

Install NextCloud on Ubuntu 20.04 with Nginx (LEMP Stack)

edit from: <https://www.linuxbabe.com/ubuntu/install-nextcloud-ubuntu-20-04-nginx-lemp-stack>

This tutorial will be showing you how to install NextCloud on Ubuntu 20.04 LTS with Nginx web server.

What's NextCloud?

NextCloud is a free open-source self-hosted cloud storage solution. It's functionally similar to **Dropbox**. Proprietary cloud storage solutions (Dropbox, Google Drive, etc) are convenient, but at a price: they can be used to collect personal data because your files are stored on their computers. If you worried about privacy, you can switch to NextCloud, which you can install on your private home server or on a virtual private server (VPS). You can upload your files to your server via NextCloud and then sync those files to your desktop computer, laptop or smartphone. This way you have full control of your data.

NextCloud Features

- Free and open-source
- End-to-end encryption, meaning files can be encrypted on client devices before uploaded to the server, so even if someone steals your server, they can not read your files.
- Can be integrated with an online office suite (**Collobora Online**, OnlyOffice) so you can create and edit your doc, ppt, xls files directly from NextCloud.
- The app store contains hundreds of apps to extend functionality (like calendar app, contacts app, note-taking app, video conferencing app, etc).
- The sync client is available on Linux, macOS, Windows, iOS and android.

Prerequisites

NextCloud is written in PHP programming language. To follow this tutorial, you first need to install LEMP stack on Ubuntu 20.04. If you haven't already done so, please check out the following tutorial.

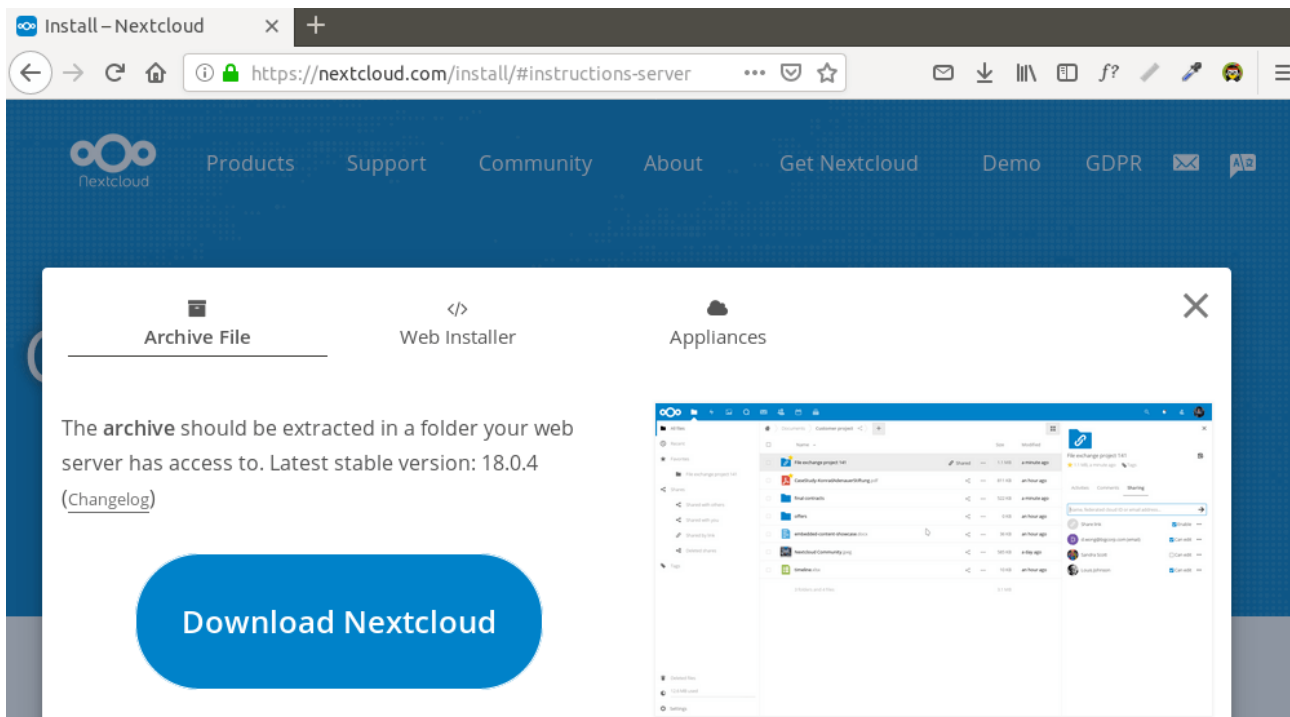
- [How to Install LEMP Stack \(Nginx, MariaDB, PHP7.4-FPM\) on Ubuntu 20.04](#)

You can install NextCloud on your home server or [a VPS \(virtual private server\)](#). You also need a domain name, so later on you will be able to enable HTTPS to encrypt the HTTP traffic. I registered my domain name from [NameCheap](#) because the price is low and they give whois privacy protection free for life. Nextcloud can be installed without a domain name, but it really doesn't make sense if you don't encrypt the HTTP connection to prevent snooping. I recommend buying a domain name, if you really want to tinker with server software and use them to the fullest potential.

Now let's install NextCloud.

Step 1: Download NextCloud on Ubuntu 20.04

Log into your Ubuntu 20.04 server. Then download the NextCloud zip archive onto your server. The latest stable version is 18.0.4 at time of this writing. You may need to change the version number. Go to <https://nextcloud.com/install> and click the [download for server](#) button to see the latest version.



You can run the following command to download it on your server.

```
wget https://download.nextcloud.com/server/releases/nextcloud/nextcloud-21.0.1.zip
```

You can always use the above URL format to download NextCloud. If a new version comes out, simply replace 18.0.4 with the new version number.

Once downloaded, extract the archive with unzip.

```
apt install unzip
```

```
unzip nextcloud-21.0.1.zip -d /usr/share/nginx/
```

The -d option specifies the target directory. NextCloud web files will be extracted to /usr/share/nginx/nextcloud/. Then we need to change the owner of this directory to www-data so that the web server (Nginx) can write to this directory.

```
chown www-data:www-data /usr/share/nginx/nextcloud/ -R
```

Step 2: Create a Database and User for Nextcloud in MariaDB Database Server

Log into MariaDB database server with the following command. Since MariaDB is now using `unix_socket` plugin to authentication user login, there's no need to enter MariaDB root password. We just need to prefix the `mysql` command with `sudo`.

```
mysql
```

Then create a database for Nextcloud. This tutorial name the database `nextcloud`. You can use whatever name you like.

```
create database nextcloud;
```

Create the database user. Again, you can use your preferred name for this user. Replace `your-password` with your preferred password.

```
create user nextclouduser@localhost identified by 'your-  
password';
```

Grant this user all privileges on the `nextcloud` database.

```
grant all privileges on nextcloud.* to nextclouduser@localhost  
identified by 'your-password';
```

Flush privileges and exit.

```
flush privileges;
```

```
exit;
```

```
MariaDB [(none)]> create database nextcloud;
Query OK, 1 row affected (0.002 sec)

MariaDB [(none)]> create user nextclouduser@localhost identified by 'your-password';
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> grant all privileges on nextcloud.* to nextclouduser@localhost identified by 'your-password';
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> exit;
Bye
```

Step 3: Create a Nginx Config File for Nextcloud

Create a `nextcloud.conf` file in `/etc/nginx/conf.d/` directory, with a command-line text editor like Nano.

```
nano /etc/nginx/conf.d/nextcloud.conf
```

Copy and paste the following text into the file. Replace `nextcloud.example.com` with your own preferred sub-domain. Don't forget to create DNS A record for this sub-domain in your DNS zone editor. If you don't have a real domain name, I recommend going to [NameCheap](#) to buy one. The price is low and they give whois privacy protection free for life.

```
server {
    listen 80;
    listen [::]:80;
    server_name nextcloud.example.com;

    # Add headers to serve security related headers
    add_header X-Content-Type-Options nosniff;
    add_header X-XSS-Protection "1; mode=block";
    add_header X-Robots-Tag none;
    add_header X-Download-Options noopen;
    add_header X-Permitted-Cross-Domain-Policies none;
    add_header Referrer-Policy no-referrer;

    #I found this header is needed on Ubuntu, but not on Arch
Linux.
    add_header X-Frame-Options "SAMEORIGIN";

    # Path to the root of your installation
    root /usr/share/nginx/nextcloud/;
```

```
access_log /var/log/nginx/nextcloud.access;
error_log /var/log/nginx/nextcloud.error;
```

```
location = /robots.txt {
    allow all;
    log_not_found off;
    access_log off;
}
```

```
# The following 2 rules are only needed for the user_webfinger
app.
```

```
# Uncomment it if you're planning to use this app.
#rewrite ^/.well-known/host-meta /public.php?service=host-meta
last;
#rewrite ^/.well-known/host-meta.json /public.php?
service=host-meta-json
# last;
```

```
location = /.well-known/carddav {
    return 301 $scheme://$host/remote.php/dav;
}
location = /.well-known/caldav {
    return 301 $scheme://$host/remote.php/dav;
}
```

```
location ~ /.well-known/acme-challenge {
    allow all;
}
```

```
# set max upload size
client_max_body_size 512M;
fastcgi_buffers 64 4K;
```

```
# Disable gzip to avoid the removal of the ETag header
gzip off;
```

```
# Uncomment if your server is build with the ngx_pagespeed
module
```

```
# This module is currently not supported.
#pagespeed off;
```

```
error_page 403 /core/templates/403.php;
error_page 404 /core/templates/404.php;
```

```
location / {
    rewrite ^ /index.php;
}
```

```
location ~ ^/(?::build|tests|config|lib|3rdparty|templates|
data)/ {
    deny all;
```

```

    }
    location ~ ^/(?:\.|autotest|occ|issue|indie|db_|console) {
        deny all;
    }

    location ~ ^/(?:index|remote|public|cron|core/ajax/update|
status|ocs/v[12]|updater/.+|ocs-provider/.+|core/templates/40[34])
\.php(?:$|/) {
        include fastcgi_params;
        fastcgi_split_path_info ^(.+\.(php|php5))(/.*)$;
        try_files $fastcgi_script_name =404;
        fastcgi_param SCRIPT_FILENAME
$document_root$fastcgi_script_name;
        fastcgi_param PATH_INFO $fastcgi_path_info;
        #Avoid sending the security headers twice
        fastcgi_param modHeadersAvailable true;
        fastcgi_param front_controller_active true;
        fastcgi_pass unix:/run/php/php7.4-fpm.sock;
        fastcgi_intercept_errors on;
        fastcgi_request_buffering off;
    }

    location ~ ^/(?:updater|ocs-provider)(?:$|/) {
        try_files $uri/ =404;
        index index.php;
    }

    # Adding the cache control header for js and css files
    # Make sure it is BELOW the PHP block
    location ~* \.(?:css|js)$ {
        try_files $uri /index.php$uri$is_args$args;
        add_header Cache-Control "public, max-age=7200";
        # Add headers to serve security related headers (It is
intended to
        # have those duplicated to the ones above)
        add_header X-Content-Type-Options nosniff;
        add_header X-XSS-Protection "1; mode=block";
        add_header X-Robots-Tag none;
        add_header X-Download-Options noopen;
        add_header X-Permitted-Cross-Domain-Policies none;
        add_header Referrer-Policy no-referrer;
        # Optional: Don't log access to assets
        access_log off;
    }

    location ~* \.(?:svg|gif|png|html|ttf|woff|ico|jpg|jpeg)$ {
        try_files $uri /index.php$uri$is_args$args;
        # Optional: Don't log access to other assets
        access_log off;
    }
}

```

Save and close the file. (To save a file in Nano text editor, press `Ctrl+O`, then press `Enter` to confirm. To exit, press `Ctrl+X`.)

Then test Nginx configuration.

```
sudo nginx -t
```

If the test is successful, reload Nginx for the changes to take effect.

```
systemctl reload nginx
```

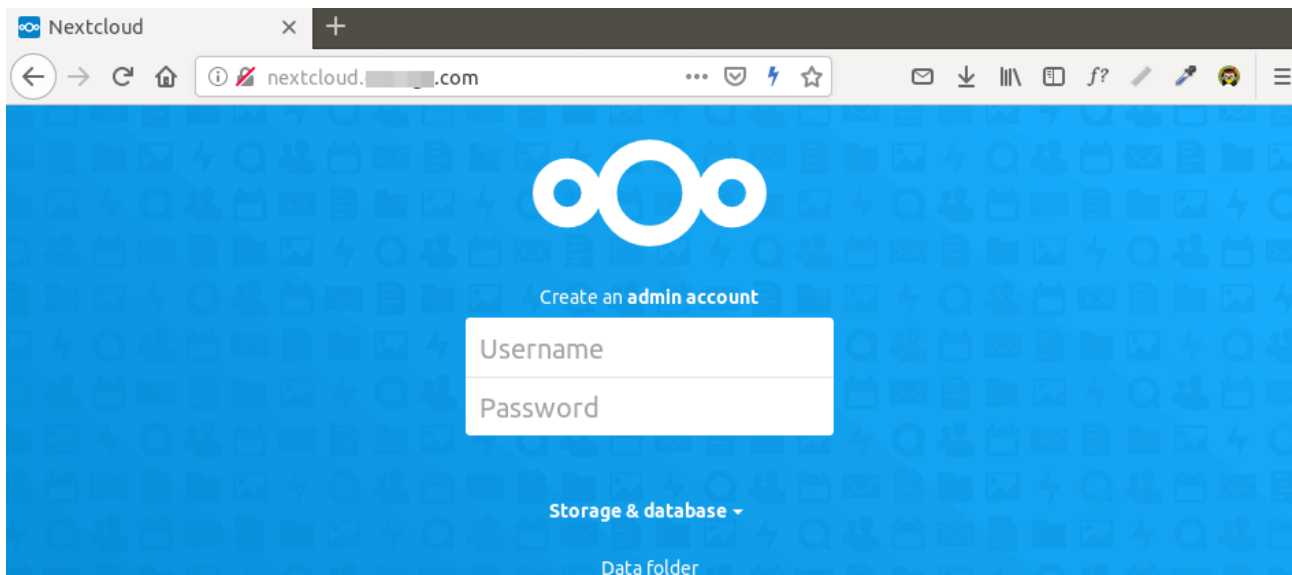
Step 4: Install and Enable PHP Modules

Run the following commands to install PHP modules required or recommended by NextCloud.

```
apt install imagemagick php-imagick php7.4-common php7.4-mysql  
php7.4-fpm php7.4-gd php7.4-json php7.4-curl php7.4-zip php7.4-xml  
php7.4-mbstring php7.4-bz2 php7.4-intl php7.4-bcmath php7.4-gmp
```

Step 5: Enable HTTPS

Now you can access the Nextcloud web install wizard in your web browser by entering the domain name for your Nextcloud installation.
`nextcloud.example.com`



If the web page can't load, you probably need to open port 80 in firewall.

```
iptables -I INPUT -p tcp --dport 80 -j ACCEPT
```

And port 443 as well.

```
iptables -I INPUT -p tcp --dport 443 -j ACCEPT
```

Before entering any sensitive information, we should enable secure HTTPS connection on Nextcloud. We can obtain a free TLS certificate from Let's Encrypt. Install Let's Encrypt client (certbot) from Ubuntu 20.04 repository.

```
apt install certbot python3-certbot-nginx
```

Python3-certbot-nginx is the Nginx plugin. Next, run the following command to obtain a free TLS certificate using the Nginx plugin.

```
certbot --nginx --agree-tos --redirect --hsts --staple-ocsp --email  
you@example.com -d nextcloud.example.com
```

Where:

- `--nginx`: Use the Nginx authenticator and installer

- `–agree-tos`: Agree to Let's Encrypt terms of service
- `–redirect`: Enforce HTTPS by adding 301 redirect.
- `–hsts`: Enable HTTP Strict Transport Security. This defends against SSL/TLS stripping attack.
- `–staple-ocsp`: Enable OCSP Stapling.
- `–email`: Email used for registration and recovery contact.
- `-d` flag is followed by a list of domain names, separated by comma. You can add up to 100 domain names.

You will be asked if you want to receive emails from EFF(Electronic Frontier Foundation). After choosing Y or N, your TLS certificate will be automatically obtained and configured for you, which is indicated by the message below.

IMPORTANT NOTES:

- Congratulations! Your certificate and chain have been saved at:
`/etc/letsencrypt/live/nextcloud.linuxbabe.com/fullchain.pem`
 Your key file has been saved at:
`/etc/letsencrypt/live/nextcloud.linuxbabe.com/privkey.pem`
 Your cert will expire on 2020-07-28. To obtain a new or tweaked version of this certificate in the future, simply run certbot again with the "certonly" option. To non-interactively renew **all** of your certificates, run "certbot renew"
- If you like Certbot, please consider supporting our work by:

Donating to ISRG / Let's Encrypt: <https://letsencrypt.org/donate>
 Donating to EFF: <https://eff.org/donate-le>

I found that Certbot may not be able to add HSTS header in the Nginx config file for Nextcloud. If you would like to enable HSTS (HTTP Strict Transport Security), then edit the file.

```
nano /etc/nginx/conf.d/nextcloud.conf
```

We can then add the following line in the SSL server block to enable HSTS header. (If it's already there, then your configuration are fine.)

```
add_header Strict-Transport-Security "max-age=31536000"
always;
```

Also, you can enable HTTP2 protocol by adding the option `http2`, which will speed up webpage loading.

```
listen 443 ssl http2; # managed by Certbot
```

Like below.

```
listen 443 ssl http2; # managed by Certbot
ssl_certificate /etc/letsencrypt/live/nextcloud.linuxbabe.com/fullchain.pem; # managed by Certbot
ssl_certificate_key /etc/letsencrypt/live/nextcloud.linuxbabe.com/privkey.pem; # managed by Certbot
include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

add_header Strict-Transport-Security "max-age=31536000" always;

ssl_trusted_certificate /etc/letsencrypt/live/nextcloud.linuxbabe.com/chain.pem; # managed by Certbot
ssl_stapling on; # managed by Certbot
ssl_stapling_verify on; # managed by Certbot
}
server {
    if ($host = nextcloud.linuxbabe.com) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    listen 80;
    server_name nextcloud.linuxbabe.com;
    return 404; # managed by Certbot
}
```

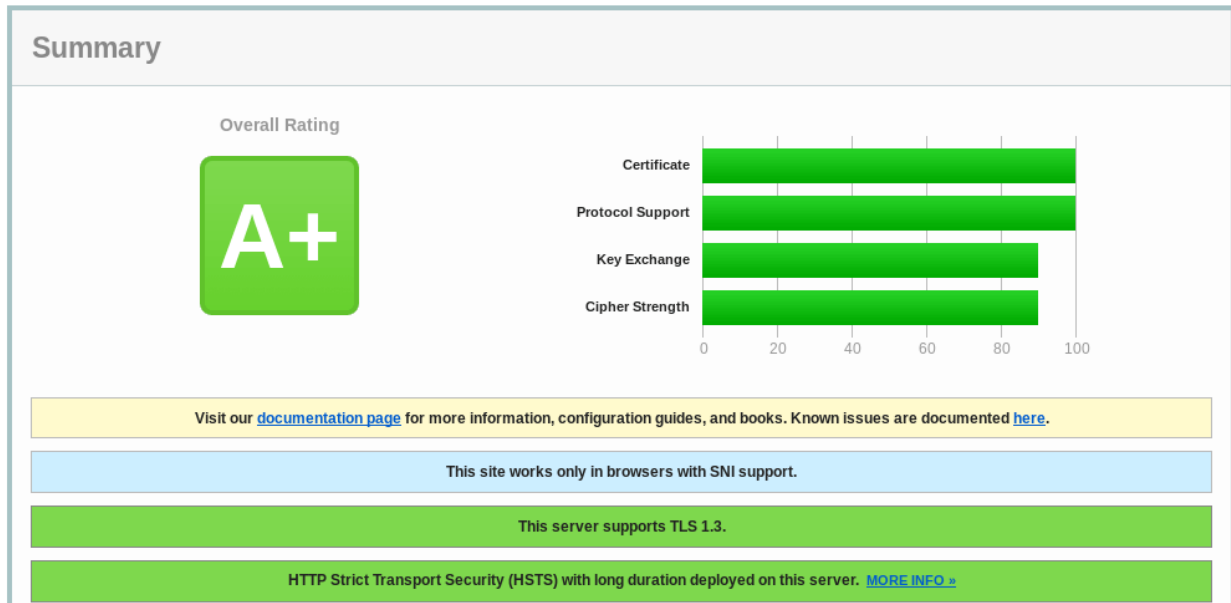
Save and close the file. Then test Nginx configurations.

```
nginx -t
```

If the test is successful, reload Nginx for the change to take effect.

```
systemctl reload nginx
```

The above configuration will get A+ score on [SSL test](#).



Step 6: Finish the Installation in your Web Browser

Now you can access the Nextcloud web install wizard using HTTPS connection.

<https://nextcloud.example.com>

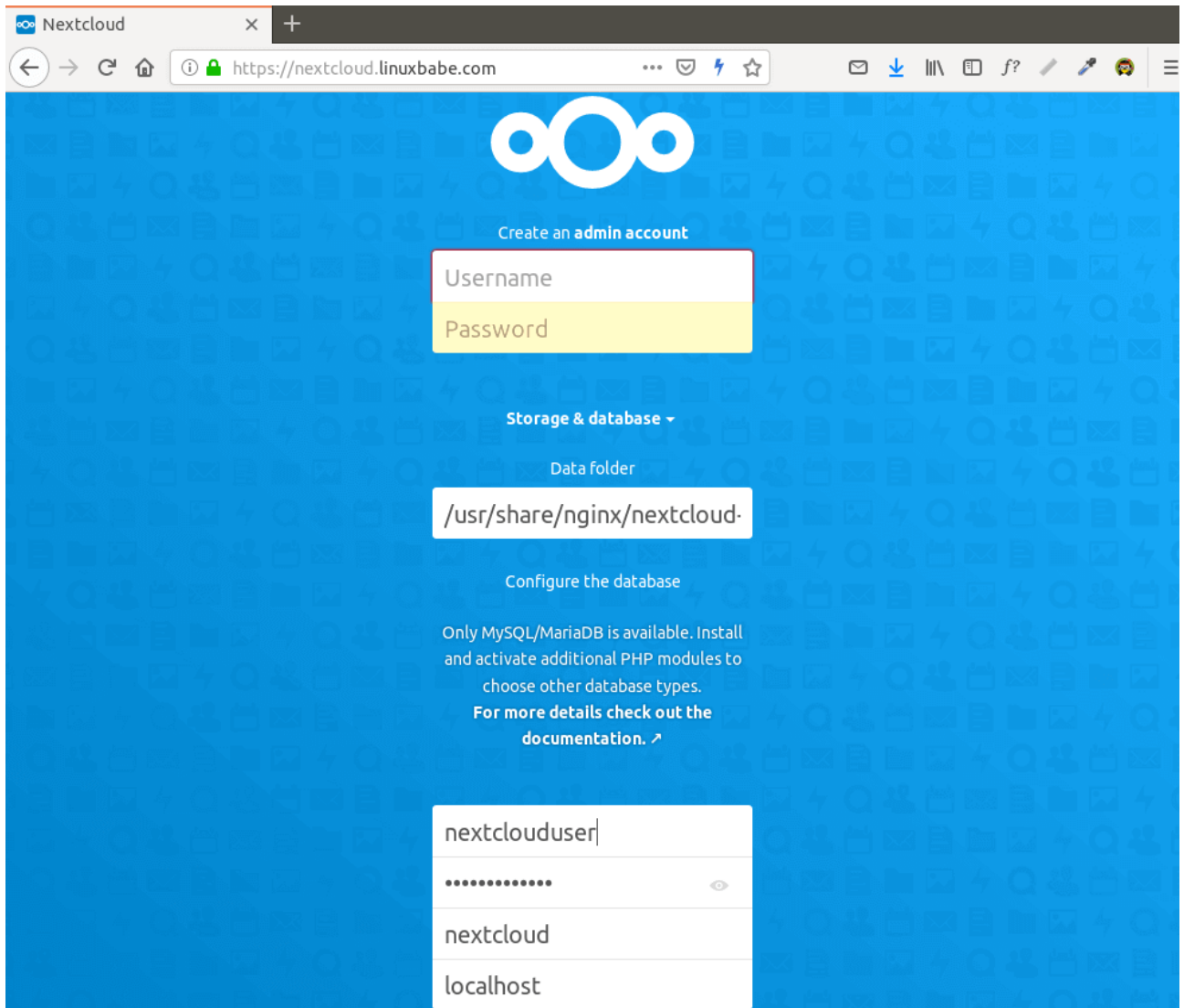
To complete the installation, you need to create an admin account, enter the path of Nextcloud data folder, enter database details you created in step 2. You can use the default `localhost` as host address, or you can enter `localhost:3306`, as MariaDB listens on port 3306.

The data folder is where users' files are stored. For security, it's best to place the data directory outside of Nextcloud webroot directory. So instead of storing users' files under `/usr/share/nginx/nextcloud/data/`, we can change it to `/usr/share/nginx/nextcloud-data`. which can be created with the following command:

```
mkdir /usr/share/nginx/nextcloud-data
```

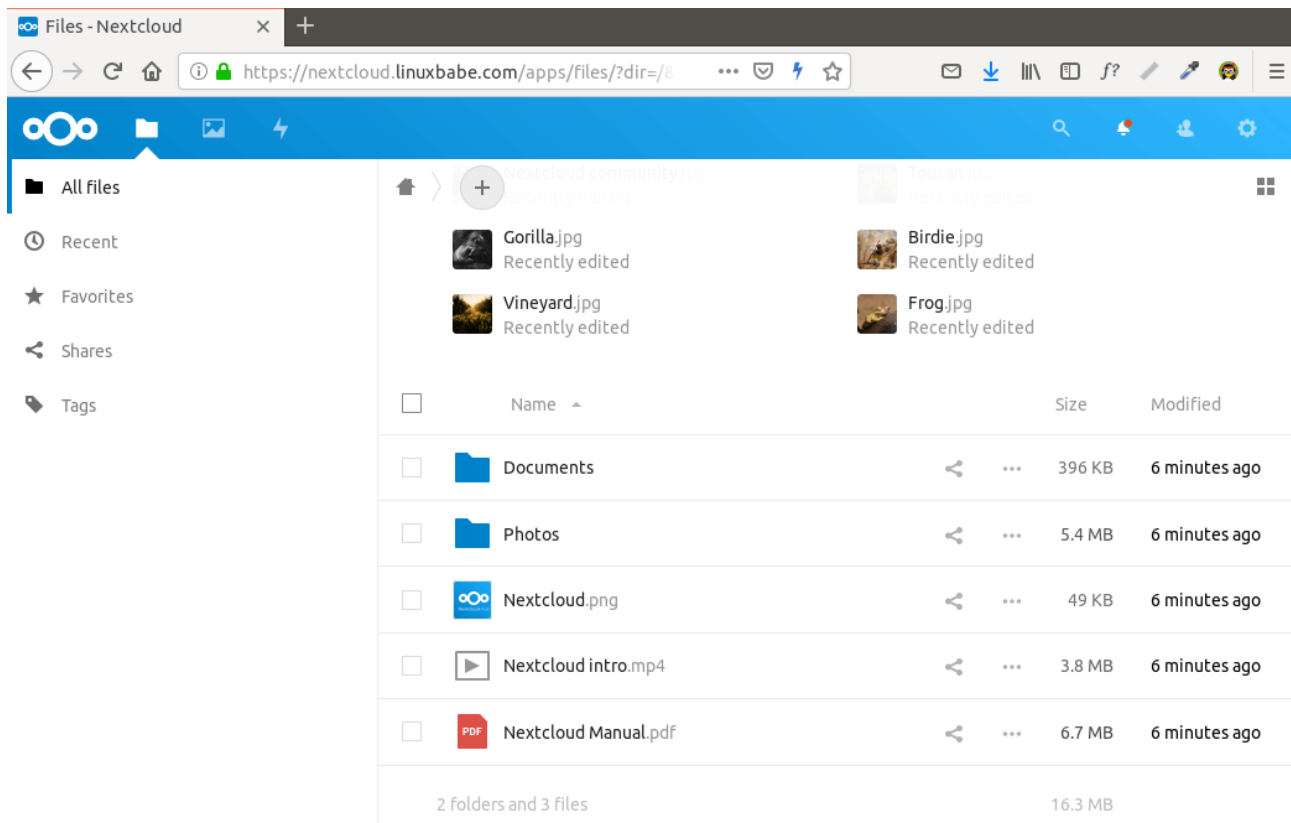
Then make sure Nginx user (`www-data`) has write permission to the data directory.

```
chown www-data:www-data /usr/share/nginx/nextcloud-data -R
```



The screenshot shows the Nextcloud installation web interface in a browser. The browser's address bar displays `https://nextcloud.linuxbabe.com`. The page has a blue background with a repeating pattern of Nextcloud logos. At the top center is the Nextcloud logo. Below it, the text "Create an admin account" is displayed. There are two input fields: "Username" and "Password". Below these fields is a section titled "Storage & database" with a dropdown arrow. Under this section, the "Data folder" is set to `/usr/share/nginx/nextcloud`. Below that, the text "Configure the database" is shown, followed by a note: "Only MySQL/MariaDB is available. Install and activate additional PHP modules to choose other database types. For more details check out the documentation." At the bottom, there is a form with four fields: "nextclouduser", a password field (represented by dots), "nextcloud", and "localhost".

Click the **Finish Setup** button, you will see the Web interface of Nextcloud. Congrats! You can start using it as your private cloud storage.



any more see...

<https://www.linuxbabe.com/ubuntu/install-nextcloud-ubuntu-20-04-nginx-lemp-stack>