PSAwise2324Team10Aufgabe06

https://131.159.74.56:61055/nextcloud/

We picked NextCloud as our web service.

Installation

The web service will use the PostgreSQL Database provided by team 4. In order to access the database from our machine, we had to install the PostgreSQL client libraries:

```
sudo apt update
sudo apt install postgresql-client
```

Install PHP, as it is one of NextCloud's requirements:

```
sudo apt install php
```

We will host NextCloud on our Apache Server, in combination with php-fpm since that is the official recommendation.

We also need to install several other required packages:

```
sudo apt install postgresql postgresql-contrib php-pgsql libapache2-mod-php
php php-gmp php8.1-bcmath php-gd php-json php-curl php-mbstring php-intl php-
imagick php-xml php-zip php-fpm php-redis php-apcu php-opcache php-ldap bzip2
zip unzip imagemagick ffmpeg redis-server
```

We also need to disable php and enable php-fpm as they are clashing.

```
a2enconf php8.1-fpm
a2dismod php8.1
a2dismod mpm_prefork
a2enmod mpm_event

systemctl start apache2
systemctl enable apache2
systemctl start php8.1-fpm
systemctl enable php8.1-fpm

a2enmod ssl rewrite headers proxy proxy_http deflate cache proxy_wstunnel
http2 proxy_fcgi env expires
```

Nextcloud Installation

Enter the directory `/var/www/` and run the following commands to download Nextcloud:

```
# download nextcloud
wget https://download.nextcloud.com/server/releases/nextcloud-27.1.4.zip

sudo unzip nextcloud-27.1.4.zip
```

We also need to change the permissions so that the www-data user owns the files related to Nextcloud.

```
sudo chown -R www-data:www-data nextcloud
```

Apache and Nginx Configuration

We use Nginx as a reverse proxy and Apache as the service that hosts our nextcloud website.

Add the DNS record to the Nginx config file:

```
server {
       ...

    server_name vmpsateam10-05.psa-team10.cit.tum.de

       # proxy to the port apache server uses
        location ~ ^/nextcloud {
                allow all;

                proxy_pass http://127.0.0.1:8081;

                proxy_set_header Host nextcloud.psa-team10.cit.tum.de;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto $scheme;

        }

}
```

Also add the new virtual host to the Apache config file:

```
<VirtualHost 127.0.0.1:8081>
        ServerName nextcloud.psa-team10.cit.tum.de

        DocumentRoot /var/www/nextcloud

        <Directory /var/www/nextcloud></Directory>
                Require all granted
                Satisfy Any
                AllowOverride All
                Options FollowSymLinks MultiViews
```

```
                <IfModule mod_dav.c>
                Dav off
                </IfModule>
        </Directory>

RewriteEngine On
RewriteRule ^/nextcloud/(.*)$ /var/www/nextcloud/$1 [L]

        ErrorLog /var/log/apache2/error.log
</VirtualHost>
```

Firewall Configuration

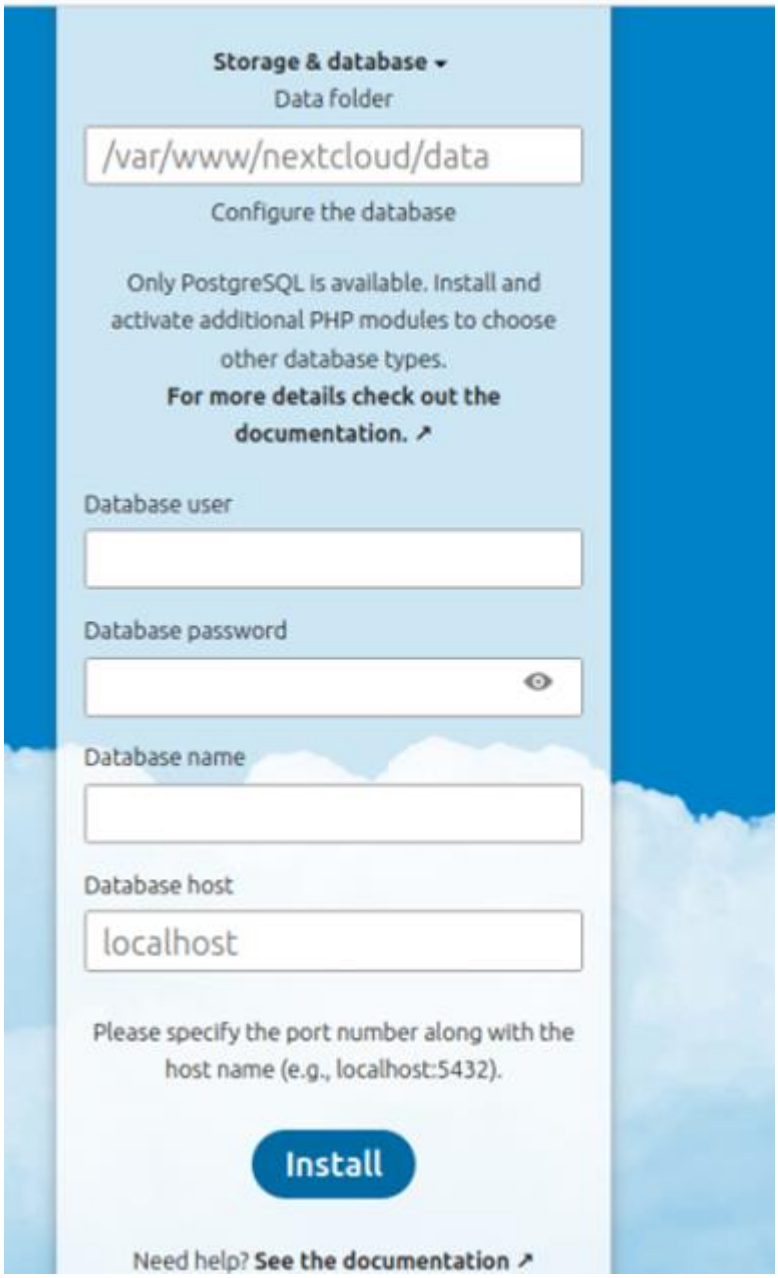
For our webservice to be useable, we need to make a few changes to the firewall. Database access through port 5432 has to be permitted. Access to the webservice using the public IP address of psa.in.tum.de needs to be allowed as well.

Add the following rules:

```
chain input {
<...>
    # DB
    tcp sport 5432 accept

    # public IP used for nextcloud
    ip saddr 131.159.74.56 accept
<...>
}
chain output {
<...>
    # DB
    tcp dport 5432 accept
<...>
}
```

Nextcloud Configuration

Access the Nextcloud service through a browser and follow the installation prompts.



Edit the file at `/var/www/nextcloud/config/config.php` to look as follows:

```php
<?php
$CONFIG = array (
  <...>
  'trusted_domains' =>
  array (
    0 => '131.159.74.56:61055',
    1 => 'nextcloud.psa-team10.cit.tum.de',
  ),
  'trusted_proxies' =>
  array (
   ),
  'datadirectory' => '/var/www/nextcloud/data',
  'dbtype' => 'pgsql',
  'version' => '27.1.4.1',
  'overwritehost' => '131.159.74.56:61055',
  'overwrite.cli.url' => '131.159.74.56:61055',
  'overwriteprotocol' => 'https',
  'dbname' => 'team10db',
  'dbhost' => '192.168.4.1',
  'dbport' => '5432',
  'dbtableprefix' => 'oc_',
  'dbuser' => 'team10',
   <...>
);
```

User Management

We created a .csv file that contains all usernames of the PSA participants as well as their email address and a randomly generated password.

Only root has access to that file.

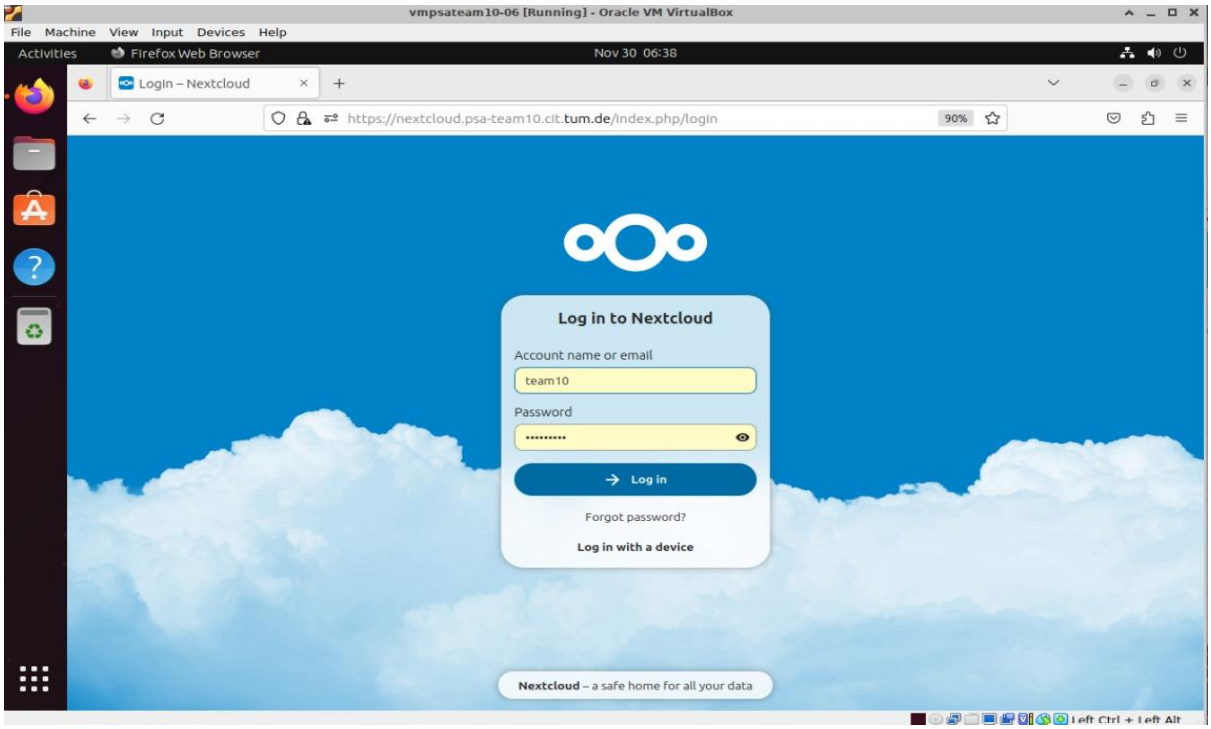
To create and enable all users, we wrote a bash script add_users.sh in `/var/www/nextcloud` that automates this process:

```bash
#!/bin/bash

# Path to Nextcloud's occ command
OCC="/var/www/nextcloud/occ"

# Path to the CSV file containing user information
CSV_FILE="/root/users.csv"

# Check if the CSV file exists
if [ ! -f "$CSV_FILE" ]; then
    echo "CSV file not found: $CSV_FILE"
    exit 1
fi


# Iterate through each line in the CSV file
while IFS=',' read -r username email password || [ -n "$username" ]; do
    # Skip the header line in the CSV file
    if [[ $username = "username" ]]; then
        continue
```

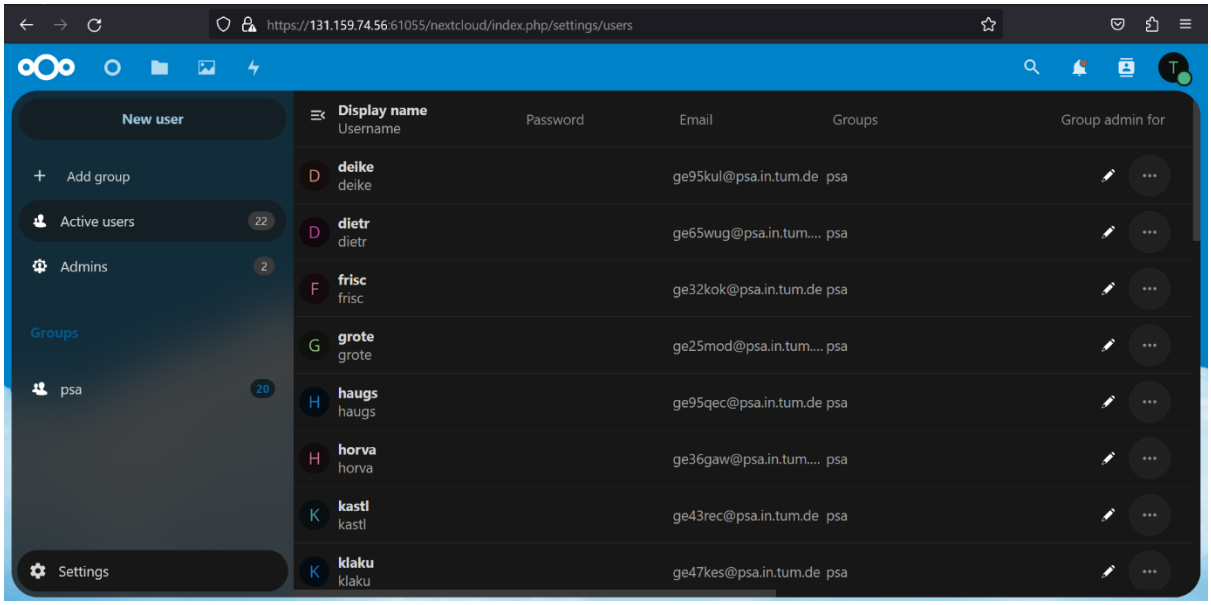
```
    fi


    # Add user using Nextcloud's occ command
    echo -e "$password\n$password" | sudo -u www-data php "$OCC" user:add --
display-name="$username" --group="psa" $username

    # Add email
    sudo -u www-data php "$OCC" user:setting $username settings email "$email"

    # Enable user
    sudo -u www-data php "$OCC" user:enable $username
    if [ $? -eq 0 ]; then
        echo "User $username added successfully."
    else
        echo "Failed to add user: $username"
    fi

done < "$CSV_FILE"
```

OCC is a command-line tool that allows us to manage Nextcloud. Here is a link to the official wiki.

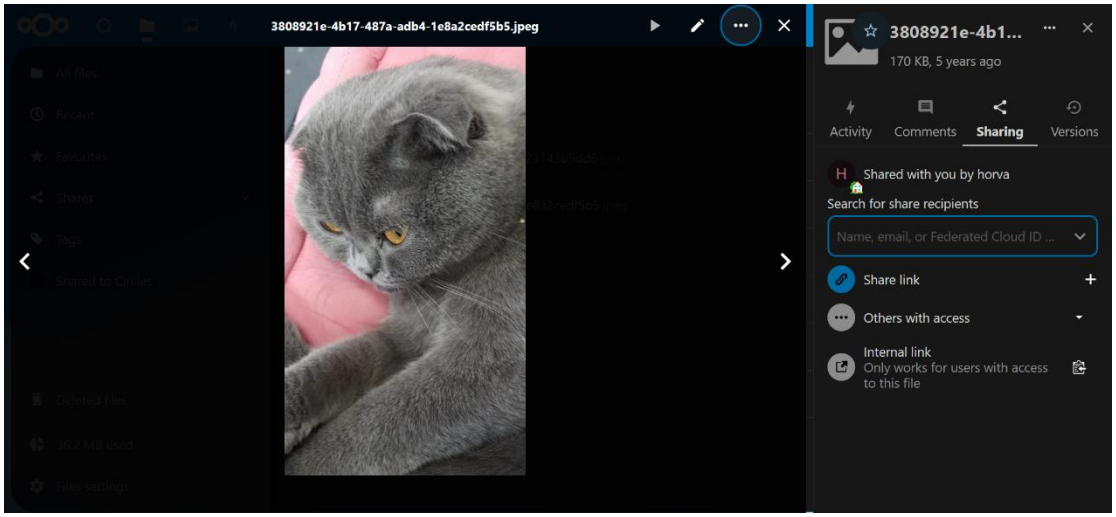
Web Interface

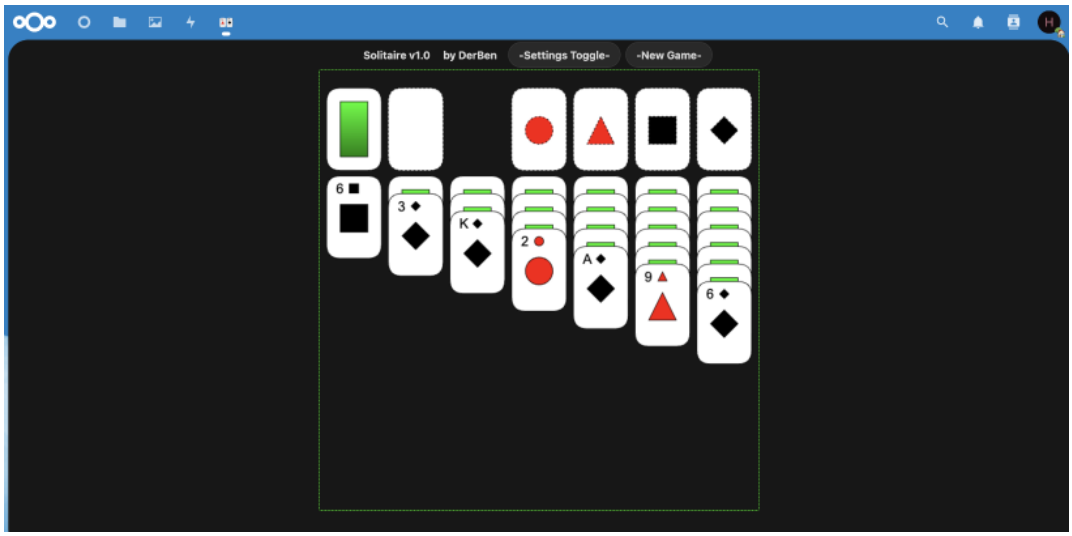


If you login as a user with admin rights, you can manage the users from the web interface as well. You can delete users, add new ones or assign admin rights or new groups.



You can also make new folders and update files, which can be shared with other users as well. You can upload files with a size up to 5MB.





It is also possible to add more apps to Nextcloud, such as a GitHub integration or even games like Solitaire.

Testing

The tests for this week can be found on VM5.

```bash
#!/bin/bash

failed_tests=0

fail() {
    ((failed_tests++))
    echo "FAIL $@"
}

ok() {
    echo "OK $@"
}

request() {
    response=$(curl -s -o /dev/null -w "%{http_code}" --noproxy "*" -L -k
"$1")
    if [ "$response" -eq "$2" ]; then
        ok "$1"
    else
        fail "$1 | status: $response"
    fi
}

count () {
    echo "$1" | grep -o "$2" | wc -l
}

# arguments are 1:uid,2:group name
check_group() {
    entry=$(sudo -u www-data php /var/www/nextcloud/occ user:info --
output=json $1)
    is_group=$(echo "$entry" | jq -r --arg group "$2" '.groups[] | select(. ==
$group)')
    if [ "$is_group" == "$2" ]; then
        ok "${1} is in ${2}"
    else
        fail "${1} is not in ${2}."
    fi
}

if systemctl is-active --quiet "nginx"; then
        ok "Nginx is running"
else
        fail "Nginx is inactive"
fi

if systemctl is-active --quiet "apache2"; then
        ok "Apache is running"
else
        fail "Apache is inactive"
fi
```

```
if ping -c 1 -W 4 192.168.4.1 >/dev/null 2>&1; then
    ok "DB reachable"
else
    fail "Can't reach DB"
fi


urls=("https://131.159.74.56:61055/nextcloud/")

for url in "${urls[@]}"; do
    request $url "200"
done

# check if contents make sense
response=$(curl --noproxy "*" -L -k "https://131.159.74.56:61055/nextcloud/" -
-silent)
if [[ "$response" == *"This application requires JavaScript for correct
operation."* ]]; then
    ok "Response makes sense"
else
    fail "Unexpected response; expected phrase missing"
fi


user_list=$(sudo -u www-data php /var/www/nextcloud/occ user:list --info --
output=json_pretty)


number_of_users=$(count "$user_list" "user_id")

expected=22
if [ "$number_of_users" -eq "$expected" ]; then
    ok "There are ${number_of_users} users, psa + root + team10"
else
    fail "There are ${number_of_users} users, should be ${expected}"
fi


number_of_admins=$(count "$user_list"  "admin")
if [ "$number_of_admins" -eq "4" ]; then
    ok "There are ${number_of_admins} admin users, root + team10"
else
    fail "There are ${number_of_admins} admin users, should be 4"
fi


check_group "root" "admin"
#check_group "root" "psa"
check_group "horva" "admin"
check_group "kastl" "admin"
check_group "team10" "admin"

echo "Failed tests: $failed_tests"
```