

Private Cloud Media Server Project

Advanced Implementation and System Administration Report

Prepared by: Hamza Hssaini

Date: August 05, 2025

[LinkedIn](#)

Abstract

This project demonstrates the deployment of a private cloud media server using VirtualBox, Ubuntu Server, and Nextcloud. It includes secure HTTPS access, external storage integration, real-time document editing with OnlyOffice, user and group access controls, automated backups with email alerts, and monitoring with Netdata. The implementation simulates real-world enterprise IT operations and showcases skills in Linux system administration, service automation, Docker, and secure network configuration.

Private Cloud Media Server Project

Abstract.....	2
Table De Figures :	6
Chapter 1: Introduction	7
1.1 Project Description.....	7
1.2 Objectives	7
1.3 Scope.....	7
1.4 System Architecture Diagram	7
Chapter 2: Implementation	8
2.1 System Setup.....	8
2.1.1 Prerequisites:.....	8
2.1.2 Creation the Virtual Machine:	8
.....	9
2.1.3 Install Ubuntu Server (22.04)	10
2.2 Server Configuration	14
2.2.1 Install LAMP Stack (Apache, MariaDB, PHP)	14
2.2.2 Install Apache Web Server	14
2.2.3 Install MariaDB (Database Server)	15
2.2.4 Install PHP and Required Modules	17
2.2.5 Install & Configure Nextcloud	18
2.2.5.1 Prerequisites Recap	18
2.2.5.2 Create MySQL Database for Nextcloud	18
2.2.5.3 Download and Install Nextcloud.....	19
2.2.5.4 Apache Configuration for Nextcloud:	20
2.2.5.5 Access Nextcloud in Browser	21
2.2.5.6 (Optional) Cleanup phpinfo page.....	22
2.3 Advanced Features	22
2.3.1 Enable HTTPS (Self-Signed Certificate) in Nextcloud	22
2.3.1.1 Create SSL Certificate and Key	22
2.3.1.2 Create an SSL Virtual Host for Apache	23

2.3.1.3 Enable SSL Module and Site.....	24
2.3.1.4 Test HTTPS Access	24
2.3.2 Connect External Storage in Nextcloud	25
2.3.2.1 Create a Directory for External Storage	25
2.3.2.2 Enable the “External Storage Support” App	26
2.3.2.3 Add Local External Storage	27
2.3.2.4 Test the Mount	27
2.3.3 Create Multiple Users and Sharing Rules	29
2.3.3.1 Create Multiple Users.....	29
2.3.3.2 Organize Users into Groups	30
2.3.3.3 Test File Sharing Rules	30
2.3.3.3 Login as Another IT User (employee1)	31
2.3.3.4 Login as HRuser (Not in IT group)	32
2.3.3.5 Summary: What This Feature Demonstrates.....	32
2.3.4 Set Up Scheduled Backups for Nextcloud	33
2.3.4.1 Create a Backup Script.....	33
2.3.4.2 Test the Script Manually.....	34
2.3.4.3 Schedule the Backup with Cron	34
2.3.4.4 Add Email Notifications via Gmail SMTP.....	35
2.3.4.5 Verification Email backup success From my Phone:.....	36
2.3.4.6 Report About Protocol SMTP :	36
2.3.5 Enable Monitoring & Logging for Nextcloud	37
2.3.5.1 System Monitoring with Netdata	37
2.3.5.1 Log Monitoring	38
2.3.6 Real-Time Document Editing with OnlyOffice or Collabora	39
2.3.6.1 <i>Install and Run ONLYOFFICE Document Server.....</i>	40
2.3.6.2 Verify ONLYOFFICE Document Server Is Running	41
2.3.6.3 Configure Nextcloud to Use ONLYOFFICE	41

2.3.6.4 Test Document Editing.....	42
Chapter 3: Troubleshooting Notes	43
Note1: Solving Post-Installation Access Issues in Nextcloud	43
1.1 Overview of the 3 Problems and Their Fixes	43
1.1.1 Incorrect Domain Redirection	43
1.1.2 Apache Showing Default Page Instead of Nextcloud	43
1.1.3 Access Through Untrusted Domain	44
Note2 : msmtpt: cannot log to ...: cannot open: Permission denied	45
Note3 : ONLYOFFICE Document Server must be accessible via HTTPS.....	45
Conclusion	46
Skills Demonstrated	46

Table De Figures :

Figure 1: Private Cloud Media Server Architecture-----	7
Figure 2: Apache2 Web Server Running on media-server -----	15
Figure 3: Browser showing PHP Info page -----	18
Figure 4: Nextcloud MySQL Database and User Created-----	19
Figure 5: Nextcloud Downloaded and Moved to Web Root -----	20
Figure 6:Apache Virtual Host Configured for Nextcloud -----	21
Figure 7: Nextcloud Web Installer -----	22
Figure 8:Generating Self-Signed SSL Certificate for Nextcloud -----	23
Figure 9:Creating Apache SSL Virtual Host for Nextcloud-----	24
Figure 10:Enabling SSL Module and Virtual Host in Apache -----	24
Figure 11:Nextcloud Dashboard Loaded via HTTPS -----	25
Figure 12:Creating External Storage Directory in Ubuntu Server-----	26
Figure 13: Enabling External Storage Support in Nextcloud-----	26
Figure 14: Configuring External Storage in Nextcloud -----	27
Figure 15: Uploading a File to External Storage -----	28
Figure 16: Admin creating users -----	29
Figure 17: Group assignment-----	30
Figure 18: sharing interface with permissions -----	31
Figure 19: shared folder appearing in employe11's account -----	31
Figure 20: HRuser cannot see the folder -----	32
Figure 21:the script file in Nano editor and the backup folder structure -----	34
Figure 22: list of generated backup files with timestamps -----	34
Figure 23: Verification Email backup success From my Phone -----	36
Figure 24 : Netdata Dashboard-----	37
Figure 25: htop dashboard-----	38
Figure 26: docker run command and running container-----	41
Figure 27: Nextcloud ONLYOFFICE app settings filled Success -----	42
Figure 28: ONLYOFFICE document editor with active collaborative editing -----	42

Chapter 1: Introduction

1.1 Project Description

Building a private cloud media server using VirtualBox, Ubuntu Server, and Nextcloud to simulate enterprise file sharing, collaboration, monitoring, and backups — all within a secure and virtualized local environment.

1.2 Objectives

- Host a Nextcloud server locally
- Enable HTTPS, file sharing, and document editing
- Add monitoring and backups
- Simulate enterprise admin tasks

1.3 Scope

- Local VM-based deployment using VirtualBox
- LAMP stack, MariaDB, Nextcloud, OnlyOffice
- External storage integration
- Netdata monitoring, automated backups with email alerts

1.4 System Architecture Diagram

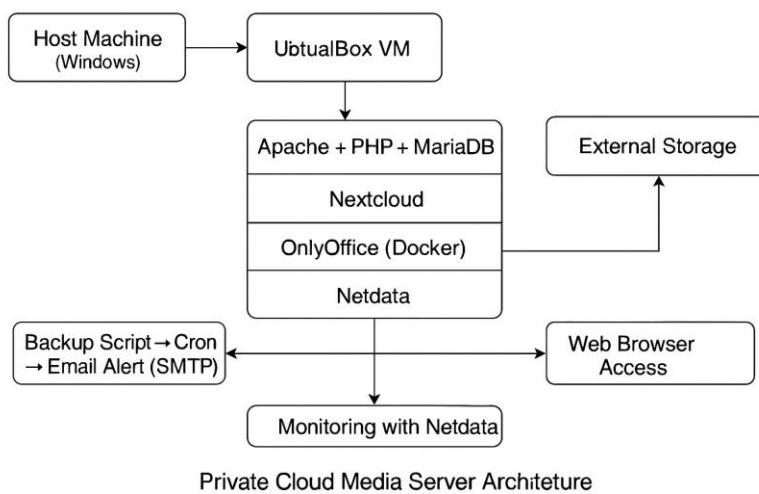


Figure 1: Private Cloud Media Server Architecture

Chapter 2: Implementation

2.1 System Setup

Install **Ubuntu Server 22.04 LTS** in **VirtualBox**, which will act as your **private cloud media server**.

2.1.1 Prerequisites:

- ✓ *Download VirtualBox*

⌚ <https://www.virtualbox.org/wiki/Downloads>

Install the latest version based on your OS (Windows/macOS/Linux).

- ✓ *Download Ubuntu Server ISO (22.04 LTS)*

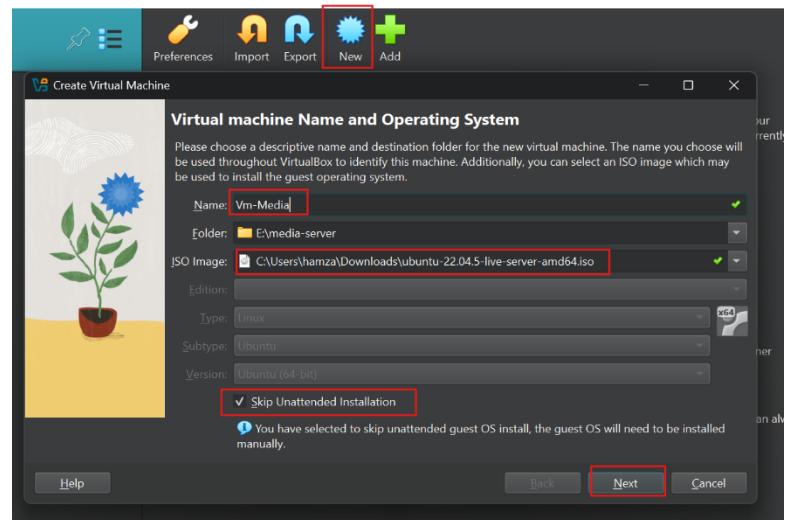
⌚ <https://ubuntu.com/download/server>

- File: ubuntu-22.04.4-live-server-amd64.iso
- Save it somewhere safe — we'll use this ISO to boot the VM.

2.1.2 Creation the Virtual Machine:

Open VirtualBox → Click "New"

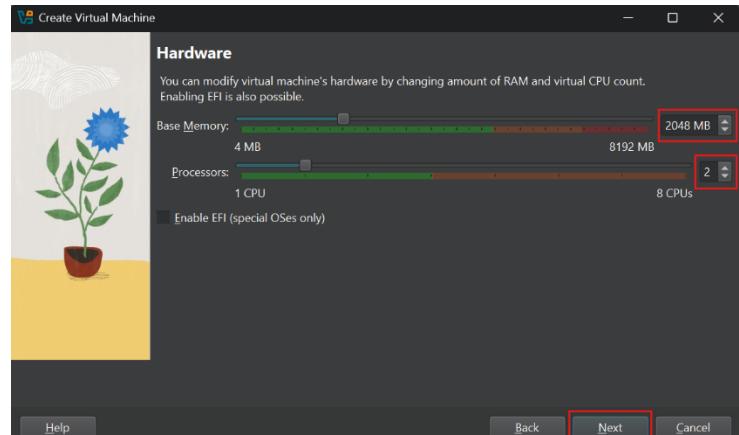
Field	Value
Name	media-vm
Type	Linux
Version	Ubuntu (64-bit)



Private Cloud Media Server Project

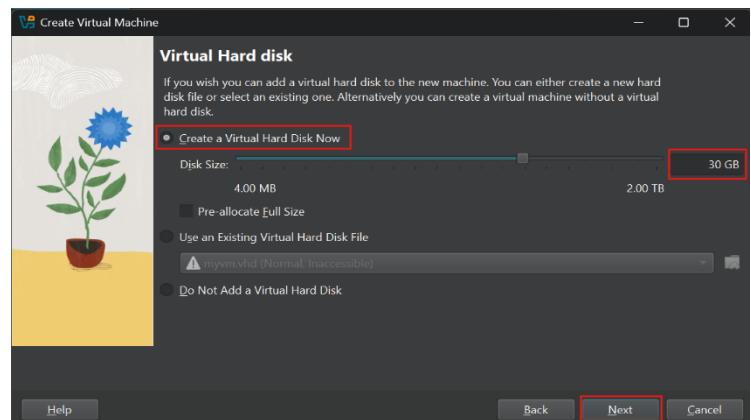
Configure hardware

Setting	Value
RAM	2048 MB (minimum)
CPU	2 CPUs
Disk	30 GB (VDI, dynamically allocated)



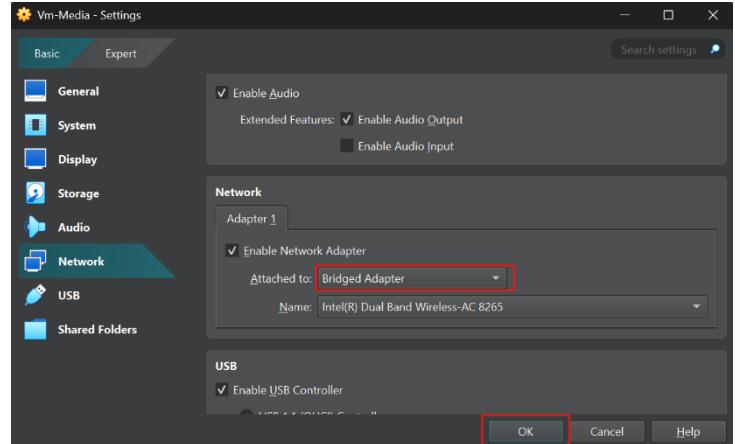
Mount Ubuntu ISO:

- In **Settings > Storage**
- Under **Controller: IDE**, click **Empty**
- Click CD icon on right → **Choose a disk file**
- Select the **Ubuntu Server ISO** you downloaded



Network Configuration

- In **Settings > Network**
- Adapter 1: **Bridged Adapter**

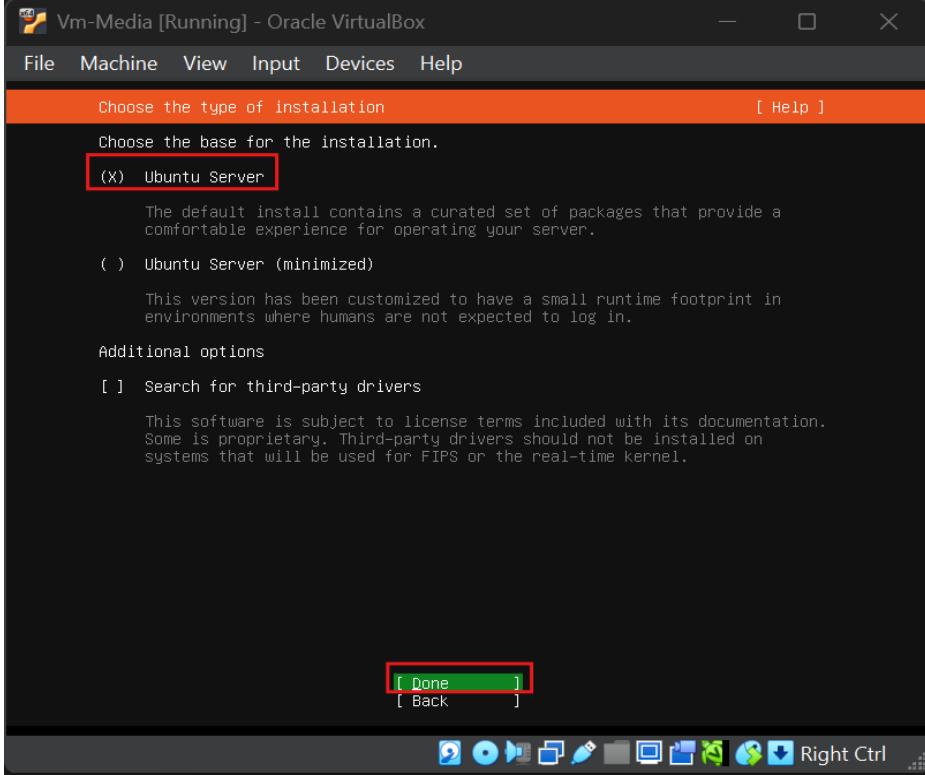


Start the VM

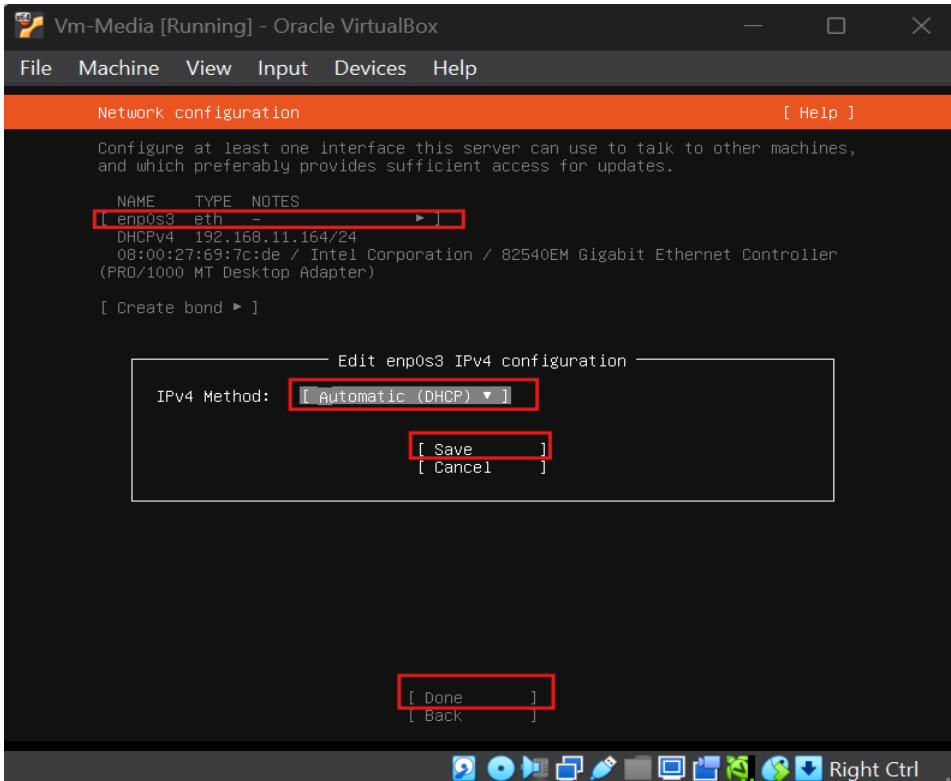
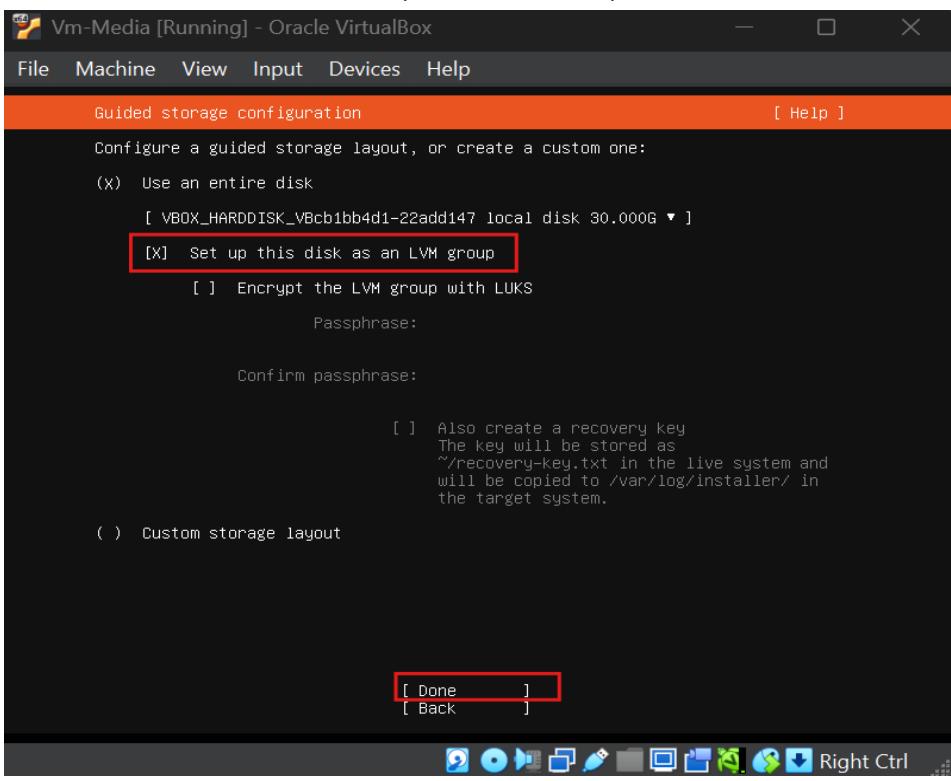
Click **Start**, and you'll enter the Ubuntu installer.

2.1.3 Install Ubuntu Server (22.04)

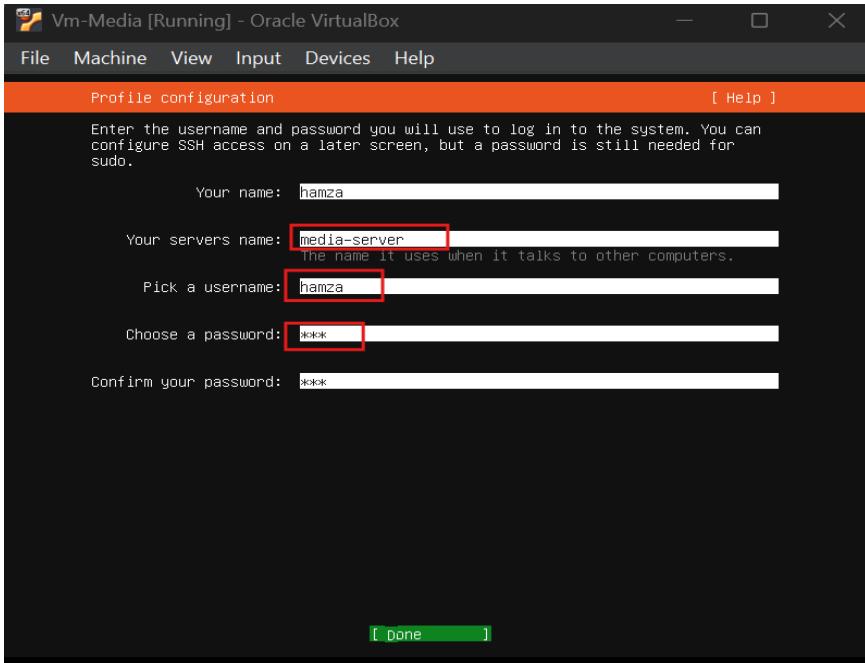
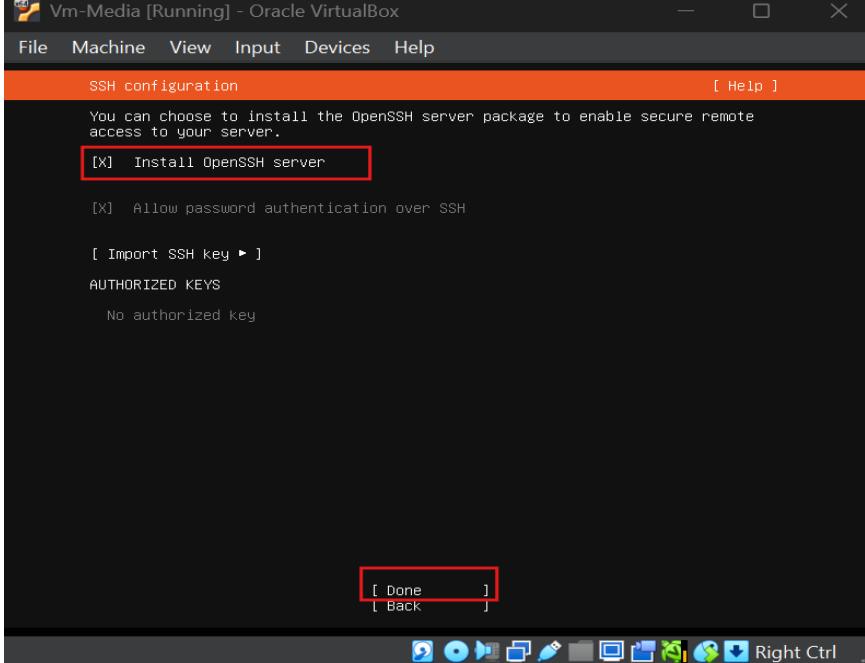
Follow these steps in the installer:

Step	Instruction
Language	Choose English (or your preferred)
Install Type	Select Ubuntu Server 

Private Cloud Media Server Project

Network	Leave as default (automatic IP by DHCP) 
Storage	Use the entire disk with LVM (recommended) 

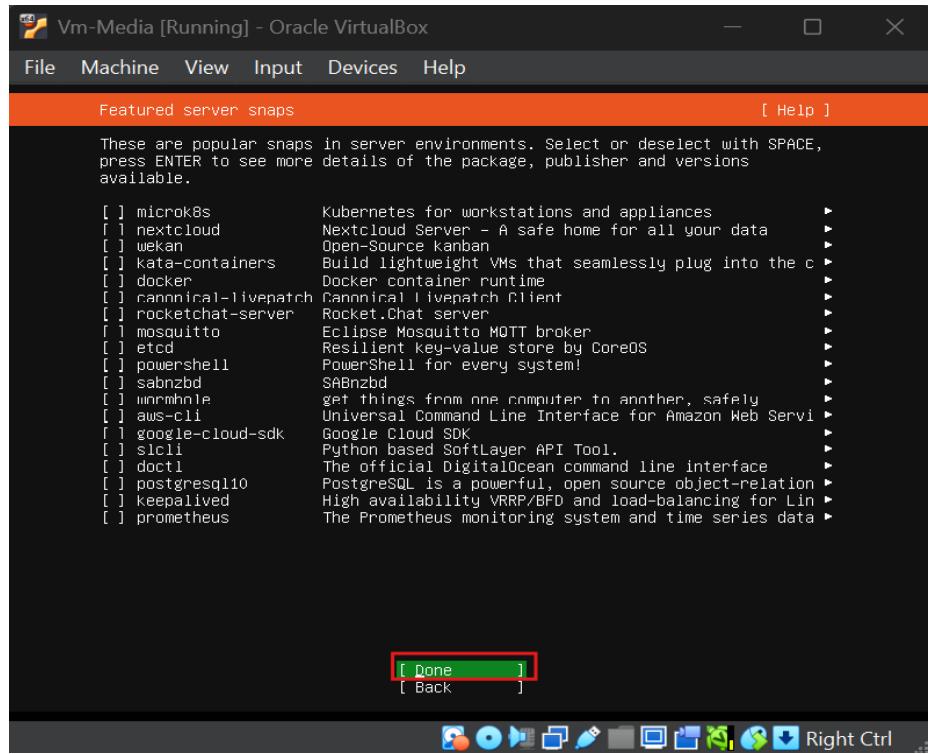
Private Cloud Media Server Project

Profile	Username: hamza, Password: your_password 
Server Name	media-server
SSH	<input checked="" type="checkbox"/> Install OpenSSH server 

Private Cloud Media Server Project

Snap Apps

✖ Skip unless needed



After setup, reboot the VM and **remove the ISO** from VirtualBox to avoid boot looping.

Once your VM is running and you see a login prompt like:

media-server login:

```
----END SSH HOST KEY KEYS----  
[ 39.544031] cloud-init[1425]: Cloud-init v. 25.1.4-0ubuntu0~22.04.1 finished at Wed, 30 Jul 2025  
14:49:35 +0000. Datasource DataSourceNone. Up 39.53 seconds  
[ OK ] Finished Cloud-init: Final Stage.  
[ OK ] Reached target Cloud-init target.  
  
media-server login:  
media-server login: _
```

After enter login and password:

```
System load: 0.01 Processes: 116  
Usage of /: 36.3% of 13.67GB Users logged in: 0  
Memory usage: 11% IPv4 address for enp0s3: 192.168.11.174  
Swap usage: 0%  
  
Expanded Security Maintenance for Applications is not enabled.  
50 updates can be applied immediately.  
To see these additional updates run: apt list --upgradable  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*copyright.  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
hamza@media-server:"$
```

2.2 Server Configuration

2.2.1 Install LAMP Stack (Apache, MariaDB, PHP)

Nextcloud needs:

- **Apache** (web server)
- **MariaDB** (or MySQL)
- **PHP** (with specific modules)

We'll install and configure all of this manually — this gives you **full control**.

2.2.2 Install Apache Web Server

sudo apt install apache2 -y

 After install:

sudo systemctl status apache2

```
hamza@media-server:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
  Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
  Active: active (running) since Wed 2025-07-30 15:06:58 UTC; 1min 26s ago
    Docs: https://httpd.apache.org/docs/2.4/
 Main PID: 2555 (apache2)
   Tasks: 55 (limit: 2220)
  Memory: 5.1M
     CPU: 260ms
  CGroup: /system.slice/apache2.service
          ├─2555 /usr/sbin/apache2 -k start
          ├─2557 /usr/sbin/apache2 -k start
          └─2558 /usr/sbin/apache2 -k start

Jul 30 15:06:58 media-server systemd[1]: Starting The Apache HTTP Server...
Jul 30 15:06:58 media-server apachectl[2554]: AH00558: apache2: Could not reliably determine the server's fully qualified name, using 127.0.0.1 for Port 80
Jul 30 15:06:58 media-server systemd[1]: Started The Apache HTTP Server.
lines 1-16/16 (END)
```

Private Cloud Media Server Project

You should see it running.

Test in your browser:

👉 Visit <http://<your-server-ip>>

You should see the "Apache2 Ubuntu Default Page"

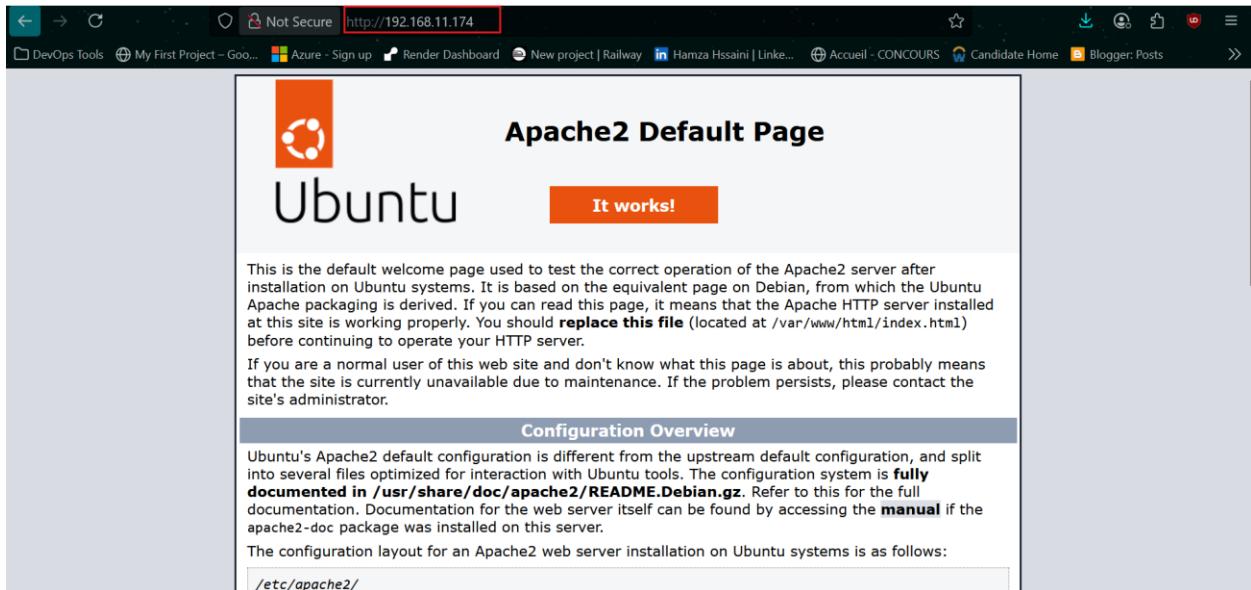


Figure 2: Apache2 Web Server Running on media-server

2.2.3 Install MariaDB (Database Server)

```
sudo apt install mariadb-server -y
```

Then secure the installation:

```
sudo mysql_secure_installation
```

During prompts:

- Set root password? **Yes**
- Remove anonymous users? **Yes**
- Disallow root login remotely? **Yes**
- Remove test database? **Yes**
- Reload privileges? **Yes**

Private Cloud Media Server Project

```
Disallow root login remotely? [Y/n] y
... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access. This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] y
... Success!

Cleaning up...

All done! If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```

✓ Then verify:

```
sudo systemctl status mariadb
```

```
● mariadb.service - MariaDB 10.6.22 database server
   Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2025-07-30 15:13:13 UTC; 4min 27s ago
     Docs: man:mariadb(8)
           https://mariadb.com/kb/en/library/systemd/
  Process: 3578 ExecStartPre=/usr/bin/install -m 755 -o mysql -g root -d /var/run/mysql (code=exited, status=0/SUCCESS)
  Process: 3571 ExecStartPre=/bin/sh -c systemctl unset-environment _WSREP_START_POSITION (code=exited, status=0/SUCCESS)
  Process: 3574 ExecStartPre=/bin/sh -c [ ! -e /usr/bin/galera_recovery ] && VAR= || VAR='/usr/bin/galera_recovery'; [ $? -eq 0 ] && rm -f $VAR
  Process: 3613 ExecStartPost=/bin/sh -c systemctl unset-environment _WSREP_START_POSITION (code=exited, status=0/SUCCESS)
  Process: 3615 ExecStartPost=/etc/mysql/debian-start (code=exited, status=0/SUCCESS)
 Main PID: 3603 (mariadb)
    Status: "Taking your SQL requests now..."
      Tasks: 9 (limit: 14654)
     Memory: 61.7M
        CPU: 2.825s
       CGroup: /system.slice/mariadb.service
                 └─3603 /usr/sbin/mariadb
```

2.2.4 Install PHP and Required Modules

Run this full command:

```
sudo apt install php8.1 libapache2-mod-php8.1 php8.1-mysql php8.1-gd php8.1-curl  
php8.1-xml php8.1-mbstring php8.1-zip php8.1-intl php8.1-bcmath php8.1-imagick -y
```

After Installation:

Verify PHP is working:

```
php -v
```

```
hamza@media-server:~$ php -v  
PHP 8.1.2-1ubuntu2.22 (cli) (built: Jul 15 2025 12:11:22) (NTS)  
Copyright (c) The PHP Group  
Zend Engine v4.1.2, Copyright (c) Zend Technologies  
    with Zend OPcache v8.1.2-1ubuntu2.22, Copyright (c), by Zend Technologies  
hamza@media-server:~$ -
```

Restart Apache:

```
sudo systemctl restart apache2
```

Create a PHP test page:

```
echo "<?php phpinfo(); ?>" | sudo tee /var/www/html/info.php
```

```
hamza@media-server:~$ echo "<?php phpinfo(); ?>  
" | sudo tee /var/www/html/info.php  
<?php phpinfo(); ?>
```

Open in your browser:

```
http://<your-server-ip>/info.php
```

Private Cloud Media Server Project

The screenshot shows a browser window with the URL `http://192.168.11.174/info.php` highlighted in red. The page content is the PHP Info page, which provides detailed information about the PHP installation. Key sections include:

- System**: Linux media-server 5.15.0-151-generic #161-Ubuntu SMP Tue Jul 22 14:25:40 UTC 2025 x86_64
- Build Date**: Jul 15 2025 12:11:22
- Build System**: Linux
- Server API**: Apache 2.0 Handler
- Virtual Directory Support**: disabled
- Configuration File (php.ini) Path**: /etc/php/8.1/apache2
- Loaded Configuration File**: /etc/php/8.1/apache2/php.ini
- Scan this dir for additional .ini files**: /etc/php/8.1/apache2/conf.d
- Additional .ini files parsed**: A long list of configuration files including /etc/php/8.1/apache2/conf.d/10-mysqlind.ini, /etc/php/8.1/apache2/conf.d/10-opcache.ini, /etc/php/8.1/apache2/conf.d/10-pdo.ini, /etc/php/8.1/apache2/conf.d/15-xml.ini, /etc/php/8.1/apache2/conf.d/20-bcmath.ini, /etc/php/8.1/apache2/conf.d/20-calendar.ini, /etc/php/8.1/apache2/conf.d/20-curl.ini, /etc/php/8.1/apache2/conf.d/20-dom.ini, /etc/php/8.1/apache2/conf.d/20-ctype.ini, /etc/php/8.1/apache2/conf.d/20-exif.ini, /etc/php/8.1/apache2/conf.d/20-filinfo.ini, /etc/php/8.1/apache2/conf.d/20-gd.ini, /etc/php/8.1/apache2/conf.d/20-gettext.ini, /etc/php/8.1/apache2/conf.d/20-iconv.ini, /etc/php/8.1/apache2/conf.d/20-imaggick.ini, /etc/php/8.1/apache2/conf.d/20-intl.ini, /etc/php/8.1/apache2/conf.d/20-mbstring.ini, /etc/php/8.1/apache2/conf.d/20-myssql.ini, /etc/php/8.1/apache2/conf.d/20-pdo_mysql.ini, /etc/php/8.1/apache2/conf.d/20-phar.ini, /etc/php/8.1/apache2/conf.d/20-posix.ini, /etc/php/8.1/apache2/conf.d/20-readline.ini, /etc/php/8.1/apache2/conf.d/20-shmop.ini, /etc/php/8.1/apache2/conf.d/20-simplexini.ini, /etc/php/8.1/apache2/conf.d/20-sysvsem.ini, /etc/php/8.1/apache2/conf.d/20-sysvshm.ini, /etc/php/8.1/apache2/conf.d/20-tokenizer.ini, /etc/php/8.1/apache2/conf.d/20-xmleader.ini, /etc/php/8.1/apache2/conf.d/20-xmlwriter.ini, /etc/php/8.1/apache2/conf.d/20-xsl.ini, /etc/php/8.1/apache2/conf.d/20-zip.ini
- PHP API**: 20210902
- PHP Extension**: 20210802
- Zend Extension**: 420210902

Figure 3: Browser showing PHP Info page

2.2.5 Install & Configure Nextcloud

2.2.5.1 Prerequisites Recap

You should already have:

- Apache installed
- MariaDB installed & secured
- PHP 8.1 and modules installed
- Apache working at: `http://<your-vm-ip>`
- SSH access from your Windows machine
- `phpinfo()` working (optional)

2.2.5.2 Create MySQL Database for Nextcloud



`sudo mysql -u root -p`

(Enter your MariaDB root password)

Then run each SQL line:

`CREATE DATABASE nextcloud;`

`CREATE USER 'hamza_user'@'localhost' IDENTIFIED BY 'Hamza@123321';`

`GRANT ALL PRIVILEGES ON nextcloud.* TO 'hamza_user'@'localhost';`

Private Cloud Media Server Project

FLUSH PRIVILEGES;

EXIT;

```
hamza@media-server:~$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 37
Server version: 10.6.22-MariaDB-0ubuntu0.22.04.1 Ubuntu 22.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE nextcloud;
Query OK, 1 row affected (0.012 sec)

MariaDB [(none)]> CREATE USER 'hamza_user'@'localhost' IDENTIFIED BY 'Hamza@123321;';
    -> ;
Query OK, 0 rows affected (0.092 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON nextcloud.* TO 'hamza_user'@'localhost';
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'GRANT ALL PRIVILEGES ON nextcloud.* TO 'hamza user'@'localhost'' at line 1
MariaDB [(none)]> GRANT ALL PRIVILEGES ON nextcloud.* TO 'hamza_user'@'localhost';
Query OK, 0 rows affected (0.026 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.025 sec)

MariaDB [(none)]> EXIT;
```

Figure 4: Nextcloud MySQL Database and User Created

2.2.5.3 Download and Install Nextcloud

Commands:

```
cd /tmp
```

```
wget https://download.nextcloud.com/server/releases/latest.zip
```

```
sudo apt install unzip -y
```

```
unzip latest.zip
```

```
sudo mv nextcloud /var/www/html/
```

Set ownership and permissions:

```
sudo chown -R www-data:www-data /var/www/html/nextcloud
```

```
sudo chmod -R 755 /var/www/html/nextcloud
```

Private Cloud Media Server Project

```
namza@media-server:~$ cd /tmp
namza@media-server:/tmp$ wget https://download.nextcloud.com/server/releases/latest.zip
--2025-07-31 13:00:24-- https://download.nextcloud.com/server/releases/latest.zip
Resolving download.nextcloud.com (download.nextcloud.com)... 5.9.202.145, 2a01:4f8:210:21c8::145
Connecting to download.nextcloud.com (download.nextcloud.com)|5.9.202.145|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 267161531 (255M) [application/zip]
Saving to: 'latest.zip'

latest.zip          53%[=====] 136.01M 6.71MB/s eta 12s

namza@media-server:/tmp$ sudo mv nextcloud/ /var/www/html/
namza@media-server:/tmp$ sudo chown -R www-data:www-data /var/www/html/nextcloud/
namza@media-server:/tmp$ sudo chmod -R 755 /var/www/html/nextcloud/
namza@media-server:/tmp$ _
```

Figure 5: Nextcloud Downloaded and Moved to Web Root

2.2.5.4 Apache Configuration for Nextcloud:

Create a new Apache config file:

```
sudo nano /etc/apache2/sites-available/nextcloud.conf
```

Paste this config inside:

```
<VirtualHost *:80>

    ServerAdmin admin@local

    DocumentRoot /var/www/html/nextcloud/

    ServerName media.local

<Directory /var/www/html/nextcloud/>

    Options +FollowSymlinks

    AllowOverride All

    Require all granted

</Directory>
```

```
ErrorLog ${APACHE_LOG_DIR}/nextcloud_error.log

CustomLog ${APACHE_LOG_DIR}/nextcloud_access.log combined

</VirtualHost>
```

Private Cloud Media Server Project

Save with CTRL+O, Enter, then CTRL+X.

Enable site & required modules:

```
sudo a2ensite nextcloud.conf
```

```
sudo a2enmod rewrite headers env dir mime
```

```
sudo systemctl reload apache2
```

```
hamza@media-server:/etc/apache2/sites-available$ sudo a2ensite nextcloud.conf
Enabling site nextcloud.
To activate the new configuration, you need to run:
  systemctl reload apache2
hamza@media-server:/etc/apache2/sites-available$ sudo a2enmod rewrite headers env dir mime
Enabling module rewrite.
Enabling module headers.
Module env already enabled
Module dir already enabled
Module mime already enabled
To activate the new configuration, you need to run:
  systemctl restart apache2
hamza@media-server:/etc/apache2/sites-available$ systemctl restart apache2
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ===
Authentication is required to restart 'apache2.service'.
Authenticating as: hamza
Password:
==== AUTHENTICATION COMPLETE ===
hamza@media-server:/etc/apache2/sites-available$ _
```

Figure 6:Apache Virtual Host Configured for Nextcloud

2.2.5.5 Access Nextcloud in Browser

Open in browser:

<http://<your-server-ip>/nextcloud>

You'll see the Nextcloud **setup page**:

- Create an **admin account**
- Fill database info:
 - DB user: hamza_user
 - DB pass: Hamza@123321;
 - DB name: nextcloud
 - Host: localhost

Click **Install**.

Private Cloud Media Server Project

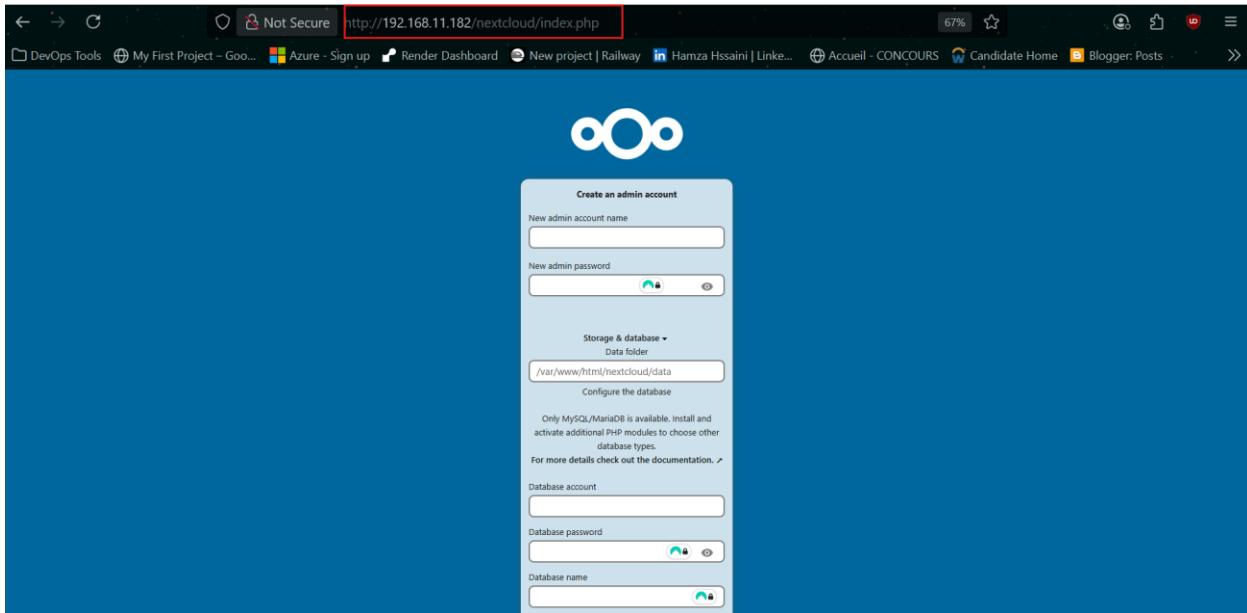


Figure 7: Nextcloud Web Installer

2.2.5.6 (Optional) Cleanup phpinfo page

Remove the test page:

```
sudo rm /var/www/html/info.php
```

2.3 Advanced Features

2.3.1 Enable HTTPS (Self-Signed Certificate) in Nextcloud

To secure Nextcloud on a local server, I generated a self-signed SSL certificate using OpenSSL and configured Apache to serve HTTPS traffic on port 443. Although browsers warn about self-signed certificates, this provides encryption for local testing and development environments.

2.3.1.1 Create SSL Certificate and Key

Run the following command to create a self-signed SSL certificate:

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
-keyout /etc/ssl/private/nextcloud.key \
-out /etc/ssl/certs/nextcloud.crt
```

Private Cloud Media Server Project

```
hamza@media-server:/etc/apache2/sites-available$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
-keyout /etc/ssl/private/nextcloud.key \
-out /etc/ssl/certs/nextcloud.crt
[sudo] password for hamza:
-----
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]:MA
State or Province Name (full name) [Some-State]:fes
Locality Name (eg, city) []:fes
Organization Name (eg, company) [Internet Widgits Pty Ltd]:hamza project lab
Organizational Unit Name (eg, section) []:IT
Common Name (e.g. server FQDN or YOUR name) []:nextcloud
Email Address []:hamzahssaini@gmail.com
hamza@media-server:/etc/apache2/sites-available$ _
```

Figure 8:Generating Self-Signed SSL Certificate for Nextcloud

2.3.1.2 Create an SSL Virtual Host for Apache

Create a new file:

```
sudo nano /etc/apache2/sites-available/nextcloud-ssl.conf
```

Paste this configuration (adjust IP and paths if needed):

```
<VirtualHost *:443>

    ServerAdmin admin@localhost

    ServerName nextcloud

    DocumentRoot /var/www/html/nextcloud

    SSLEngine on

    SSLCertificateFile /etc/ssl/certs/nextcloud.crt

    SSLCertificateKeyFile /etc/ssl/private/nextcloud.key

    <Directory /var/www/html/nextcloud>

        Options +FollowSymlinks

        AllowOverride All

        Require all granted

    </Directory>
```

Private Cloud Media Server Project

```
ErrorLog ${APACHE_LOG_DIR}/nextcloud_error.log  
CustomLog ${APACHE_LOG_DIR}/nextcloud_access.log combined  
</VirtualHost>
```

```
GNU nano 6.2                                         /etc/apache2/sites-available/nextcloud-ssl.conf  
<VirtualHost *:443>  
    ServerAdmin admin@localhost  
    ServerName nextcloud  
  
    DocumentRoot /var/www/html/nextcloud  
  
    SSLEngine on  
    SSLCertificateFile /etc/ssl/certs/nextcloud.crt  
    SSLCertificateKeyFile /etc/ssl/private/nextcloud.key  
  
    <Directory /var/www/html/nextcloud>  
        Options +FollowSymlinks  
        AllowOverride All  
        Require all granted  
    </Directory>  
  
    ErrorLog ${APACHE_LOG_DIR}/nextcloud_error.log  
    CustomLog ${APACHE_LOG_DIR}/nextcloud_access.log combined  
</VirtualHost>
```

Figure 9:Creating Apache SSL Virtual Host for Nextcloud

2.3.1.3 Enable SSL Module and Site

Enable Apache's SSL features and the new virtual host:

```
hamza@media-server:/etc/apache2/sites-available$ sudo a2enmod ssl  
Considering dependency setenvif for ssl:  
Module setenvif already enabled  
Considering dependency mime for ssl:  
Module mime already enabled  
Considering dependency socache_shmcb for ssl:  
Enabling module socache_shmcb.  
Enabling module ssl.  
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self-signed certificates.  
To activate the new configuration, you need to run:  
    systemctl restart apache2  
hamza@media-server:/etc/apache2/sites-available$ sudo a2ensite nextcloud-ssl.conf  
Enabling site nextcloud-ssl.  
To activate the new configuration, you need to run:  
    systemctl reload apache2  
hamza@media-server:/etc/apache2/sites-available$ sudo systemctl reload apache2  
hamza@media-server:/etc/apache2/sites-available$
```

Figure 10:Enabling SSL Module and Virtual Host in Apache

2.3.1.4 Test HTTPS Access

Now go to your browser and open:

<https://nextcloud>

Private Cloud Media Server Project

⚠ You'll see a browser warning like "Connection Not Secure" → click **Advanced > Accept Risk** to continue.

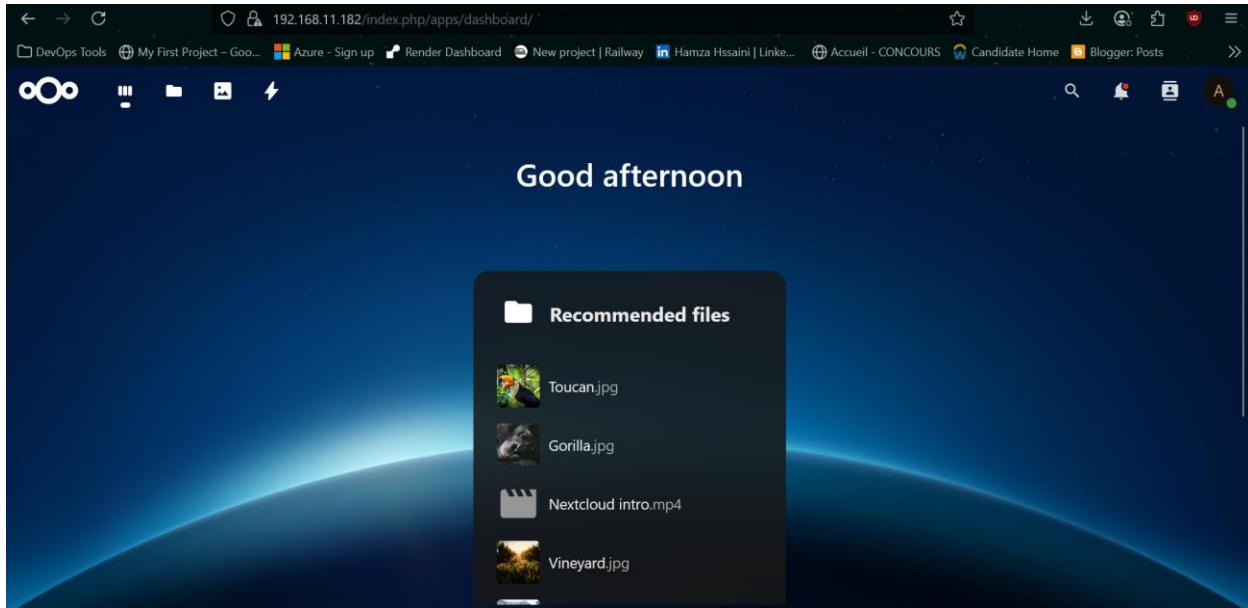


Figure 11:Nextcloud Dashboard Loaded via HTTPS

2.3.2 Connect External Storage in Nextcloud

To simulate enterprise setups, I added an external directory (/mnt/external-storage) to Nextcloud using the "External Storage Support" app. This allows users to access shared disks or mounted network folders. I granted permissions to the admin group and verified uploads were successful inside the external mount.

Simulate how real companies use Nextcloud to **integrate shared storage** like USB drives, NFS shares, or backup disks.

You'll mount a folder on your VM (like a second disk), and connect it to Nextcloud using the "**External Storage Support**" app.

2.3.2.1 Create a Directory for External Storage

In your Ubuntu VM:

```
sudo mkdir /mnt/external-storage
```

```
sudo chown -R www-data:www-data /mnt/external-storage
```

```
sudo chmod -R 770 /mnt/external-storage
```

Private Cloud Media Server Project

This simulates a second drive mounted to your server.

```
hamza@media-server:/etc/apache2/sites-available$ sudo mkdir /mnt/external-storage
[sudo] password for hamza:
hamza@media-server:/etc/apache2/sites-available$ sudo chown -R www-data:www-data /mnt/external-storage
hamza@media-server:/etc/apache2/sites-available$ sudo chmod -R 770 /mnt/external-storage
hamza@media-server:/etc/apache2/sites-available$
```

Figure 12:Creating External Storage Directory in Ubuntu Server

2.3.2.2 Enable the “External Storage Support” App

1. Login to Nextcloud (as admin)
2. Go to: Apps (top right menu > "Apps")
3. Search for:
External storage support
4. Click **Enable**

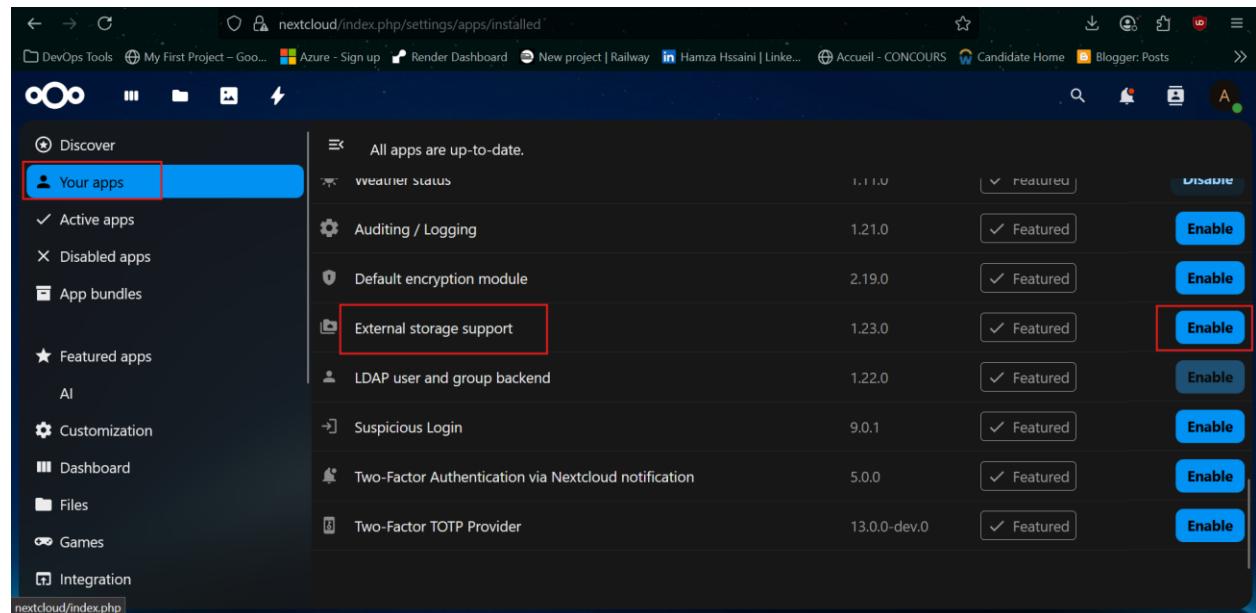


Figure 13: Enabling External Storage Support in Nextcloud

Private Cloud Media Server Project

2.3.2.3 Add Local External Storage

1. Go to:

Settings (bottom left) → Administration → External storages

2. Add a new external storage:

Option	Value
Folder name	ExternalDisk
External storage	Local
Configuration	/mnt/external-storage
Available for	admin or other users/groups

The screenshot shows the Nextcloud administration interface with the URL `nextcloud/index.php/settings/admin/externalstorages` in the address bar. On the left, there's a sidebar with sections like Personal, Administration, and others. The main area is titled "External storage" and contains a table with columns: Folder name, External storage, Authentication, Configuration, and Available for. A row for "ExternalDisk" is selected, highlighted with a red box. The "Folder name" field has "ExternalDisk" entered, "External storage" is set to "Local", "Authentication" is "None", "Configuration" is "/mnt/external-storage", and "Available for" is "All people". Below the table, there's a checkbox for "Allow people to mount external storage". At the bottom, there are "Global credentials" settings and a "Save" button.

Figure 14: Configuring External Storage in Nextcloud

2.3.2.4 Test the Mount

1. Go to the **Files** section of your Nextcloud
2. You should see a new folder named **ExternalDisk**
3. Try uploading a small file into it

Private Cloud Media Server Project

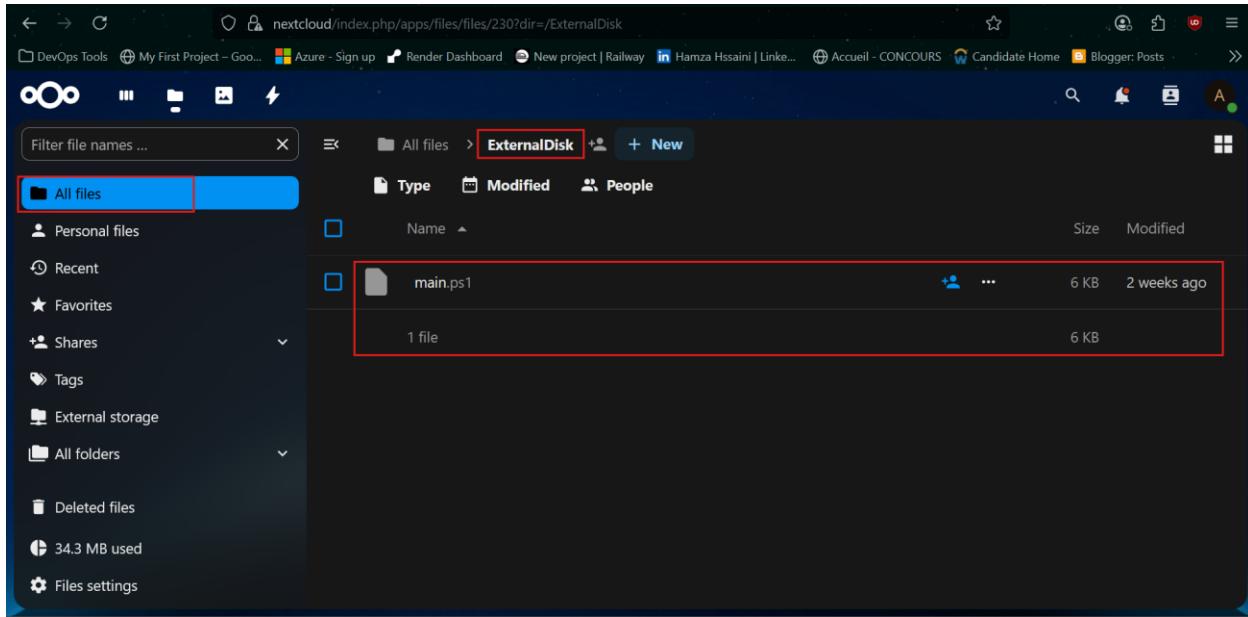
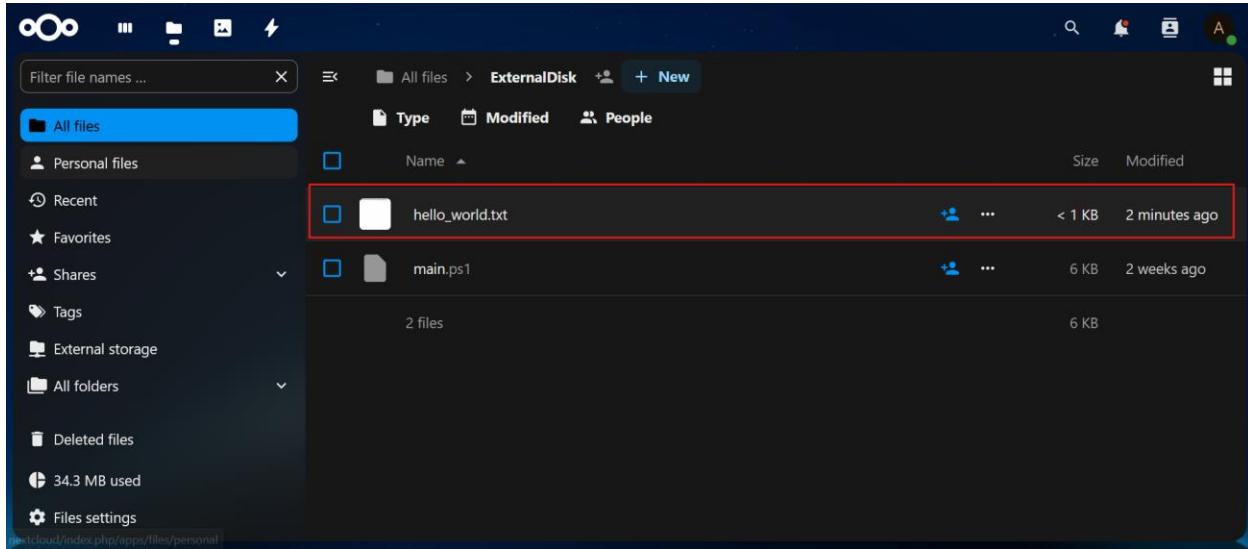


Figure 15: Uploading a File to External Storage

Create file inside external-storage folder from the server:

```
root@media-server:/mnt/external-storage# nano hello_world.txt
root@media-server:/mnt/external-storage#
```

Verification from the page nextcloud:



2.3.3 Create Multiple Users and Sharing Rules

To simulate a real-world business scenario, we implement user and group management in Nextcloud. This includes:

- Creating multiple users
 - Grouping users by department or role
 - Configuring sharing rules and permission levels
- This demonstrates that you can manage user access like an IT administrator in an enterprise.

2.3.3.1 Create Multiple Users

1. Log in to Nextcloud as the **admin** user.
2. Go to the **top-right menu > accounts**.
3. Click “**+ New account**” and fill out:
 - Username: employee1, employee2, etc.
 - Password: (Choose something simple for demo purposes)
 - Group: Type a group name like IT (it will auto-create it if it doesn't exist)
 - Set quota: e.g., 1 GB
4. Click “**Add new account**”.

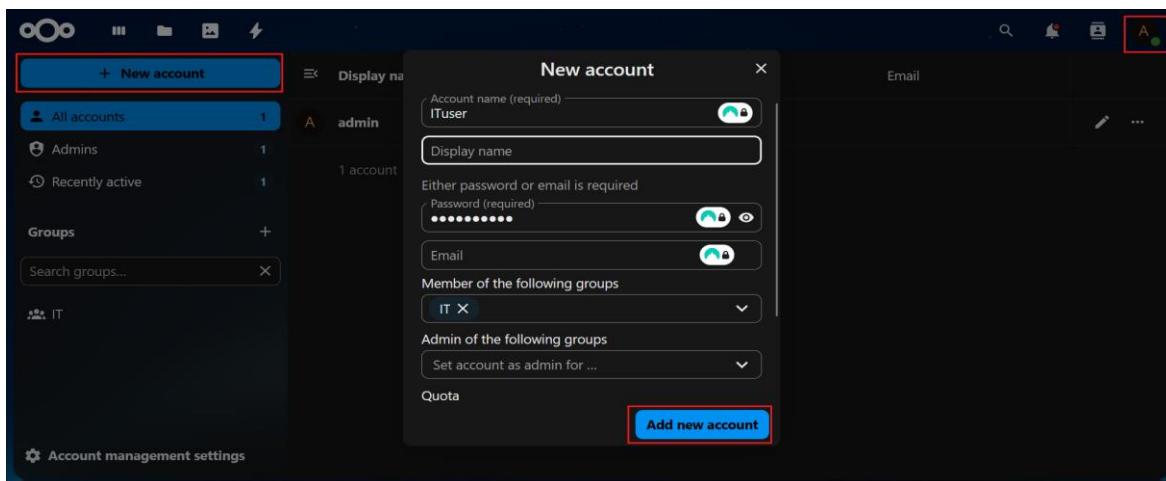


Figure 16: Admin creating users

→ Repeat for multiple users (e.g., ituser, hruser, manager1).

Private Cloud Media Server Project

2.3.3.2 Organize Users into Groups

💡 Why Groups?

In companies, resources are often shared based on departments or roles. Groups make it easier to control access.

1. For each user, assign them to a group:

- IT users → IT
- HR users → HR
- Managers → Managers

The screenshot shows a user management interface with a dark theme. On the left, there's a sidebar with 'All accounts' (5), 'Admins' (1), 'Recently active' (5), 'Groups' (+), and a search bar for 'Search groups...'. Below these are three groups: 'IT' (2), 'HR' (1), and 'Managers' (1). A red box highlights the 'IT' group. The main area displays a table of users with columns: Display name, Account name, Password, and Email. A red box highlights the row for 'employee1'. The table shows the following data:

Display name	Account name	Password	Email
A admin	admin	admin	
E employee1	employee1		
H HRuser	HRuser		
I ITuser	ITuser		
M manager1	manager1		

At the bottom, there's a section for 'Account management settings'.

Figure 17: Group assignment

2.3.3.3 Test File Sharing Rules

- 💡 Login as employee1 (IT group)
 - Go to **Files**.
 - Click “+” → **New Folder**.
 - Name it: Team Files.
- 💡 Share the Folder with a Group
 - Click the **share icon** (right side of the folder).
 - Type **IT** and select the group.
 - Choose permission level:

Private Cloud Media Server Project

- Allow editing
- Deny resharing (optional)
- Deny Deleting

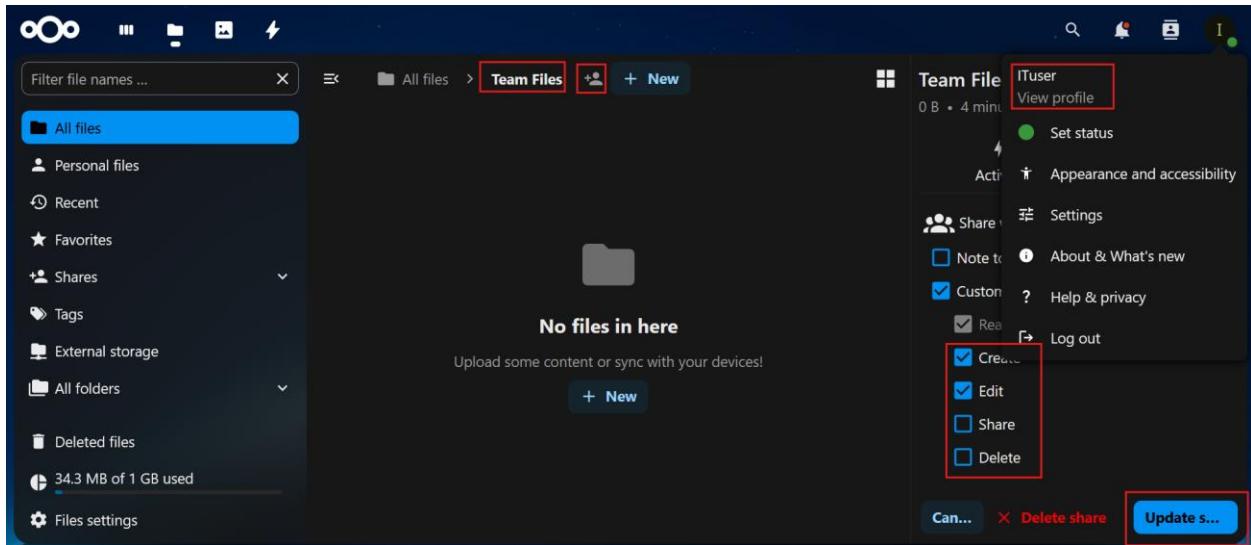


Figure 18: sharing interface with permissions

2.3.3.3 Login as Another IT User (employee1)

- Go to **Files**.
- You should see the Team Files folder shared by ITUser.
- Try uploading, editing, or reading a file based on permissions set.

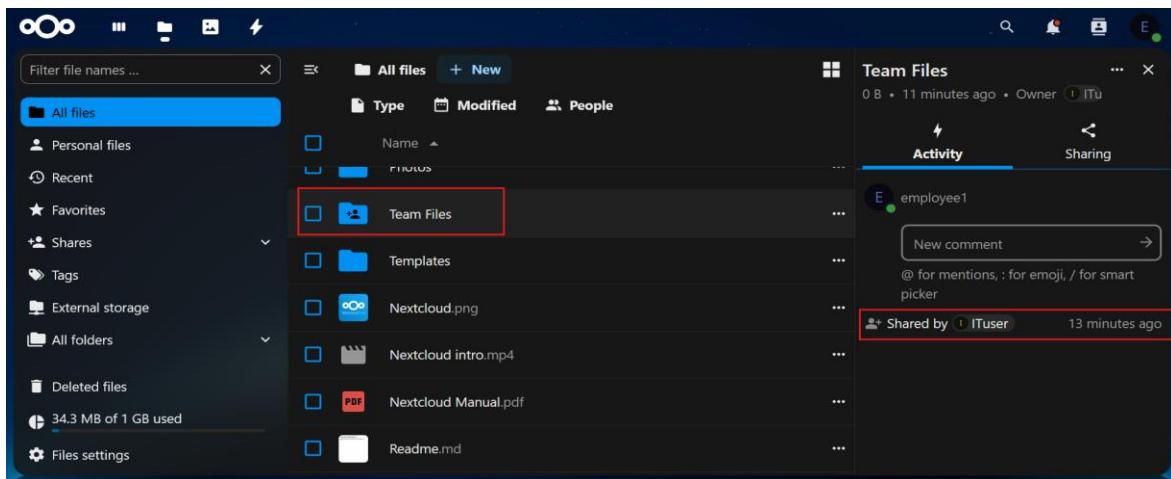


Figure 19: shared folder appearing in employee1's account

Private Cloud Media Server Project

2.3.3.4 Login as HRuser (Not in IT group)

- Confirm that the Team Files folder is **not visible** to unauthorized users.

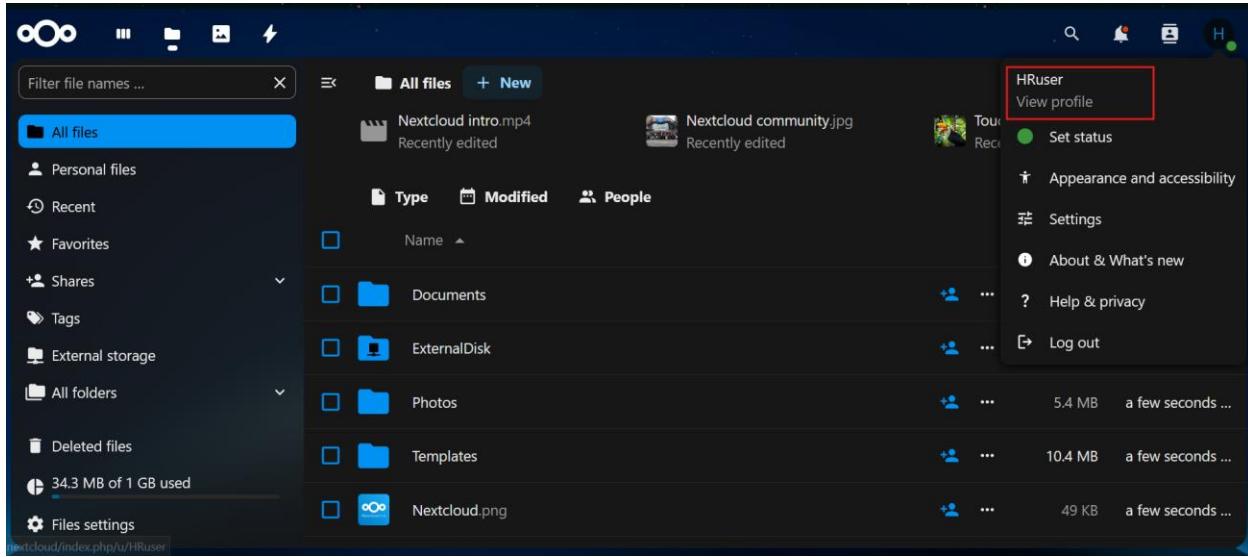


Figure 20: HRuser cannot see the folder

2.3.3.5 Summary: What This Feature Demonstrates

Element	Description
User Management	Ability to create and manage multiple users
Group Assignment	Organize users by departments (IT, HR, etc.)
Permission Control	Fine-grained control of who can see, edit, or share folders
Access Testing	Proved isolation and access per group, like in enterprise environments

2.3.4 Set Up Scheduled Backups for Nextcloud

Ensure that your Nextcloud data and database are backed up **automatically** and **securely** using **cron jobs**. This simulates real-world IT operations where daily backups are critical.

❖ What to Back Up?

You need to back up **two components**:

1. **Nextcloud data folder**

- Usually located at </var/www/html/nextcloud/data/>
- Contains all user-uploaded files

2. **Nextcloud database (MariaDB/MySQL)**

- Contains all configurations, user info, shares, etc.

❖ Tools Used

- cron: To schedule automatic jobs
- tar: To compress folders into archive files
- mysqldump: To export the database
- Optional: rsync or external disk for off-server storage

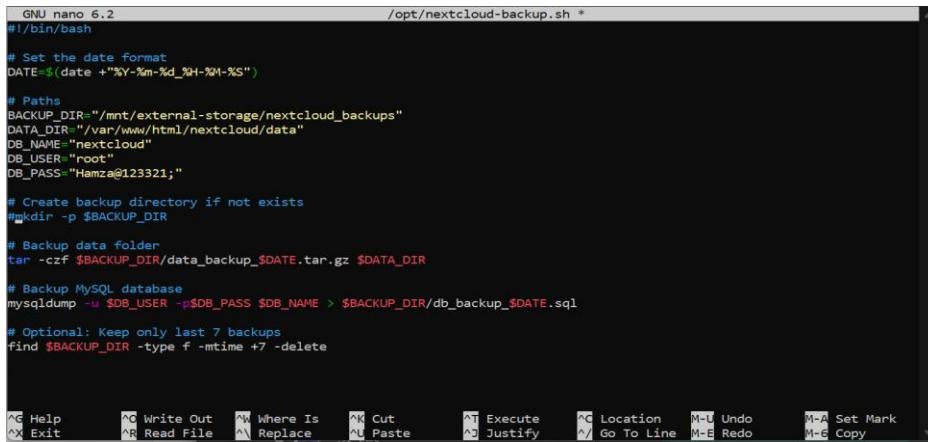
2.3.4.1 Create a Backup Script

Create a script that backs up both the data and database.

```
sudo nano /opt/nextcloud-backup.sh
```

Write the following content:

Private Cloud Media Server Project



```
GNU nano 6.2                               /opt/nextcloud-backup.sh *
#!/bin/bash

# Set the date format
DATE=$(date +"%Y-%m-%d_%H-%M-%S")

# Paths
BACKUP_DIR="/mnt/external-storage/nextcloud_backups"
DATA_DIR="/var/www/html/nextcloud/data"
DB_NAME="nextcloud"
DB_USER="root"
DB_PASS="Hamza@123321"

# Create backup directory if not exists
#mkdir -p $BACKUP_DIR

# Backup data folder
tar -czf $BACKUP_DIR/data_backup_$DATE.tar.gz $DATA_DIR

# Backup MySQL database
mysqldump -u $DB_USER -p$DB_PASS $DB_NAME > $BACKUP_DIR/db_backup_$DATE.sql

# Optional: Keep only last 7 backups
find $BACKUP_DIR -type f -mtime +7 -delete
```

Figure 21:the script file in Nano editor and the backup folder structure

Then make it executable:

```
sudo chmod +x /opt/nextcloud-backup.sh
```

2.3.4.2 Test the Script Manually

Run the backup script:

```
sudo /opt/nextcloud-backup.sh
```

Check your `/mnt/external-storage/nextcloud_backups/` folder for `.tar.gz` and `.sql` files.

```
root@media-server:/mnt/external-storage# cd nextcloud_backups/
root@media-server:/mnt/external-storage/nextcloud_backups# ls
data_backup_2025-08-01_14-33-20.tar.gz  db_backup_2025-08-01_14-33-20.sql
```

Figure 22: list of generated backup files with timestamps

2.3.4.3 Schedule the Backup with Cron

To automate the backup process, I used a **cron job** to schedule the script to run daily or weekly.

🔧 Test Schedule:

First, I scheduled the script to run at 4:00 PM to verify it worked:

```
sudo crontab -e
```

Added:

```
0 16 * * * /opt/nextcloud-backup.sh
```

This means: run the script every day at 4:00 PM.

After confirming success, I updated the schedule to run **every Monday at 8:00 AM**, which is a more realistic production setting:

```
0 8 * * 1 /opt/nextcloud-backup.sh
```

- ⚡ This simulates weekly scheduled backups as done in real companies.

2.3.4.4 Add Email Notifications via Gmail SMTP

To demonstrate real enterprise monitoring practices, I configured the server to send **email alerts** after each backup:

⊕ 🔒 Why Use Gmail SMTP?

Since my server is connected to the internet via a **bridged network adapter**, I used Gmail's SMTP service to reliably send outgoing emails, avoiding ISP blocks on port 25.

⊕ ✎ Setup Summary:

- **Installed SMTP client:**

```
sudo apt install msmtplib -y
```

- **Created Gmail App Password** (for enhanced security)
- **Configured msmtplib** via `/etc/msmtprc`:

```
GNU nano 6.2                               /etc/msmtprc *
defaults
auth      on
tls       on
tls_trust_file /etc/ssl/certs/ca-certificates.crt
#logfile /home/hamza/msmtprc.log

account    gmail
host       smtp.gmail.com
port       587
from      hamzahssaini0@gmail.com
user      hamzahssaini0@gmail.com
password   fmai hmtp gwso npan

account default : gmail
```

- **Modified the backup script**

to send email on success or failure:

```
&& echo "☑ Backup successful at $DATE" | sudo msmtplib $EMAIL \
|| echo "☒ Backup FAILED at $DATE" | sudo msmtplib $EMAIL
```

- ⚡ This allowed me to **receive real-time backup alerts**, just like a production monitoring system.

- **Result:**

- The backup runs every Monday at 8:00 AM.
- Both the data folder and database are backed up.
- Emails are sent automatically to notify about success or failure.
- I also used an external mount point (/mnt/external-storage/) to store the backups, simulating real infrastructure.

2.3.4.5 Verification Email backup success From my Phone:

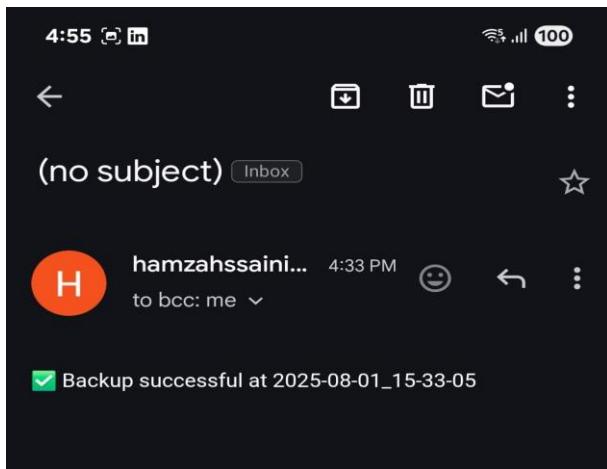


Figure 23: Verification Email backup success From my Phone

2.3.4.6 Report About Protocol SMTP :

To notify me of backup success or failure, I configured email alerts using Gmail SMTP over a secure connection (smtp.gmail.com, port 587 with STARTTLS).

I used Gmail **App Passwords** for authentication and ensured the messages contained **no sensitive content**, only success/failure status.

All communication between my server and Gmail was **encrypted with TLS**, making it safe even over public internet.

In production, more advanced options like **SMTP with GPG-encrypted payloads** or **centralized monitoring/alerting platforms** (e.g., Prometheus + Alertmanager) could be considered.

2.3.5 Enable Monitoring & Logging for Nextcloud

To simulate enterprise-level system monitoring, I integrated various tools to monitor the health, usage, and performance of the Nextcloud environment.

2.3.5.1 System Monitoring with Netdata

I installed [Netdata](#), with these commands if you want to learn more visit this link:

<https://learn.netdata.cloud/docs/netdata-agent/installation/linux>

```
wget -O /tmp/netdata-kickstart.sh https://get.netdata.cloud/
```

```
kickstart.sh && sh /tmp/netdata-kickstart.sh
```

A real-time system monitoring tool that provides a full web dashboard for:

- **CPU, memory, disk usage**
- **Apache activity**
- **Network traffic**
- **Database queries**
- **Live performance alerts**

I accessed Netdata at <http://192.168.11.182:19999> and confirmed it collected over **3,800 live metrics** from my local server.

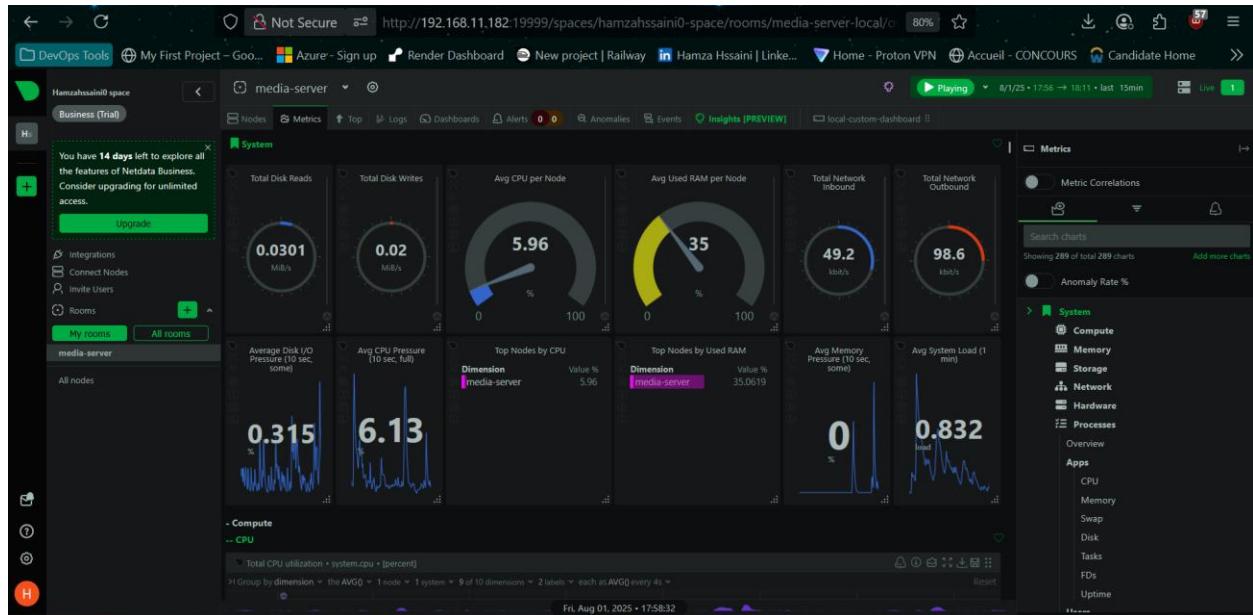


Figure 24 : Netdata Dashboard

2.3.5.1 Log Monitoring

In addition to Netdata, I configured and monitored:

- **Apache Logs** for HTTP and SSL issues

```
sudo tail -f /var/log/apache2/error.log
```

```
sudo tail -f /var/log/apache2/nextcloud_error.log
```

- **Nextcloud Logs** for login failures, file sync errors, etc.

```
sudo tail -f /var/www/html/nextcloud/data/nextcloud.log
```

- System Monitoring with htop

```
sudo apt install htop -y
```

```
htop
```

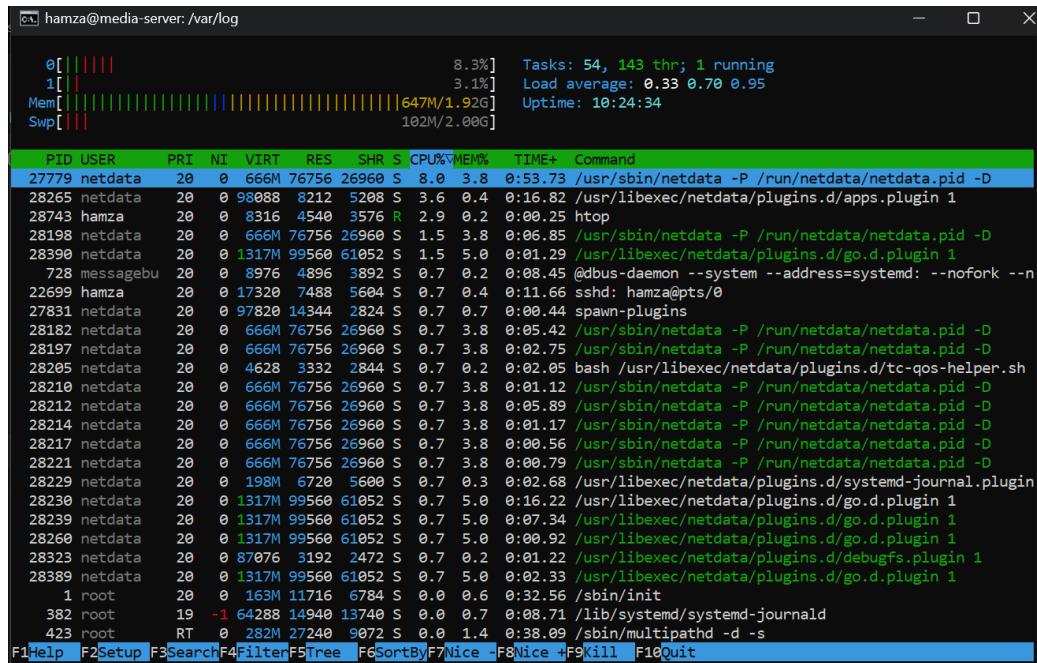


Figure 25: htop dashboard

These tools combined gave me **both system-level** and **application-level** insights into my Nextcloud server.

2.3.6 Real-Time Document Editing with OnlyOffice or Collabora

To enable real-time editing and collaboration of office documents (Word, Excel, PowerPoint) directly inside the Nextcloud interface using **OnlyOffice** or **Collabora Online**.

This feature simulates how modern companies work with online documents—collaborating instantly, without using Google Drive or Microsoft 365, while keeping all files securely stored in their private cloud.

What Is OnlyOffice / Collabora?

OnlyOffice and **Collabora** are powerful, open-source document editing suites that integrate directly with Nextcloud. They allow multiple users to:

-  Create and edit .docx, .xlsx, and .pptx files in the browser
-  Collaborate in **real-time** with other team members
-  Keep sensitive files stored **locally**, not on external cloud services

Why Is This Important?

In professional environments, companies need:

- Full **control** over document storage and access
- The ability for teams to **work together on documents** live
- A secure alternative to Google Docs or Office 365 that supports **enterprise compliance (like GDPR)**

Integrating OnlyOffice or Collabora into Nextcloud turns your cloud into a **complete private collaboration platform** — perfect for enterprise and education use.

What I Will Implement?

For this project, I will:

- Deploy **OnlyOffice Document Server** using Docker
- Integrate it with my existing Nextcloud instance
- Test editing .docx and .xlsx files in real time from multiple user accounts
- Capture screenshots and logs to demonstrate the setup and usage

Result

After completing this feature, users of my Nextcloud project will be able to:

- Open Word or Excel files from Nextcloud
- Edit them live in their browser
- Collaborate with other users instantly
- Do all this without sending the files to Google, Microsoft, or any external service

2.3.6.1 Install and Run ONLYOFFICE Document Server

1. Pull the ONLYOFFICE Docker image:

```
docker pull onlyoffice/documentserver
```

2. Create directory for SSL certificates:

```
sudo mkdir -p /etc/onlyoffice /certs
```

3. Generate self-signed SSL certificates:

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 \  
-keyout /etc/onlyoffice/certs/tls.key \  
-out /etc/onlyoffice/certs/tls.crt \  
-subj "/C=MA/ST=FES/L=FES/O=Student/CN=192.168.11.182"
```

4. Run ONLYOFFICE Document Server container with certificates:

```
docker run -d -p 8082:80 -p 8443:443 \  
--name onlyoffice-document-server \  
-e JWT_ENABLED=true \  
-e JWT_SECRET=secret \  
-e SSL_ENABLED=true \  
-v /etc/onlyoffice/certs:/var/www/onlyoffice/Data/certs:ro \  
onlyoffice/documentserver
```

Replace the CN value with your server IP or domain name.

Private Cloud Media Server Project

```
root@media-server:/etc/onlyoffice/certs# docker run -d -p 8082:80 -p 8443:443 \
--name onlyoffice-document-server \
-e JWT_ENABLED=true \
-e JWT_SECRET=secret \
-e SSL_ENABLED=true \
-v /etc/onlyoffice/certs:/var/www/onlyoffice/Data/certs:ro \
onlyoffice/documentserver
07485b6442bc018c9de2e1a7a4a0a7ee3a05932a7417aad564583ad9d0df8c37
```

Figure 26: docker run command and running container

2.3.6.2 Verify ONLYOFFICE Document Server Is Running

- ✓ Check exposed ports:

```
docker port onlyoffice-document-server
```

```
root@media-server:/etc/onlyoffice/certs# docker port onlyoffice-document-server
80/tcp -> 0.0.0.0:8082
80/tcp -> [::]:8082
443/tcp -> 0.0.0.0:8443
443/tcp -> [::]:8443
```

- ✓ Run health check inside the container:

```
docker exec -it onlyoffice-document-server curl -k https://localhost:8443/healthcheck
```

Expected output:

```
{"status":"ok"} OR nothing (This mean no errors)
```

```
root@media-server:/etc/onlyoffice/certs# docker exec -it onlyoffice-document-server curl -k https://192.168.11.182:8443/healthcheck
true
root@media-server:/etc/onlyoffice/certs#
```

2.3.6.3 Configure Nextcloud to Use ONLYOFFICE

- ↳ Install and enable ONLYOFFICE app in Nextcloud if not done
- ↳ Open Nextcloud Admin Settings → ONLYOFFICE
- ↳ Fill in the following settings:

Setting	Value
Document Editing Service address	https://192.168.11.182:8443
ONLYOFFICE Docs internal requests	https://192.168.11.182:8443
Server address for internal requests	http://192.168.11.182
JWT Secret	Secret

Replace IP with your actual server IP.

Private Cloud Media Server Project

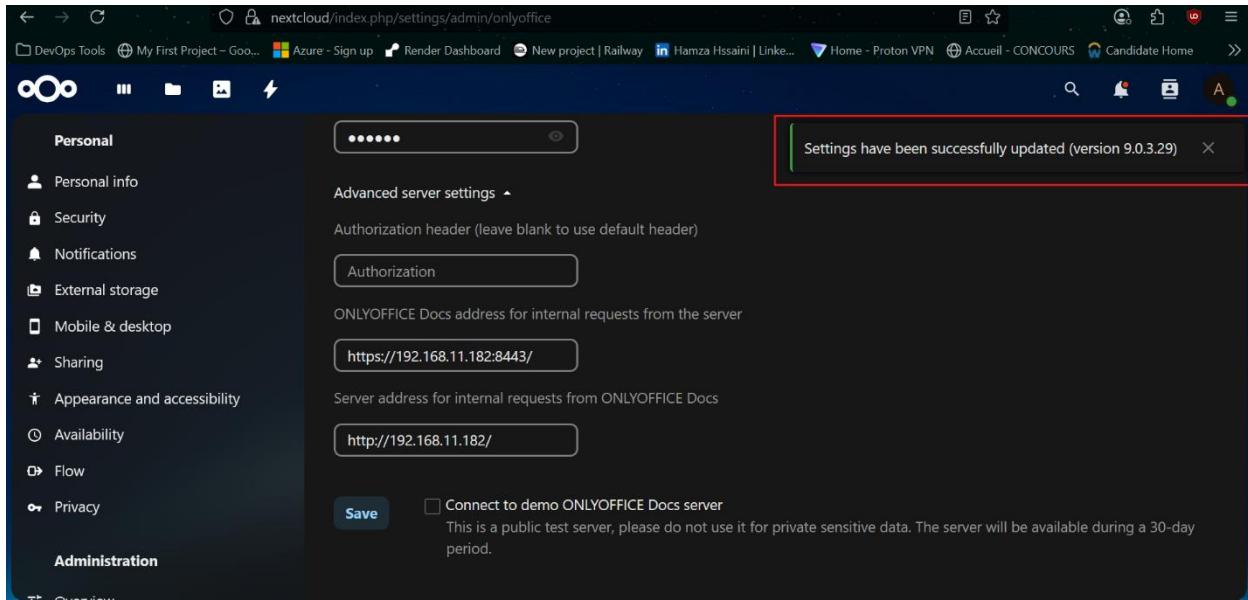


Figure 27: Nextcloud ONLYOFFICE app settings filled Success

- » Add trusted domain in Nextcloud CLI (run on Nextcloud server):

```
sudo -u www-data php /var/www/html/nextcloud/occ config:system:set  
trusted_domains 3 --value=192.168.11.182
```

```
root@media-server:/etc/onlyoffice/certs# sudo -u www-data php /var/www/html/nextcloud/occ config:system:set trusted_domains 3 --value=192.168.11.182  
system config value trusted_domains => 3 set to string 192.168.11.182
```

2.3.6.4 Test Document Editing

- » Upload a Word or Excel document to Nextcloud
- » Open the file — it should load in ONLYOFFICE editor in the browser
- » Test editing with multiple logged-in users simultaneously for live collaboration

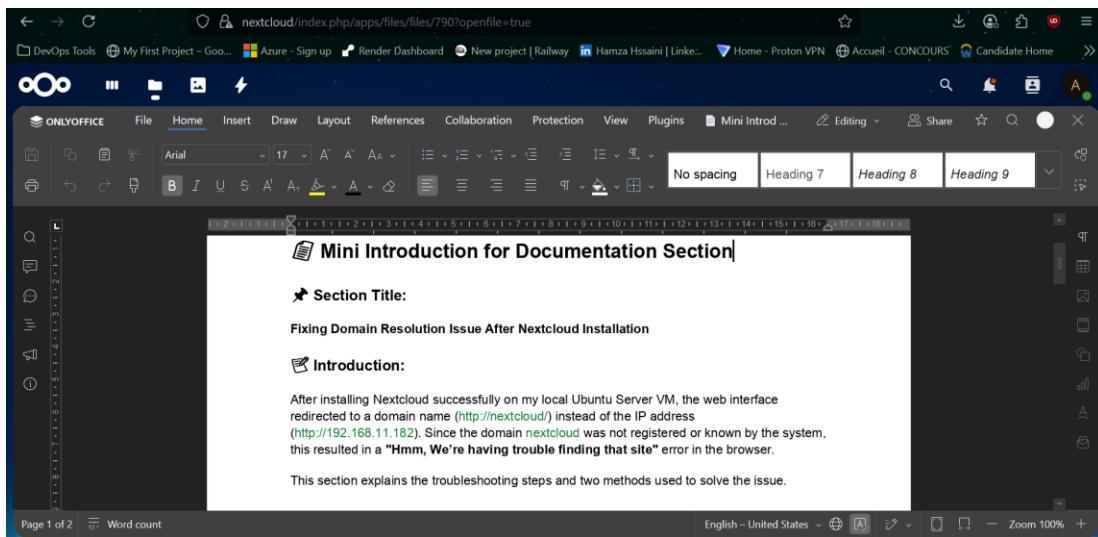


Figure 28: ONLYOFFICE document editor with active collaborative editing

Chapter 3: Troubleshooting Notes

Note 1: Solving Post-Installation Access Issues in Nextcloud

After successfully installing Nextcloud on my local Ubuntu Server virtual machine, I encountered several **common but critical access-related issues** that are often faced by system administrators in real-world deployments. These problems prevented proper access to the web interface and needed to be resolved for a stable, user-friendly environment.

This section documents the **three main problems** encountered, explains their root causes, and shows how each was resolved. This serves as a guide for other users who install Nextcloud in similar local environments and want to avoid or fix these issues efficiently.

1.1 Overview of the 3 Problems and Their Fixes

1.1.1 Incorrect Domain Redirection

☞ **Problem:**

After installation, the browser redirected to `http://nextcloud/` which failed to load because `nextcloud` wasn't a recognized hostname.

☞ **Fix:**

- Changed Apache's `ServerName` to the VM's IP in `/etc/apache2/sites-available/nextcloud.conf`
- Added a custom entry in the Windows hosts file to resolve `nextcloud` to the VM's IP

```
GNU nano 6.2                                     /etc/apache2/sites-available/nextcloud.conf
VirtualHost *:80>
  ServerAdmin admin@local
  DocumentRoot /var/www/html/nextcloud/
  ServerName 192.168.11.182
```

- Restarted Apache `systemctl status apache2`

1.1.2 Apache Showing Default Page Instead of Nextcloud

☞ **Problem:**

When visiting `http://nextcloud/`, the browser showed Apache's default Ubuntu welcome page instead of the Nextcloud dashboard.

☞ **Fix:**

- Disabled Apache's default site using `a2dissite 000-default.conf`

Private Cloud Media Server Project

- Ensured nextcloud.conf was enabled with `a2ensite nextcloud.conf`
- Reloaded Apache to apply the changes `systemctl status apache2`

1.1.3 Access Through Untrusted Domain

↳ Problem:

Nextcloud displayed the error: “Access through untrusted domain” when accessed via `http://nextcloud/`.

↳ Fix:

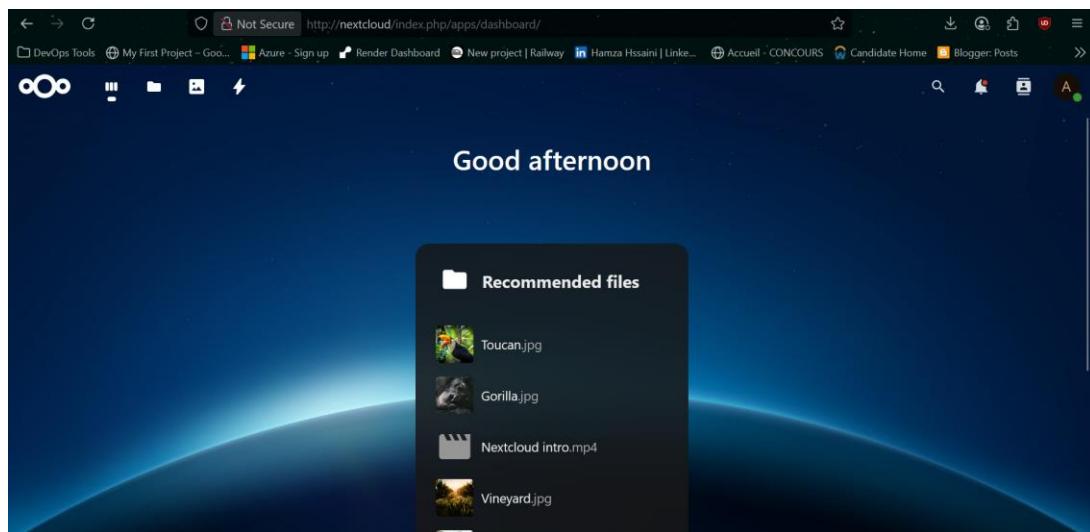
- Edited Nextcloud’s configuration file at `/var/www/html/nextcloud/config/config.php`
- Added `nextcloud` to the `trusted_domains` array

```
array (
    0 => '192.168.11.182',
    1 => 'nextcloud',
),
```

- Restarted Apache `systemctl status apache2`

↳ ✓ Result:

After applying these fixes, Nextcloud became fully accessible via both the IP address and the `nextcloud` hostname from my host machine. These steps ensured a smoother end-user experience and demonstrate key system administration skills such as virtual host management, domain resolution, and secure web service configuration.



Note2 : msmtpt: cannot log to ...: cannot open: Permission denied

↳ **Problem:**

msmtpt: cannot log to ...: cannot open: Permission denied

↳ **Solution:**

“During testing, I encountered permission issues with msmtpt log file access under different runtime environments (cron/root/user). Since email delivery was successful and verified, I disabled logging for simplicity in my student setup. In production, I would either configure AppArmor to allow secure logging or use systemd journal for email tracking.”

Note3 : ONLYOFFICE Document Server must be accessible via HTTPS

- **ONLYOFFICE Document Server must be accessible via HTTPS**, not HTTP, to connect to Nextcloud.

If not, you'll see the error:

Error when trying to connect (Mixed Active Content is not allowed.
HTTPS address for ONLYOFFICE Docs is required.)

-  If you encounter SSL errors, confirm that certificates are correctly mounted in the container and the CN matches your server IP.
-  Ports **8082 (HTTP)** and **8443 (HTTPS)** must be open and accessible from your Nextcloud server and clients.
-  To temporarily disable SSL verification (not recommended for production), run inside ONLYOFFICE container:

```
docker exec onlyoffice-document-server bash -c \
"echo -e '[ssl]\nverify-peer = false\nverify-host = false' >
/etc/onlyoffice/documentserver/default.json"
docker restart onlyoffice-document-server
```

Conclusion

This project simulates a full-featured private cloud media server environment using VirtualBox and open-source tools. It demonstrates cloud administration tasks including service deployment, HTTPS security, backup automation, monitoring, and user access control.

The project reflects real-world scenarios and challenges faced by system administrators and DevOps engineers. It strengthened skills in Ubuntu server administration, network configuration, security practices, and automation

Skills Demonstrated

Skill	Technology Used
Virtualization	VirtualBox
Server OS	Ubuntu 22.04
Web Hosting	Apache
DB Management	MariaDB
Cloud Storage	Nextcloud
Monitoring	Netdata, htop
Security	HTTPS, SMTP
Automation	Bash, Cron
Collaboration	OnlyOffice with Docker
Troubleshooting	Apache, DNS, SSL

[LinkedIn](#)