some fields there will be a default value,  
If you enter '.', the field will be left blank.  
-----  
Country Name (2 letter code) [AU]:.  
State or Province Name (full name) [Some-State]:.  
Locality Name (eg, city) []:.  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:.  
Organizational Unit Name (eg, section) []:.  
Common Name (e.g. server FQDN or YOUR name) []:192.168.43.219  
Email Address []:.  
  
Please enter the following 'extra' attributes  
to be sent with your certificate request  
A challenge password []:.  
An optional company name []:.  
dev@raspberrypi:~$ ls  
bitwarden.csr  bitwarden.key  Bookshelf  Desktop  Documents  Downloads  get-docker.sh  Music  Pictures  private-ca.key  Public  self-signed-ca-cert.crt  Templates  Videos  
dev@raspberrypi:~$
```

- To be compatible with the most recent versions of Google Chrome, iOS and macOS, we also need to manually create a file called **bitwarden.ext**

nano bitwarden.ext

- Create a text file bitwarden.ext with the following content, and change the domain names to your setup.

```
dev@raspberrypi: ~
GNU nano 5.4 bitwarden.ext *
authorityKeyIdentifier=keyid,issuer
basicConstraints=CA:FALSE
keyUsage = digitalSignature, nonRepudiation, keyEncipherment, dataEncipherment
extendedKeyUsage = serverAuth
subjectAltName = @alt_names

[alt_names]
IP.1 = 192.168.43.219
```

- Create the bitwarden certificate, signed from the root CA:

openssl x509 -req -in bitwarden.csr -CA self-signed-ca-cert.crt -CAkey private-ca.key -CAcreateserial -out bitwarden.crt -days 365 -sha256 -extfile bitwarden.ext

```
dev@raspberrypi: ~
dev@raspberrypi:~$ nano bitwarden.ext
dev@raspberrypi:~$ openssl x509 -req -in bitwarden.csr -CA self-signed-ca-cert.crt -CAkey private-ca.key -CAcreateserial -out bitwarden.crt -days 365 -sha256 -extfile bitwarden.ext
Signature ok
subject=CN = 192.168.43.219
Getting CA Private Key
Enter pass phrase for private-ca.key:
```

Step 7: Move SSL Certificates

sudo mv bitwarden.crt bitwarden.key /etc/ssl/certs

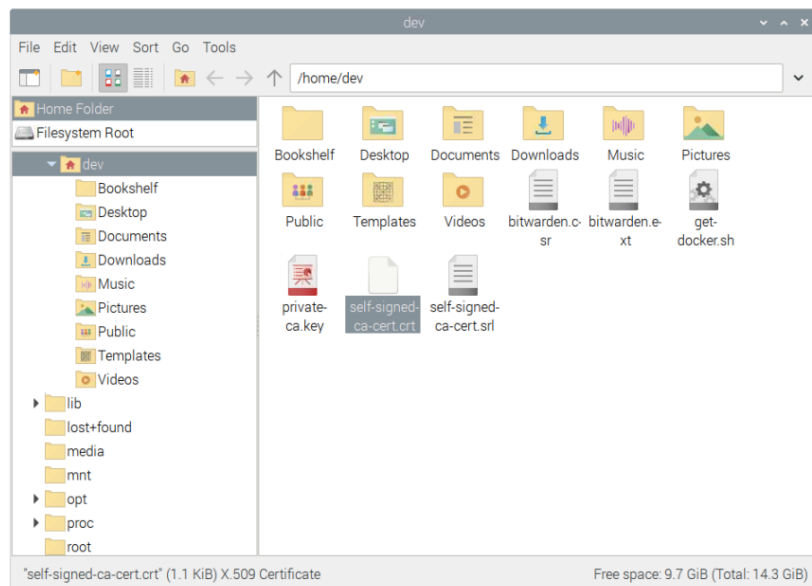
```
dev@raspberrypi: ~
dev@raspberrypi:~$ nano bitwarden.ext
dev@raspberrypi:~$ openssl x509 -req -in bitwarden.csr -CA self-signed-ca-cert.crt -CAkey private-ca.key -CAcreateserial -out bitwarden.crt -days 365 -sha256 -extfile bitwarden.ext
Signature ok
subject=CN = 192.168.43.219
Getting CA Private Key
Enter pass phrase for private-ca.key:
dev@raspberrypi:~$ sudo mv bitwarden.crt bitwarden.key /etc/ssl/certs
dev@raspberrypi:~$
```

Step 8: Running Bitwarden server with certificates

docker run -d --name bitwarden --restart unless-stopped -v /bw-data:/data -v /etc/ssl/certs:/ssl -e ROCKET_TLS='{certs="/ssl/bitwarden.crt",key="/ssl/bitwarden.key"}' -p 8080:80 bitwardenrs/server:latest-arm32v6

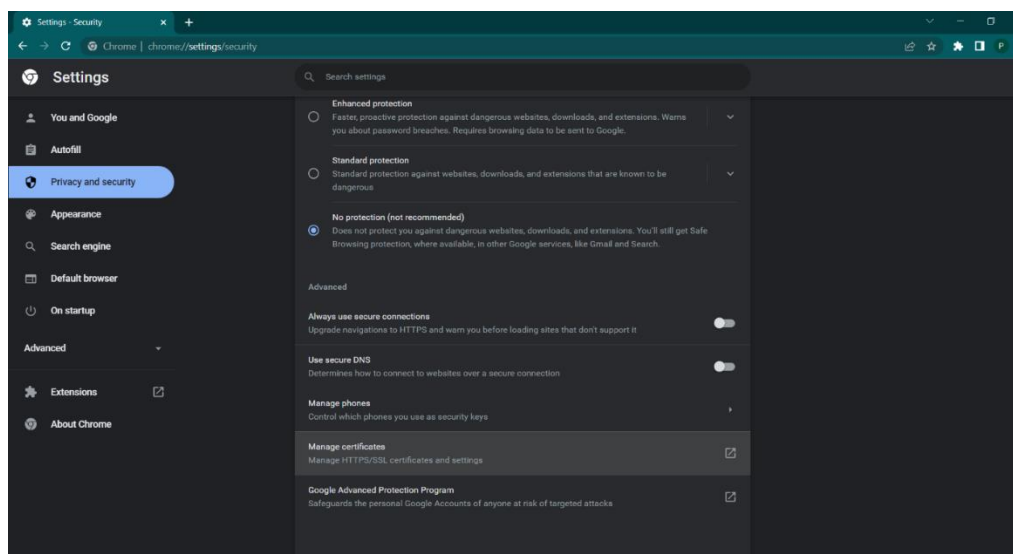
```
dev@raspberrypi:~$ docker run -d --name bitwarden --restart unless-stopped -v /bw-data:/data -v /etc/ssl/certs:/ssl -e ROCKET_TLS='(certs="/ssl/bitwarden.crt",key="/ssl/bitwarden.key")' -p 8080:80 bitwardens/server:latest-arm32v6
e682a1749e828deab3ddb27d592a2ac6db4220346c33fd514fc74d1d6e7427e5
dev@raspberrypi:~$
```

Step 9: Installing Certificates

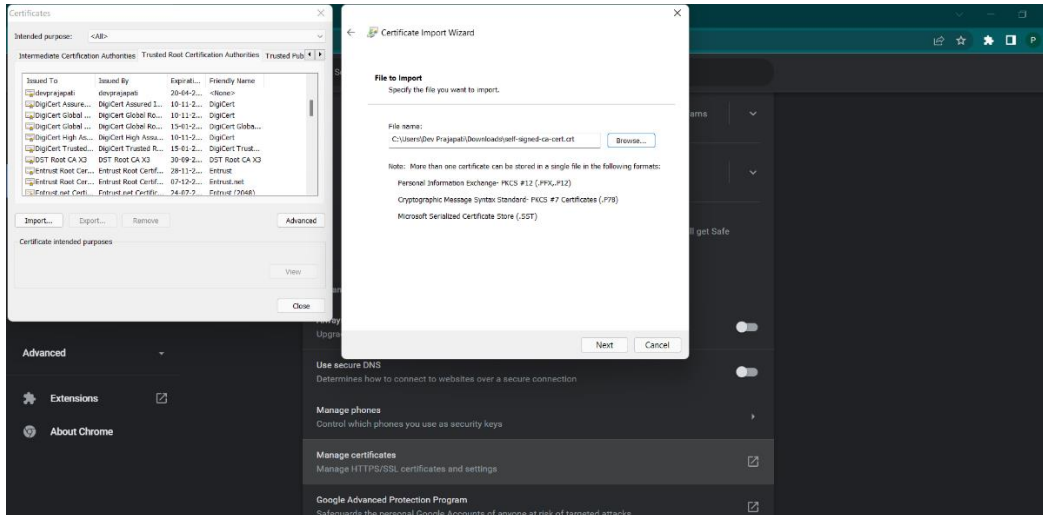


Installation of SSL Certificate on Google Chrome

- Go to Settings
- Privacy and Security
- Manage Certificates

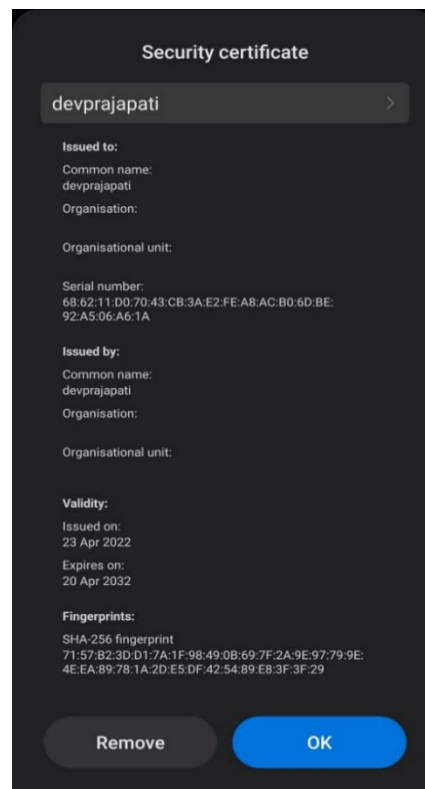
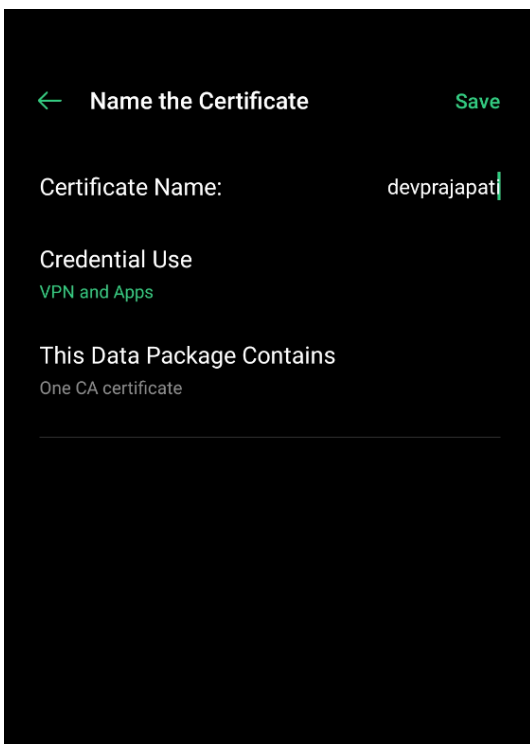


- Click on Trusted Root Certification Authorities
- Click on import and select your certificate.

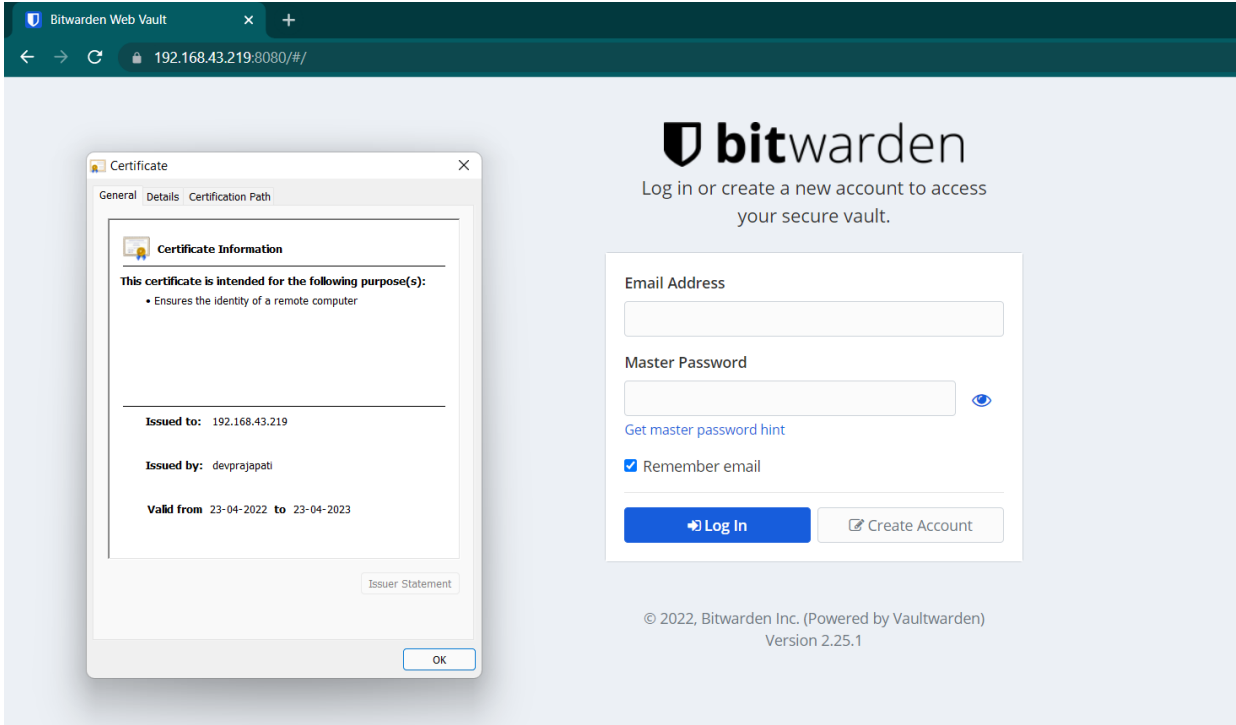
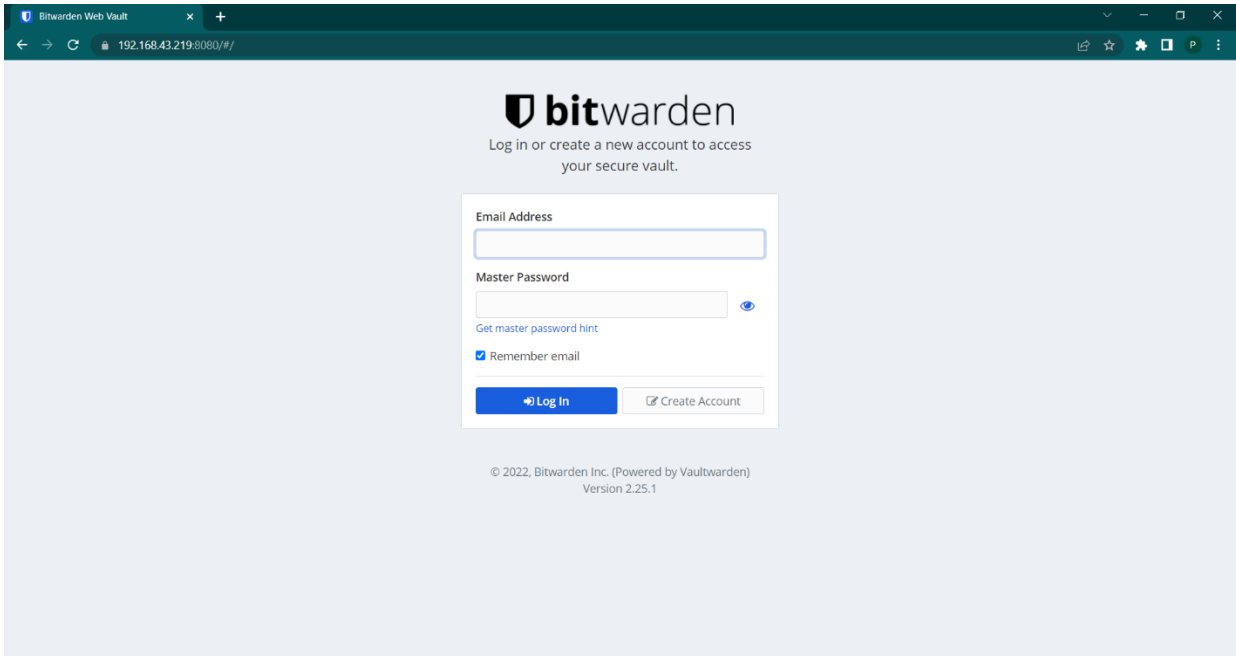


Installation of SSL Certificate on Android

- Download the file
- The following screen pops up



Step 10: Open Bitwarden



Part 2 - Installing & Using Nextcloud on Raspberry Pi

Step 1: Downloading Nextcloud (from Docker)

docker pull arm32v7/nextcloud

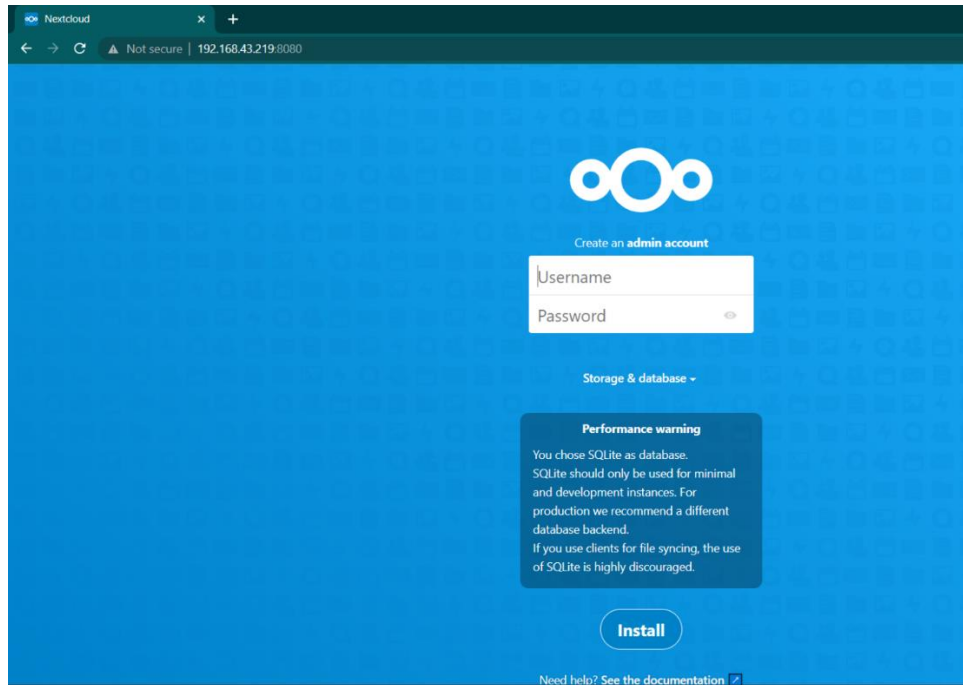
```
dev@raspberrypi: ~  
dev@raspberrypi:~$ docker pull arm32v7/nextcloud  
Using default tag: latest  
latest: Pulling from arm32v7/nextcloud  
a6b2963233cd: Downloading [=====> ] 25.73MB/26.58MB  
72b6b2706f2a: Download complete  
82d413824c0d: Downloading [=====> ] 25.74MB/69.32MB  
311ab91865a3: Download complete  
bee189dc1428: Download complete  
d2eff9fbb6ce: Download complete  
ca2198b0316e: Download complete  
06101efac629: Verifying Checksum  
3d192fe8f5a9: Download complete  
0b4998e5373f: Waiting  
0b09264b7c8e: Waiting  
dcd0e36deab4: Waiting  
9a981b8bef9b: Waiting  
ea167d2cf2d6: Waiting  
8af42082129f: Waiting  
815f332c8017: Waiting  
07a3e6b7af70: Waiting  
ed8c477d3e22: Waiting
```

To start Nextcloud immediately, execute the following

docker run --name nextcloud -d -p 8080:80 nextcloud

```
dev@raspberrypi:~$ docker run --name nextcloud -d -p 8080:80 nextcloud  
Unable to find image 'nextcloud:latest' locally  
latest: Pulling from library/nextcloud  
Digest: sha256:38a03bc483831ceb8fa6f2bc5c050227f66ddaaf46643979ee1a07c0536b1d0a  
Status: Downloaded newer image for nextcloud:latest  
6a5eaa02b9a5afe766ee705e3852b5f494ff35ab9e3496726476f5ed17bb32ea  
dev@raspberrypi:~$
```

The Nextcloud setup screen should show up after a few minutes at the IP address of the pi – **192.168.43.219:8080**



Step 2: Creating the database

By default, Nextcloud uses SQLite Database which is not recommended for permanent use. Therefore, we'll create a dedicated database container for our Nextcloud Installation.

Stop and remove nextcloud

```
docker stop nextcloud && docker rm nextcloud
```

Download the database software called PostgreSQL

```
docker pull postgres
```

```
dev@raspberrypi:~$ docker pull postgres
Using default tag: latest
latest: Pulling from library/postgres
a6b2963233cd: Already exists
ae91ea7162f2: Pull complete
a225ae1d0877: Pull complete
5135069f8a2e: Pull complete
d2e36cb2927d: Pull complete
f6edb7db1e25: Pull complete
b08ab87ae4f0: Pull complete
20b77220fab8: Pull complete
bae5538138bf: Pull complete
710262e6196e: Pull complete
395d4ed02571: Pull complete
d756a410b5ba: Pull complete
87f370d2193a: Pull complete
Digest: sha256:ab0be6280ada8549f45e6662ab4f00b7f601886fcd55c5976565d4636d87c8b2
Status: Downloaded newer image for postgres:latest
docker.io/library/postgres:latest
dev@raspberrypi:~$
```

In order to make the Nextcloud container see the database container, they need to be on the same docker network.

Step 3: Creating a Docker Network

Create a Docker network and name it “nextcloud-net”

docker network create --driver bridge nextcloud-net

```
dev@raspberrypi:~$ docker network create --driver bridge nextcloud-net
2764f7228ea794c4d8f44b83dc1a6a4826c5c6b783bc682c47b1dca1f9f3b849
dev@raspberrypi:~$
```

Step 4: Running PostgreSQL and Nextcloud

```
dev@raspberrypi:~$ docker run --name postgres -e POSTGRES_PASSWORD=123456 --network nextcloud-net -d postgres
2c2278fab903a8167c1adf0a6aa2ab3723c9cbe1822709a77db63d4d86b6402
```

Now the database container should start in the background and automatically create the default user “postgres” along with the default database of the same name.

If you want to optionally mount the database folder to see what’s inside:

docker run --name nextcloud -d -p 8080:80 -v /home/pi/nextcloud:/var/www/html --network nextcloud-net nextcloud

```
dev@raspberrypi:~$ docker run --name postgres -e POSTGRES_PASSWORD=123456 --network nextcloud-net -d postgres
2c2278fab903a8167c1adf0a6aa2ab3723c9cbe1822709a77db63d4d86b6402
dev@raspberrypi:~$ docker run --name nextcloud -d -p 8080:80 -v /home/pi/nextcloud:/var/www/html --network nextcloud-net nextcloud
f47534d1fa40a0124fb75b138ba84cc029298f7cd419779314c180ad2ac3f7b
dev@raspberrypi:~$
```

Step 5: Running Nextcloud and creating an admin account

- Open the IP of the raspberry pi to run the Nextcloud website
- Change the database to PostgreSQL
- Enter the credentials

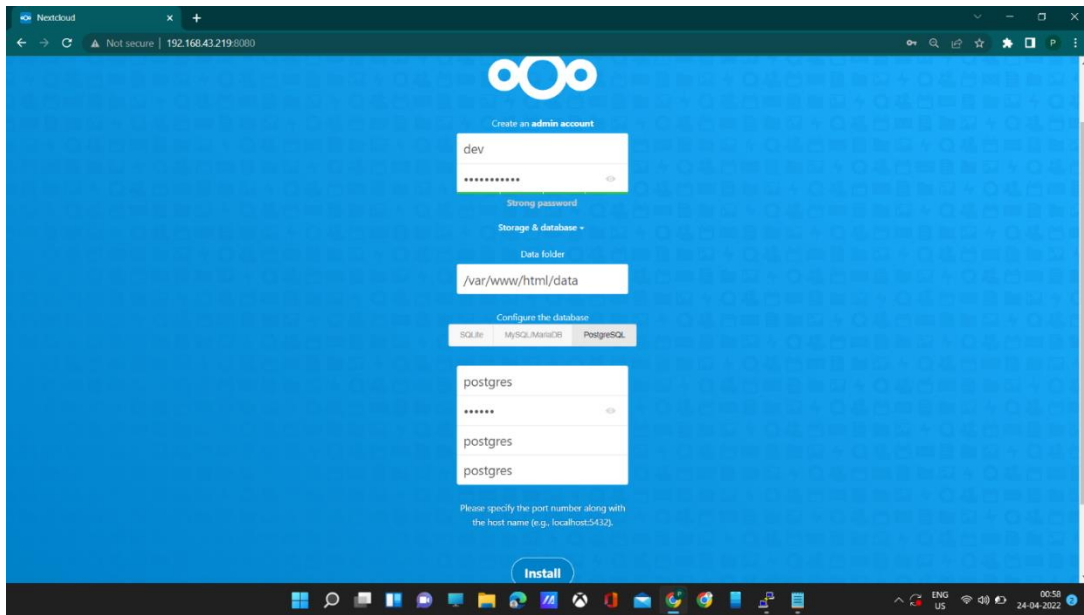
Database user: postgres

Database password: 123456

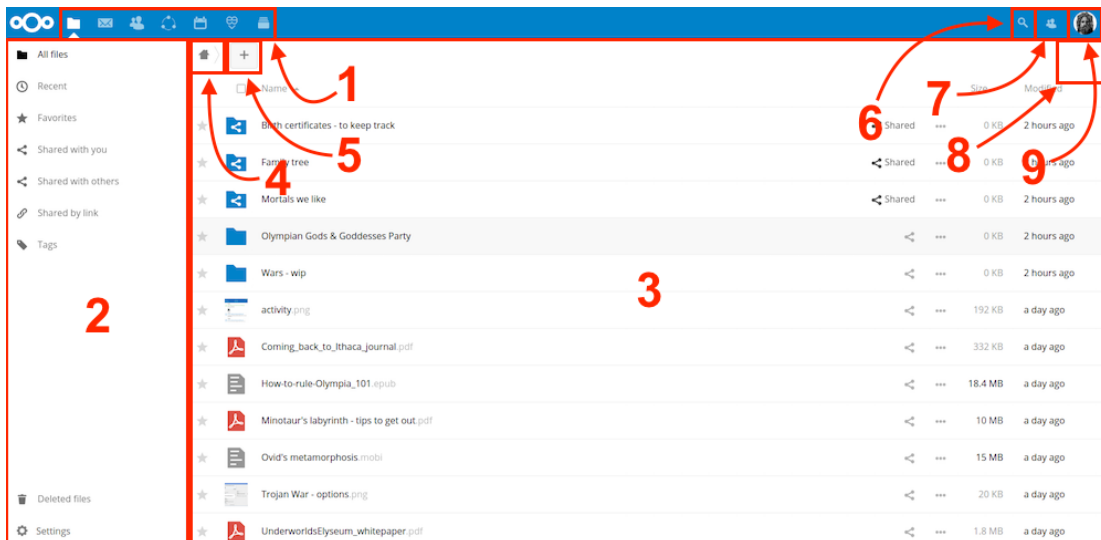
Database name: postgres

Container: postgres

- Finish & Install



Step 6: Nextcloud Features



The Nextcloud user interface contains the following fields and functions:

- **Apps Selection Menu (1):** Located in the upper left corner, you'll find all your apps which are available on your instance of Nextcloud. Clicking on an app's icon will redirect you to the app.
- **Apps Information field (2):** Located in the left sidebar, this provides filters and tasks associated with your selected app. For example, when you are using the Files apps

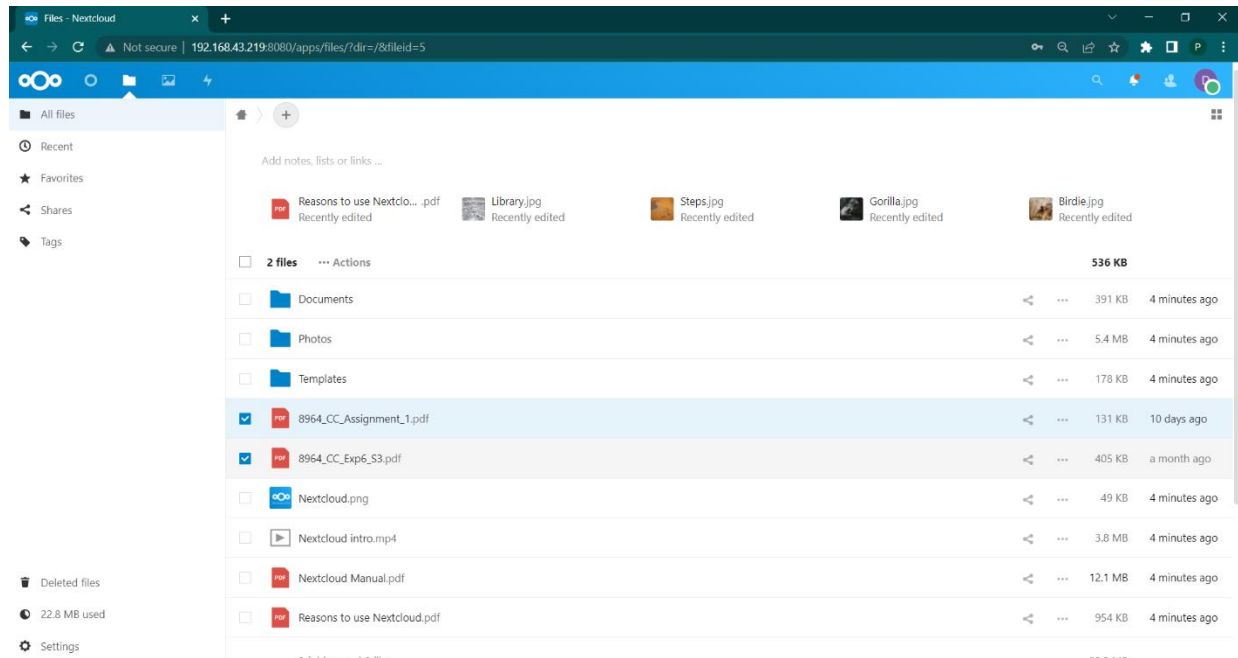
you have a special set of filters for quickly finding your files, such as files that have been shared with you, and files that you have shared with others. You'll see different items for other apps.

- **Application View (3):** The main central field in the Nextcloud user interface. This field displays the contents or user features of your selected app.
- **Navigation Bar (4):** Located over the main viewing window (the Application View), this bar provides a type of breadcrumbs navigation that enables you to migrate to higher levels of the folder hierarchy up to the root level (home).
- **New button (5):** Located in the Navigation Bar, the New button enables you to create new files, new folders, or upload files.

You can also drag and drop files from your file manager into the Files Application View to upload them to your instance.

- **The search field (6):** Click on the Magnifier in the upper right corner to search for files and entries of the current app.
- **Contacts Menu (7):** Gives you an overview of your contacts and users on your server. Dependent on the given details and available apps, you can directly start a video call with them or send emails.
- **Grid view button (8):** This looks like four little squares, which toggles the grid view for folders and files.
- **Settings menu (9):** Click on your profile picture, located to the right of the Search field, to open your Settings dropdown menu. Your Settings page provides the following settings and features:
 - Links to download desktop and mobile apps
 - Server usage and space availability
 - Password management
 - Name, email, and profile picture settings
 - Manage connected browsers and devices
 - Group memberships
 - Interface language settings
 - Manage notifications
 - Federated Cloud ID and social media sharing buttons
 - SSL/TLS certificate manager for external storage
 - Your Two-factor Settings
 - Nextcloud Version information

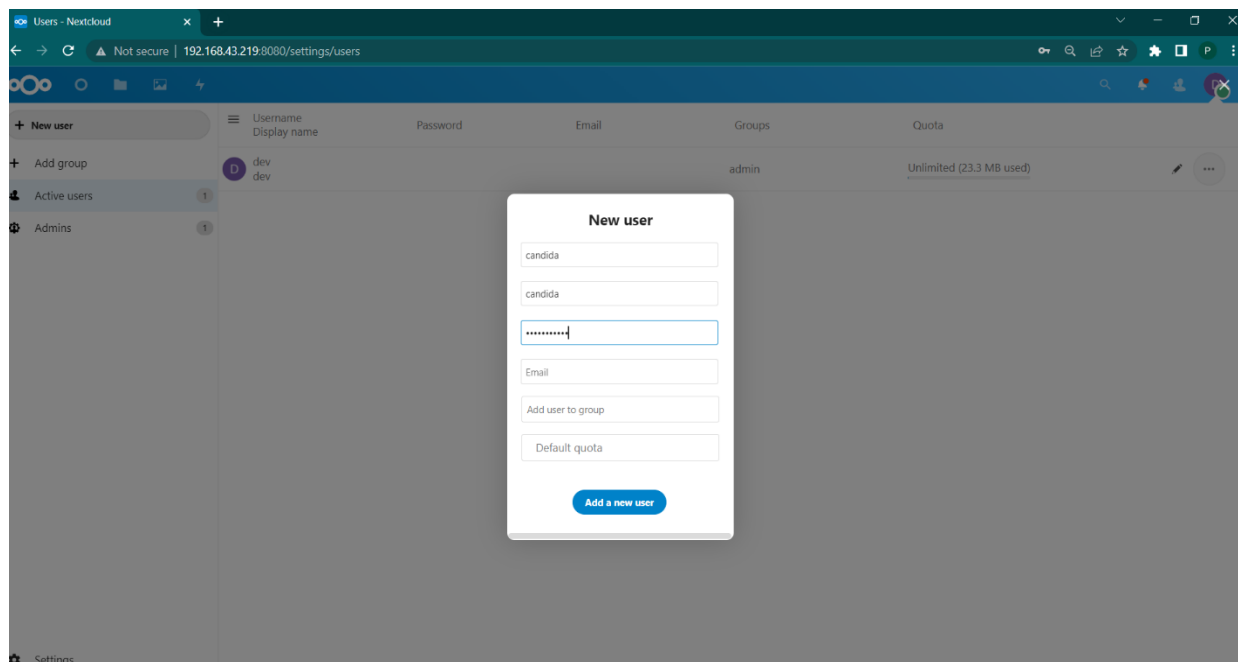
Step 7: Upload files to Nextcloud



Step 8: Add a new user

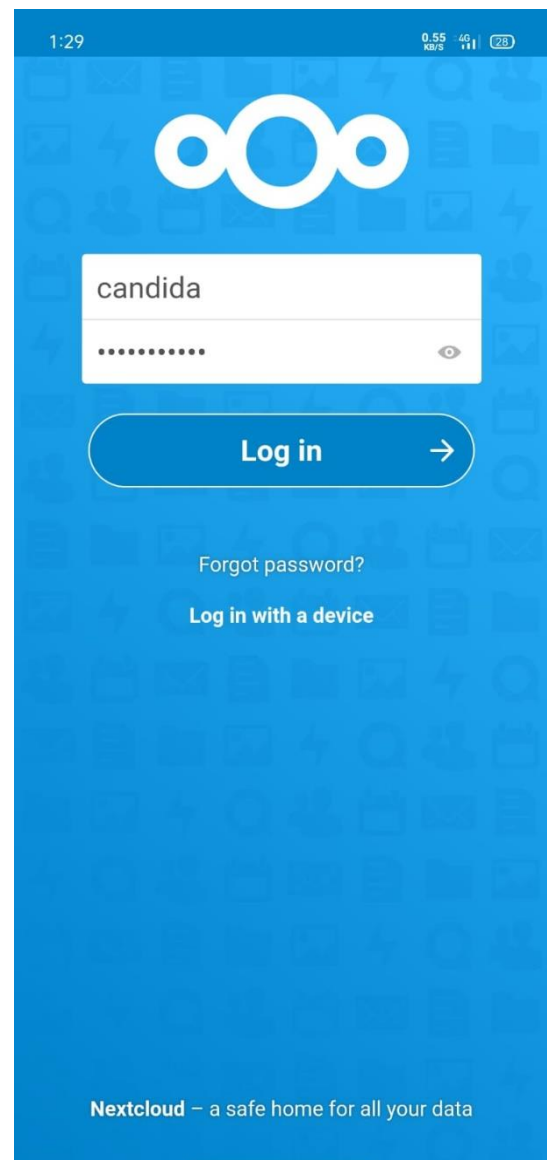
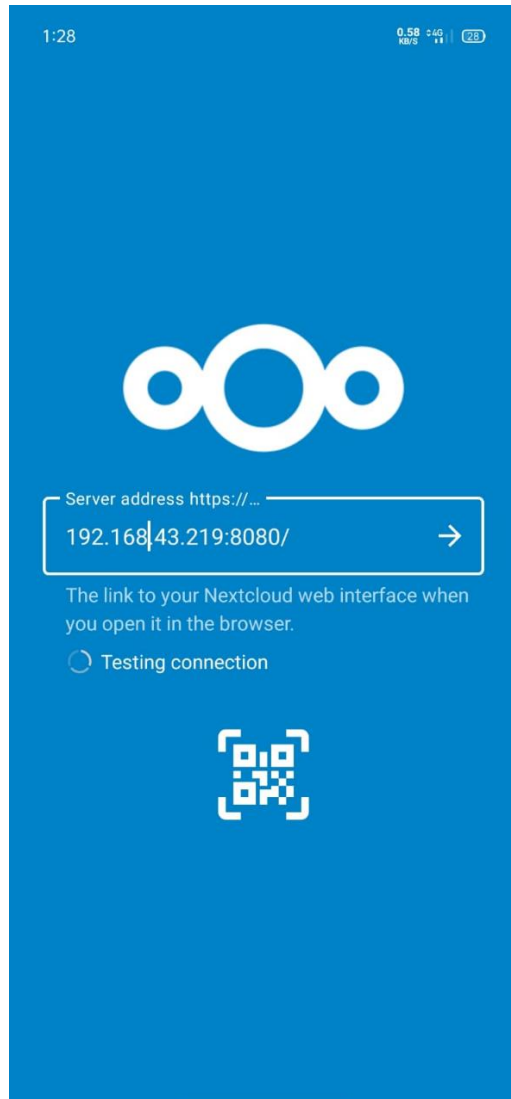
Go to the Users Page and click “New User”

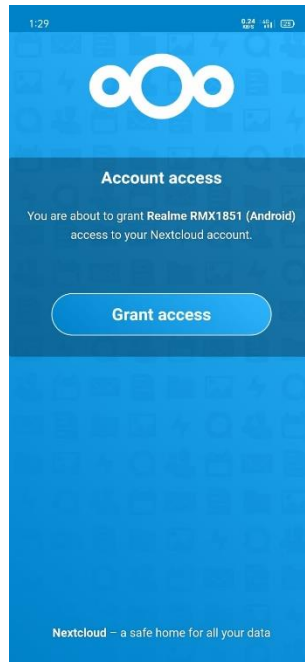
Add the username and password and optionally select a group



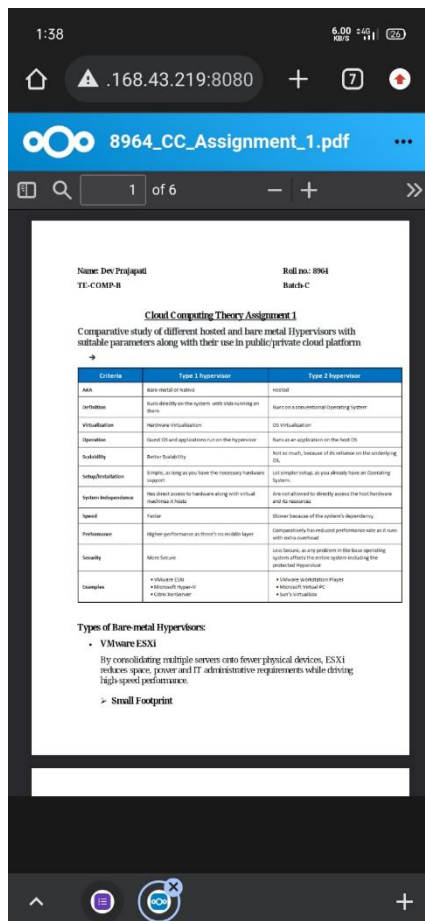
Step 9: Using Nextcloud on Android

Install the Nextcloud apk from the play store and login as a user, with the server address as <https://192.168.43.219:8080/>

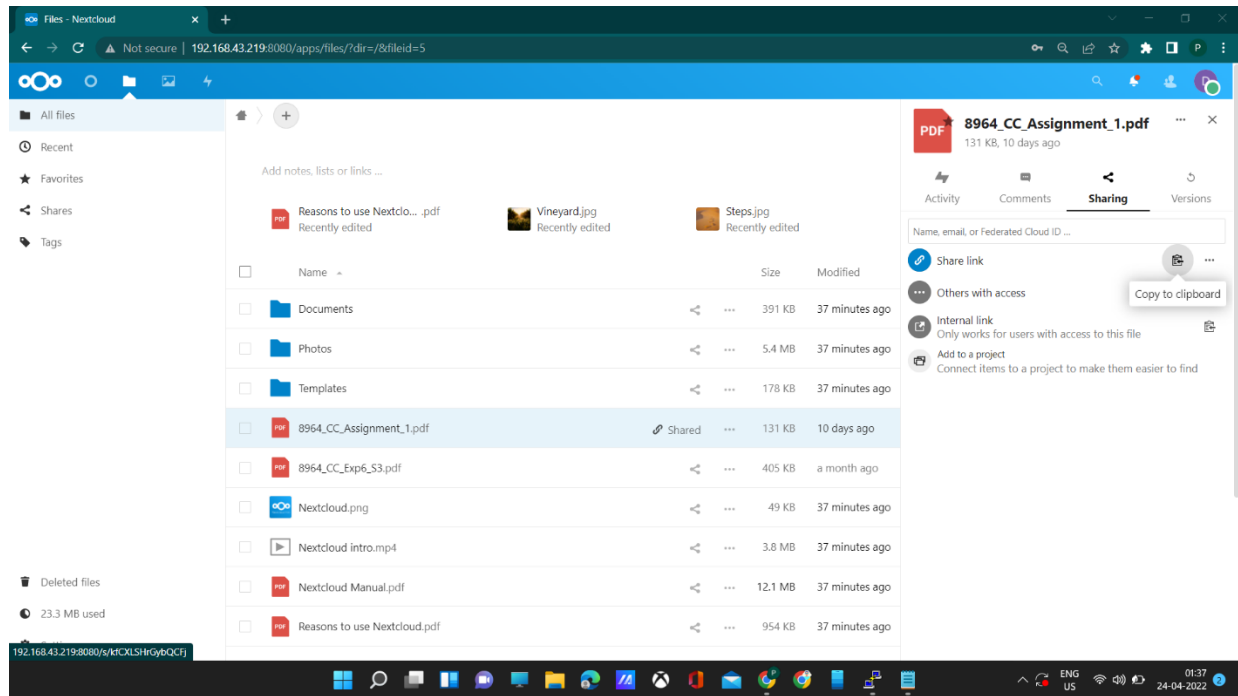




One can Upload files from mobile and will be stored on the Nextcloud cloud database



One can share the file using a URL link and others can download/view it from any device if they have nextcloud installed and they have logged as a user or admin



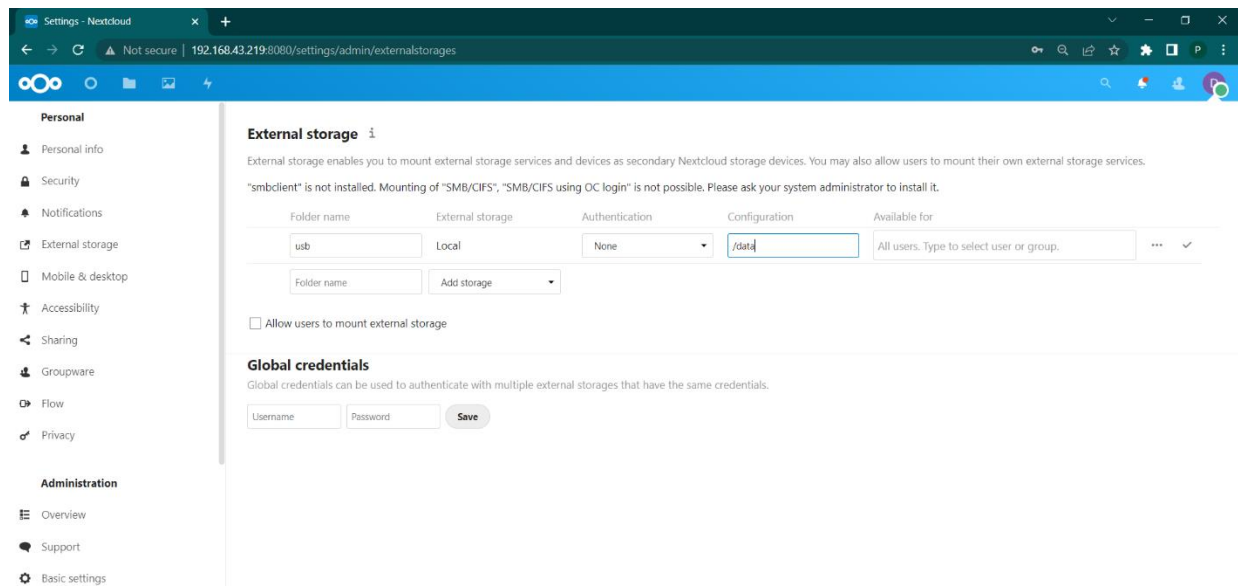
Step 10: Adding an external hard drive.

- Connect it to the USB Ports of Raspberry Pi and switch back to an SSH Session
- Find all the partition IDs by entering – `lsblk`
- Format it using – `sudo mkfs.ext4 /dev/sda`
- Create a mount point for the drive – `sudo mkdir /media/usbdrive`
- Mount the drive with – `sudo mount /dev/sda /media/usbdrive`
- Finally, we need to make sure that the default user for web servers, [www-data](#) has access to this mounted folder – `sudo chown -R www-data:www-data /media/usbdrive`

```
clear
lsblk
sudo mkfs.ext4 /dev/sda
sudo mkdir /media/usbdrive
sudo mount /dev/sda /media/usbdrive
sudo chown -R www-data:www-data /media/usbdrive
```

- Now we have to tell the Nextcloud container about this folder – `docker stop nextcloud`
&& `docker rm nextcloud`
- Run Nextcloud with USB drive mounted location

`docker run --name nextcloud -d -p 8080:80 -v /media/usbdrive:/data --network nextcloud-net -v /home/pi/nextcloud:/var/www/html nextcloud`
- Go to Apps and Enable External Storage
- In the Administration Settings click on External Storages
- Add Storages Local and add the directory path
- All users should now be able to see and write to this external storage



Conclusion:

This design gives the freedom to use storage space and savings through strong system control. We configure a private cloud on Raspberry Pi and then provide services to use private storage on similar lines with Dropbox and Google Drive. Thus, the confidentiality and integrity of information or data are improved. The client or user store on the server is not reliably maintained in all periods. The first method seems to be expensive, therefore configuring a server on our own raspberry module seems to be a better option for our private storage. The design also aims to provide reliable cloud services for personal use and make them a long-term, cost-effective, personal cloud storage product.

References:

- <https://youtu.be/eCJA1F72izc>
- <https://youtu.be/CHWHQFwxFcE>
- <https://github.com/docker/docker-install>
- <https://github.com/dani-garcia/vaultwarden/wiki/Private-CA-and-self-signed-certs-that-work-with-Chrome>