Version - **2024.01**

Dernière mise-à-jour : 2024/04/01 11:16

DOF310 - Harbor

Contenu du Module

- Harbor
  - LAB #5 - Installation et Configuration de Harbor
    - 5.1 - Installation de Docker et de Docker Compose
    - 5.2 - Installation de Harbor
    - 5.3 - Création d'un Nouveau Projet
    - 5.4 - Création d'un Dépôt
    - 5.5 - Pousser des Images Supplémentaires
    - 5.6 - Travailler avec des Artefacts
    - 5.7 - Copier un Artefact d'un Projet vers un autre Projet
    - 5.8 - Utilisation de Labels
    - 5.9 - Authentification et RBAC
    - 5.10 - Utilisation d'Images Docker Personnalisées
    - 5.11 - Rétention de Tags
    - 5.12 - Immutabilité des Tags

Ressources

Lab #5

- wget https://www.dropbox.com/scl/fi/tww1xm3wtpgnv974k6vbr/myEntrypoint.sh?rlkey=qypry4ryhoax8ddtqzy5td16d&dl=0
- wget https://www.dropbox.com/scl/fi/bu8z1a5dxjs22gfy9fugb/Dockerfile?rlkey=nnl68r193nc6lor6429n03322&dl=0

Harbor

Harbor est un registre open source qui :

- sécurise les artefacts (images) avec des politiques
- utilise un contrôle d'accès basé sur les rôles
- garantit que les images sont analysées et exemptes de vulnérabilités
- signe les images comme fiables.

LAB #5 - Installation et Configuration de Harbor

5.1 - Installation de Docker et de Docker Compose

Connectez-vous à votre instance de Debian11 en tant que trainee au 10.0.2.46 puis devenez l'utilisateur **root**.

Installez les dépendances de Docker :

```
root@debian11:~# apt-get install apt-transport-https ca-certificates curl gnupg2 software-properties-common
```

Téléchargez la clef GPG officielle de docker :

```
root@debian11:~# curl -fsSL https://download.docker.com/linux/debian/gpg | apt-key add -
OK
```

Ajoutez le dépôt stable de docker :

```
root@debian11:~# add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/debian $(lsb_release -cs)
stable"
```

Installez maintenant **docker-ce** :

```
root@debian11:~# apt-get install docker-ce lvm2 libdevmapper-dev
```

Téléchargez docker-compose :

```
root@debian11:~#  wget https://github.com/docker/compose/releases/download/v2.14.0/docker-compose-linux-x86_64
```

Rendez docker-compose exécutable et déplacez-le dans le répertoire **/usr/bin:** :

```
root@debian11:~# chmod +x docker-compose-linux-x86_64

root@debian11:~# mv docker-compose-linux-x86_64 /usr/bin/docker-compose
```

5.2 - Installation de Harbor

Téléchargez Harbor :

```
root@debian11:~# wget
https://github.com/goharbor/harbor/releases/download/v2.6.2/harbor-offline-installer-v2.6.2.tgz
```

Désarchivez Harbor :

```
root@debian11:~# tar xvzf harbor-offline-installer*.tgz
harbor/harbor.v2.6.2.tar.gz
harbor/prepare
harbor/LICENSE
harbor/install.sh
harbor/common.sh
harbor/harbor.yml.tmpl
```

Placez-vous dans le répertoire **harbor** et copiez le fichier **harbor.yml.tmpl** vers **harbor.yml** :

```
root@debian11:~# cd harbor/
```

```
root@debian11:~/harbor# cp harbor.yml.tmpl harbor.yml
```

Créez le fichier **/etc/docker/daemon.json** afin de configurer Docker pour l'utilisation du dépôt non-sécurisé :

```
root@debian11:~/harbor# vi /etc/docker/daemon.json
root@debian11:~/harbor# cat /etc/docker/daemon.json
{
"insecure-registries" : ["10.0.2.46:5000", "0.0.0.0"]
}
```

Modifiez maintenant l'entrée **hostname** du fichier **harbor/harbor.yml** et commentez la section concernant les connexions https :

```
root@debian11:~/harbor# vi harbor.yml
root@debian11:~/harbor# head -n 20 harbor.yml
# Configuration file of Harbor

# The IP address or hostname to access admin UI and registry service.
# DO NOT use localhost or 127.0.0.1, because Harbor needs to be accessed by external clients.
hostname: 10.0.2.46

# http related config
http:
  # port for http, default is 80. If https enabled, this port will redirect to https port
  port: 80

# https related config
# https:
  # https port for harbor, default is 443
  # port: 443
  # The path of cert and key files for nginx
  # certificate: /your/certificate/path
  # private_key: /your/private/key/path
```

```
# # Uncomment following will enable tls communication between all harbor components
```

Constatez la présence du script **install.sh** dans le répertoire **harbor/** :

```
root@debian11:~/harbor# ls
common.sh             harbor.yml       install.sh  prepare
harbor.v2.6.2.tar.gz  harbor.yml.tmpl  LICENSE
```

Désactivez le service nginx dans la VM Debian11 :

```
root@debian11:~/harbor# systemctl stop nginx

root@debian11:~/harbor# systemctl disable nginx
```

Installez maintenant Harbor avec le support pour **Chart Museum** et **Trivy** :

```
root@debian11:~/harbor# ./install.sh --with-trivy --with-chartmuseum

[Step 0]: checking if docker is installed ...

Note: docker version: 20.10.21

[Step 1]: checking docker-compose is installed ...

Note: docker-compose version: 2.14.0

[Step 2]: loading Harbor images ...
...
[Step 3]: preparing environment ...
...
[Step 4]: preparing harbor configs ...
...
[Step 5]: starting Harbor ...
➜
```

```
    Chartmusuem will be deprecated as of Harbor v2.6.0 and start to be removed in v2.8.0 or later.
    Please see discussion here for more details. https://github.com/goharbor/harbor/discussions/15057
[+] Running 13/13
 ⠿ Network harbor_harbor              Created
0.2s
 ⠿ Network harbor_harbor-chartmuseum  Created
0.3s
 ⠿ Container harbor-log               Started
14.1s
 ⠿ Container registry                 Started
15.8s
 ⠿ Container registryctl              Started
18.1s
 ⠿ Container harbor-portal            Started
18.1s
 ⠿ Container redis                    Started
16.9s
 ⠿ Container chartmuseum              Started
15.7s
 ⠿ Container harbor-db                Started
16.4s
 ⠿ Container trivy-adapter            Started
14.8s
 ⠿ Container harbor-core              Started
15.0s
 ⠿ Container nginx                    Started
13.4s
 ⠿ Container harbor-jobservice        Started
13.2s
✔ ----Harbor has been installed and started successfully.----
```

Constatez les conteneurs créés par l'installation :

```
root@debian11:~/harbor# docker ps -a
```

```
CONTAINER ID   IMAGE                                  COMMAND                  CREATED          STATUS
PORTS                                NAMES
b8e4e3934c7a   goharbor/harbor-jobservice:v2.6.2      "/harbor/entrypoint.…"   15 minutes ago   Up 15 minutes
(healthy)                                           harbor-jobservice
579b4b43877f   goharbor/nginx-photon:v2.6.2           "nginx -g 'daemon of…"   15 minutes ago   Up 15 minutes
(healthy)   0.0.0.0:80->8080/tcp, :::80->8080/tcp   nginx
57c8697b4d39   goharbor/trivy-adapter-photon:v2.6.2   "/home/scanner/entry…"   16 minutes ago   Up 15 minutes
(healthy)                                           trivy-adapter
ce4823c62886   goharbor/harbor-core:v2.6.2            "/harbor/entrypoint.…"   16 minutes ago   Up 15 minutes
(healthy)                                           harbor-core
57992f8f573a   goharbor/chartmuseum-photon:v2.6.2     "./docker-entrypoint…"   16 minutes ago   Up 15 minutes
(healthy)                                           chartmuseum
e2df863f7009   goharbor/harbor-portal:v2.6.2          "nginx -g 'daemon of…"   16 minutes ago   Up 15 minutes
(healthy)                                           harbor-portal
3e7e2093a4a2   goharbor/harbor-db:v2.6.2              "/docker-entrypoint.…"   16 minutes ago   Up 15 minutes
(healthy)                                           harbor-db
406aa384c742   goharbor/redis-photon:v2.6.2           "redis-server /etc/r…"   16 minutes ago   Up 15 minutes
(healthy)                                           redis
ddf08dcc8390   goharbor/harbor-registryctl:v2.6.2     "/home/harbor/start.…"   16 minutes ago   Up 15 minutes
(healthy)                                           registryctl
b5cedbab739d   goharbor/registry-photon:v2.6.2        "/home/harbor/entryp…"   16 minutes ago   Up 15 minutes
(healthy)                                           registry
13665ace83fd   goharbor/harbor-log:v2.6.2             "/bin/sh -c /usr/loc…"   16 minutes ago   Up 15 minutes
(healthy)   127.0.0.1:1514->10514/tcp                harbor-log
```

Ainsi que les images téléchargées :

```
root@debian11:~/harbor# docker images
REPOSITORY                       TAG       IMAGE ID       CREATED       SIZE
goharbor/harbor-exporter         v2.6.2    ae230e335853   4 weeks ago   95.4MB
goharbor/chartmuseum-photon      v2.6.2    be361eee8a36   4 weeks ago   225MB
goharbor/redis-photon            v2.6.2    84fa3669bb10   4 weeks ago   154MB
goharbor/trivy-adapter-photon    v2.6.2    beef2527c7b2   4 weeks ago   441MB
goharbor/notary-server-photon    v2.6.2    b43e7c963f06   4 weeks ago   112MB
```

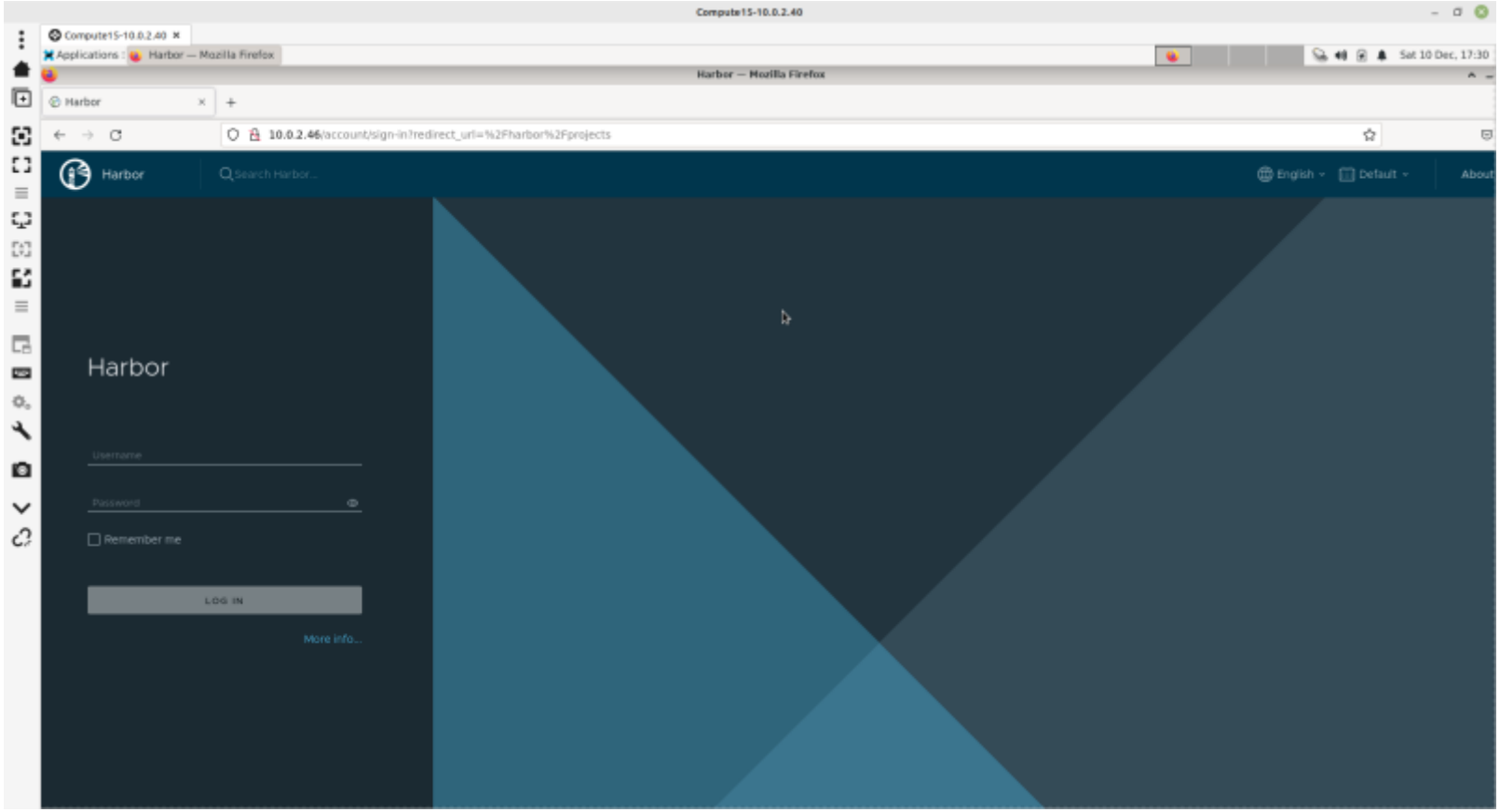
```
goharbor/notary-signer-photon   v2.6.2    4e9982286316   4 weeks ago   109MB
goharbor/harbor-registryctl     v2.6.2    d403d4e0f990   4 weeks ago   137MB
goharbor/registry-photon        v2.6.2    596e445b18f0   4 weeks ago   77.5MB
goharbor/nginx-photon           v2.6.2    cbbdf28947a5   4 weeks ago   154MB
goharbor/harbor-log             v2.6.2    3ce902023f73   4 weeks ago   161MB
goharbor/harbor-jobservice      v2.6.2    05791a75c8b8   4 weeks ago   250MB
goharbor/harbor-core            v2.6.2    484e2a64616d   4 weeks ago   213MB
goharbor/harbor-portal          v2.6.2    932297030d4e   4 weeks ago   162MB
goharbor/harbor-db              v2.6.2    351b19e366cb   4 weeks ago   225MB
goharbor/prepare                v2.6.2    eecc993c7643   4 weeks ago   164MB
```

Constatez maintenant les réseaux créés par l'installation :

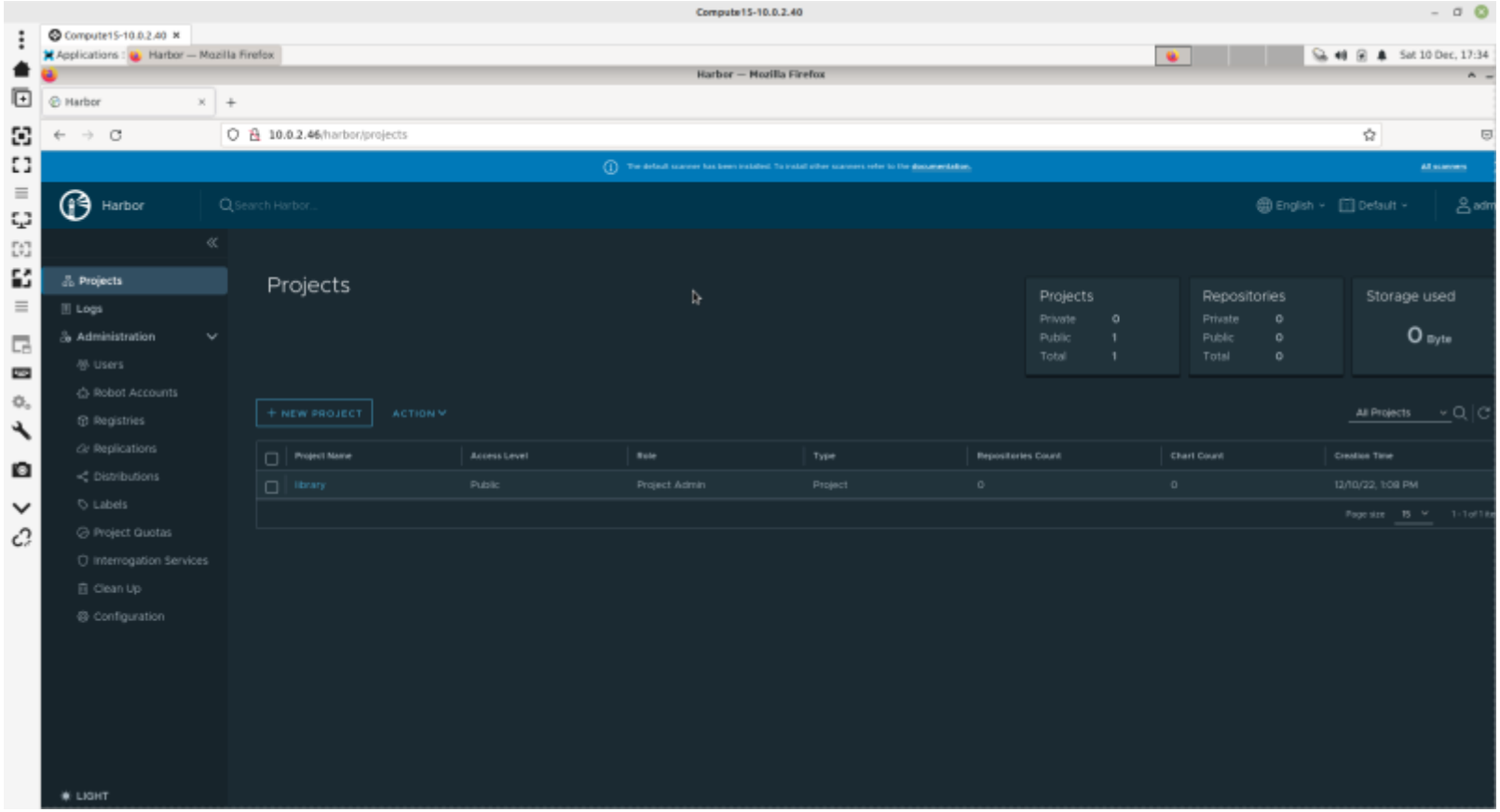
```
root@debian11:~/harbor# docker network ls
NETWORK ID     NAME                          DRIVER    SCOPE
c080bf952007   bridge                        bridge    local
380a2401fb01   harbor_harbor                 bridge    local
b5d0cc4623f6   harbor_harbor-chartmuseum      bridge    local
3e48278ae275   host                          host      local
4dd9174d8e25   none                          null      local
```

5.3 - Création d'un Nouveau Projet

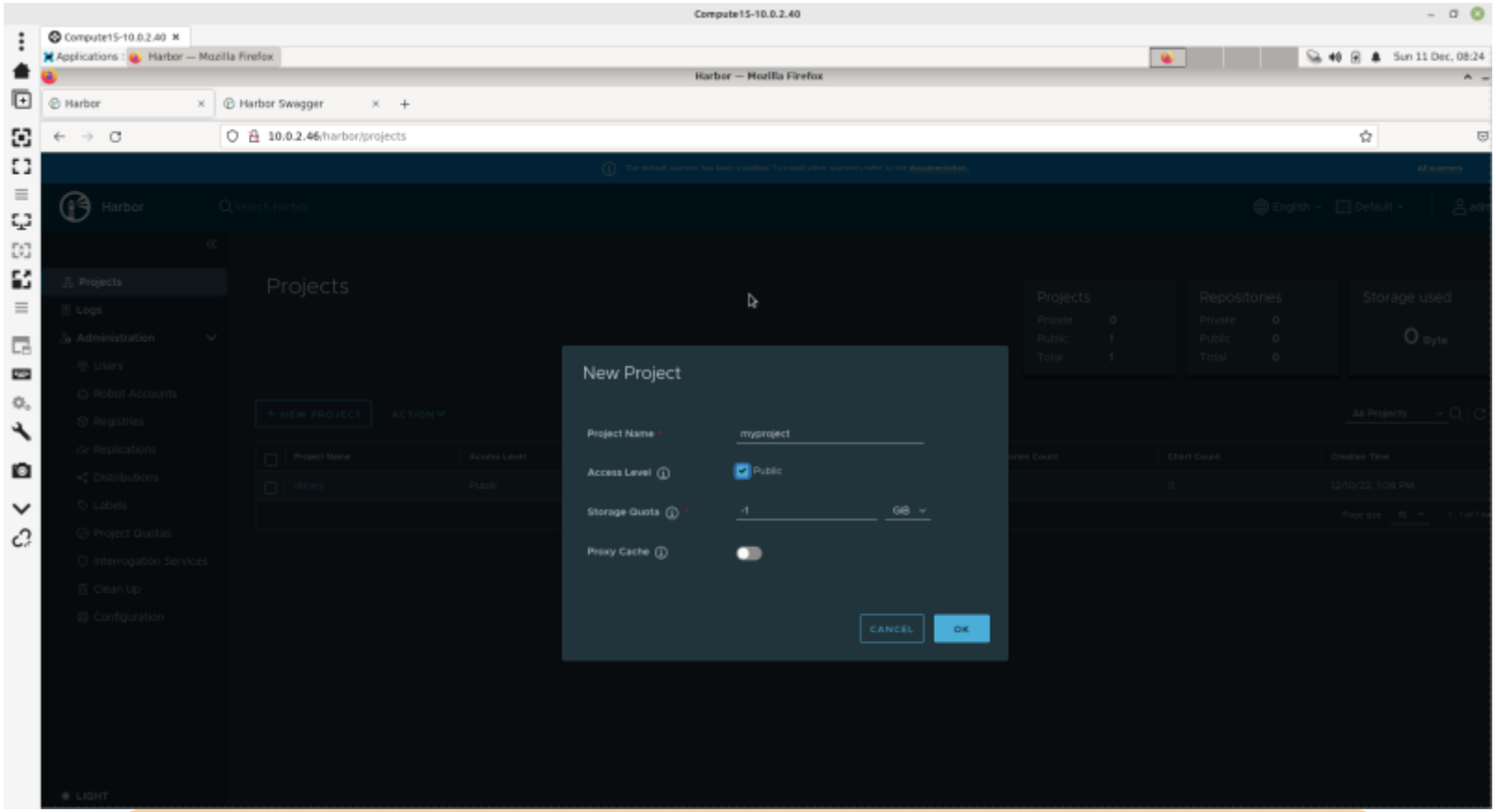
Connectez-vous à votre VM **Gateway_10.0.2.40_VNC** et ouvrez le navigateur web. Allez à l'URL http://10.0.2.46 :



Connectez-vous avec les identifiants par défaut - admin / Harbor12345 :

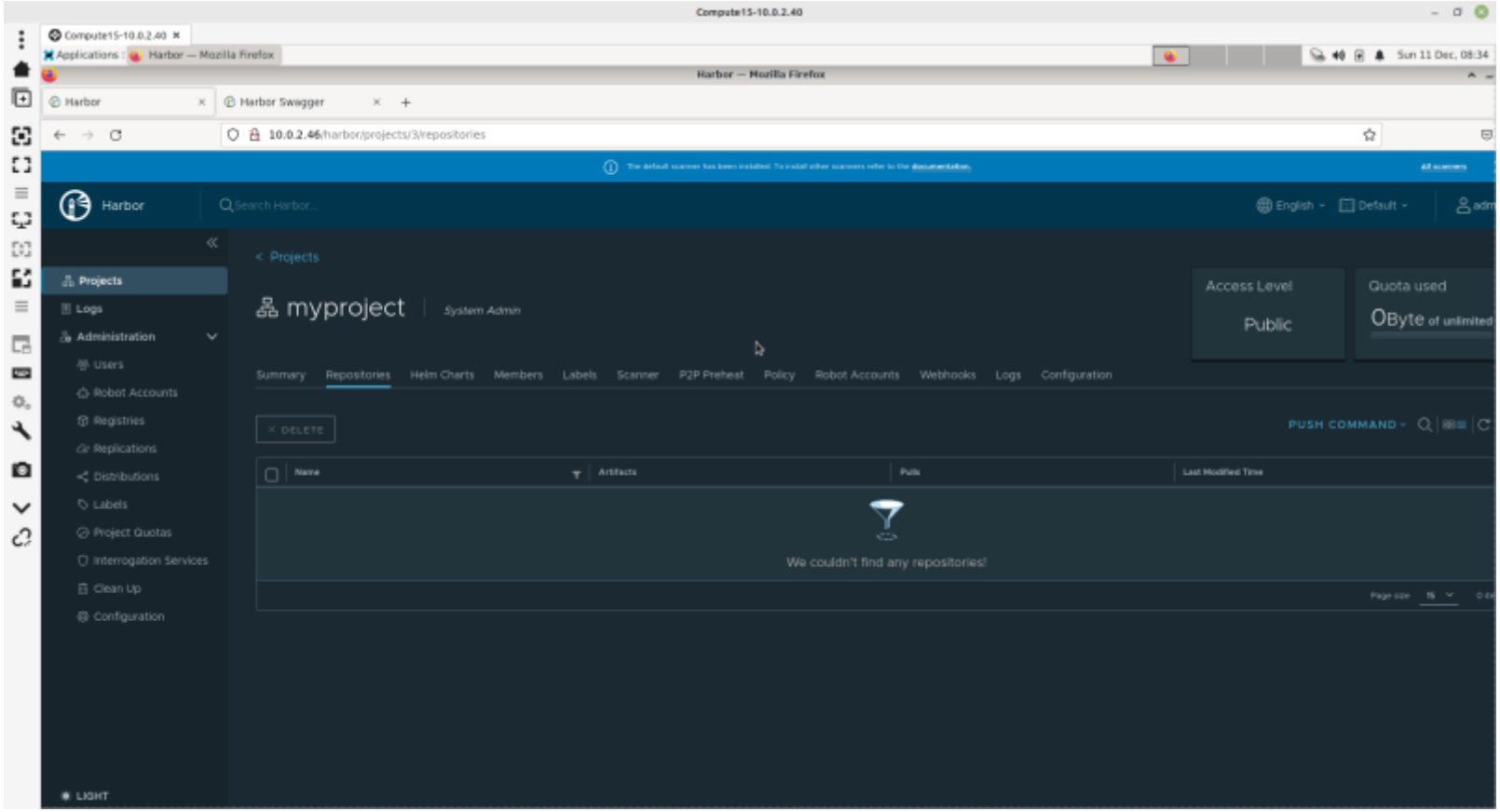


Harbor est organisé autour de **Projets**. Dans un projet se trouvent des **Dépôts** ou *Repositories*. Le projet nous permet de configurer le RBAC. Créez donc un nouveau projet en cliquant sur le bouton **New Project** :



Important : Notez que la valeur de **-1** indique que le projet a un quota de disque **illimité**.

Cliquez sur le bouton **OK** puis cliquez sur le nom **myproject** :



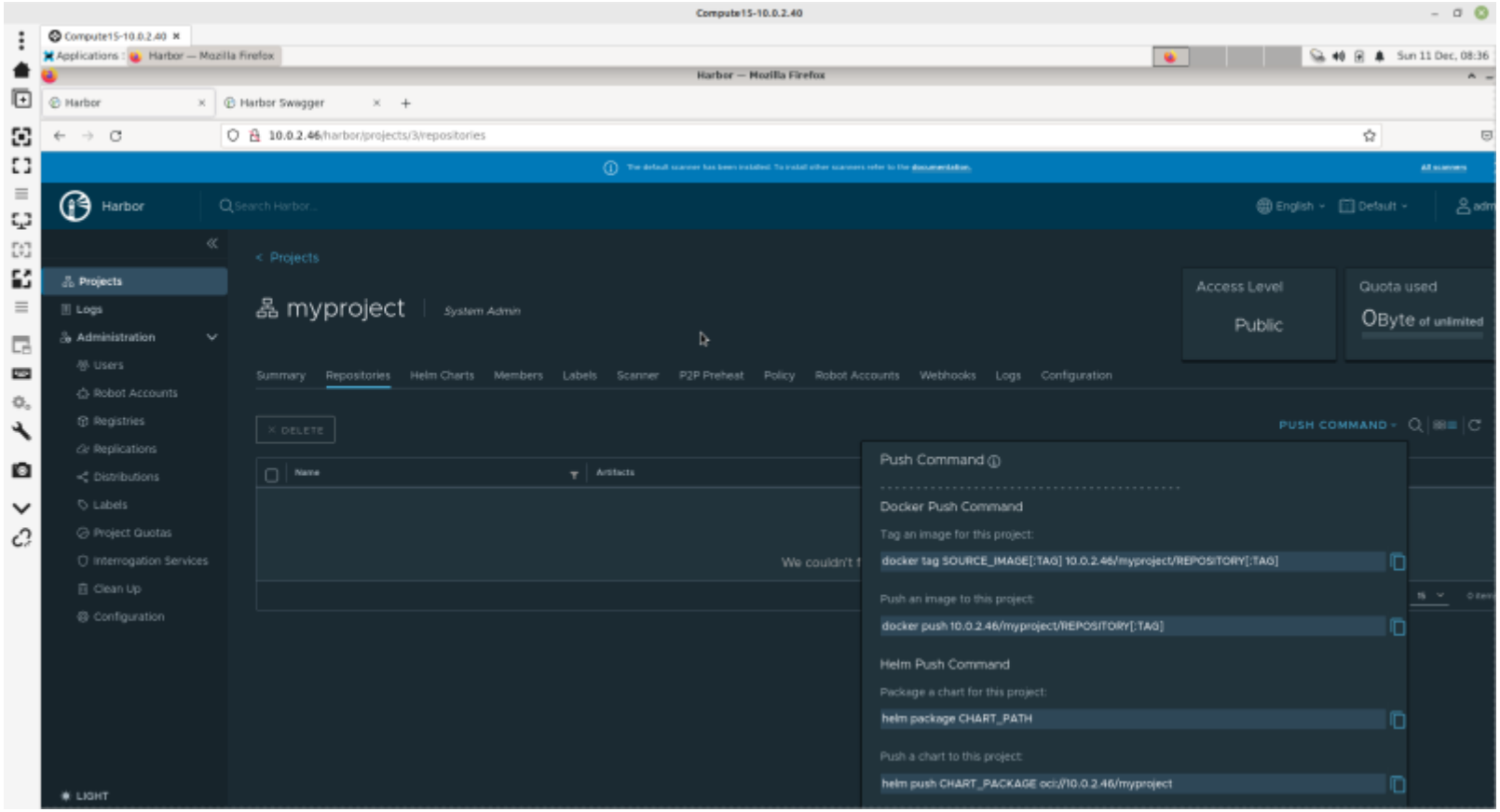
5.4 - Création d'un Dépôt

Téléchargez maintenant l'image de **hello-world** à partir de docker.io :

```
root@debian11:~/harbor# docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:faa03e786c97f07ef34423fccceeec2398ec8a5759259f94d99078f264e9d7af
Status: Downloaded newer image for hello-world:latest
```

```
docker.io/library/hello-world:latest
```

Retournez dans Harbor et cliquez sur **PUSH COMMAND** à droite. Dans la liste des commandes vous vous apercevrez la commande pour *tagger* une image pour le porjet **myproject** :



Copiez cette commande puis modifiez la ainsi :

```
root@debian11:~/harbor# docker tag hello-world 10.0.2.46:80/myproject/hello-world

root@debian11:~/harbor# docker images
```

```
REPOSITORY                          TAG       IMAGE ID       CREATED          SIZE
goharbor/harbor-exporter            v2.6.2    ae230e335853   4 weeks ago      95.4MB
goharbor/chartmuseum-photon         v2.6.2    be361eee8a36   4 weeks ago      225MB
goharbor/redis-photon               v2.6.2    84fa3669bb10   4 weeks ago      154MB
goharbor/trivy-adapter-photon       v2.6.2    beef2527c7b2   4 weeks ago      441MB
goharbor/notary-server-photon       v2.6.2    b43e7c963f06   4 weeks ago      112MB
goharbor/notary-signer-photon       v2.6.2    4e9982286316   4 weeks ago      109MB
goharbor/harbor-registryctl         v2.6.2    d403d4e0f990   4 weeks ago      137MB
goharbor/registry-photon            v2.6.2    596e445b18f0   4 weeks ago      77.5MB
goharbor/nginx-photon               v2.6.2    cbbdf28947a5   4 weeks ago      154MB
goharbor/harbor-log                 v2.6.2    3ce902023f73   4 weeks ago      161MB
goharbor/harbor-jobservice          v2.6.2    05791a75c8b8   4 weeks ago      250MB
goharbor/harbor-core                v2.6.2    484e2a64616d   4 weeks ago      213MB
goharbor/harbor-portal              v2.6.2    932297030d4e   4 weeks ago      162MB
goharbor/harbor-db                  v2.6.2    351b19e366cb   4 weeks ago      225MB
goharbor/prepare                    v2.6.2    eecc993c7643   4 weeks ago      164MB
hello-world                         latest    feb5d9fea6a5   14 months ago    13.3kB
10.0.2.46:80/myproject/hello-world  latest    feb5d9fea6a5   14 months ago    13.3kB
```

Important : Notez la spécification du port **80**. Sans spécifier le port le port par défaut **443** serait utilisé.

Connectez-vous à Harbor :

```
root@debian11:~/harbor# docker login 10.0.2.46:80
Username: admin
Password: Harbor12345
Error response from daemon: Get "https://10.0.2.46:80/v2/": http: server gave HTTP response to HTTPS client
```

Important : Notez l'erreur **Error response from daemon: Get**

“https://10.0.2.46:80/v2/”: http: server gave HTTP response to HTTPS client. Ceci est du à la configuration de docker dans le fichier **/etc/docker/daemon.json** où le port 80 n'a pas été spécifié :

```
root@debian11:~/harbor# cat /etc/docker/daemon.json
{
"insecure-registries" : ["10.0.2.46:5000", "0.0.0.0"]
}
```

Modifiez donc ce fichier :

```
root@debian11:~/harbor# vi /etc/docker/daemon.json
root@debian11:~/harbor# cat /etc/docker/daemon.json
{
"insecure-registries" : ["10.0.2.46:5000", "10.0.2.46:80", "0.0.0.0"]
}
```

Redémarrez le service docker :

```
root@debian11:~/harbor# systemctl restart docker
root@debian11:~/harbor# systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2022-12-11 08:57:36 CET; 20s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 487567 (dockerd)
      Tasks: 44
     Memory: 56.2M
        CPU: 2.275s
     CGroup: /system.slice/docker.service
             ├─487567 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
```

```
             ├─487820 /usr/bin/docker-proxy -proto tcp -host-ip 127.0.0.1 -host-port 1514 -contain>
             ├─488485 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 80 -container-i>
             └─488494 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 80 -container-ip 172>

Dec 11 08:57:14 debian11 dockerd[487567]: time="2022-12-11T08:57:14.186814789+01:00" level=info ms>
Dec 11 08:57:14 debian11 dockerd[487567]: time="2022-12-11T08:57:14.571406314+01:00" level=info ms>
Dec 11 08:57:19 debian11 dockerd[487567]: time="2022-12-11T08:57:19.175392656+01:00" level=info ms>
Dec 11 08:57:21 debian11 dockerd[487567]: time="2022-12-11T08:57:21.464318214+01:00" level=error m>
Dec 11 08:57:21 debian11 dockerd[487567]: time="2022-12-11T08:57:21.464360074+01:00" level=error m>
Dec 11 08:57:31 debian11 dockerd[487567]: time="2022-12-11T08:57:31.545302044+01:00" level=info ms>
Dec 11 08:57:35 debian11 dockerd[487567]: time="2022-12-11T08:57:35.749031343+01:00" level=info ms>
Dec 11 08:57:36 debian11 dockerd[487567]: time="2022-12-11T08:57:36.596800068+01:00" level=info ms>
```

Maintenant redémarrez Harbor :

```
root@debian11:~/harbor# docker-compose down -v
[+] Running 13/13
 ⠿ Container harbor-jobservice        Removed                                                  5.7s
 ⠿ Container nginx                    Removed                                                  5.8s
 ⠿ Container chartmuseum              Removed                                                  0.2s
 ⠿ Container trivy-adapter            Removed                                                  5.8s
 ⠿ Container registryctl              Removed                                                 13.9s
 ⠿ Container harbor-portal            Removed                                                  4.7s
 ⠿ Container harbor-core              Removed                                                  5.3s
 ⠿ Container harbor-db                Removed                                                  8.9s
 ⠿ Container registry                 Removed                                                  4.0s
 ⠿ Container redis                    Removed                                                  8.7s
 ⠿ Container harbor-log               Removed                                                 14.2s
 ⠿ Network harbor_harbor              Removed                                                  0.8s
 ⠿ Network harbor_harbor-chartmuseum  Removed                                                  0.6s

root@debian11:~/harbor# docker-compose up -d
[+] Running 13/13
 ⠿ Network harbor_harbor              Created                                                  0.4s
```

```
⠿ Network harbor_harbor-chartmuseum  Created                                  0.3s
⠿ Container harbor-log               Started                                13.5s
⠿ Container harbor-db                Started                                12.9s
⠿ Container harbor-portal            Started                                14.8s
⠿ Container chartmuseum              Started                                14.3s
⠿ Container redis                    Started                                16.8s
⠿ Container registryctl              Started                                16.3s
⠿ Container registry                 Started                                15.2s
⠿ Container trivy-adapter            Started                                14.0s
⠿ Container harbor-core              Started                                14.9s
⠿ Container harbor-jobservice        Started                                14.8s
⠿ Container nginx                    Started                                15.0s
```

Connectez-vous à Harbor :

```
root@debian11:~/harbor# docker login 10.0.2.46:80
Username: admin
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

Les coordonnées de connexion sont stockées dans le fichier **/root/.docker/config.json** :

```
root@debian11:~/harbor# cat /root/.docker/config.json
{
        "auths": {
                "10.0.2.46:80": {
                        "auth": "YWRtaW46SGFyYm9yMTIzNDU="
                }
        }
```

```
}root@debian11:~/harbor#
```

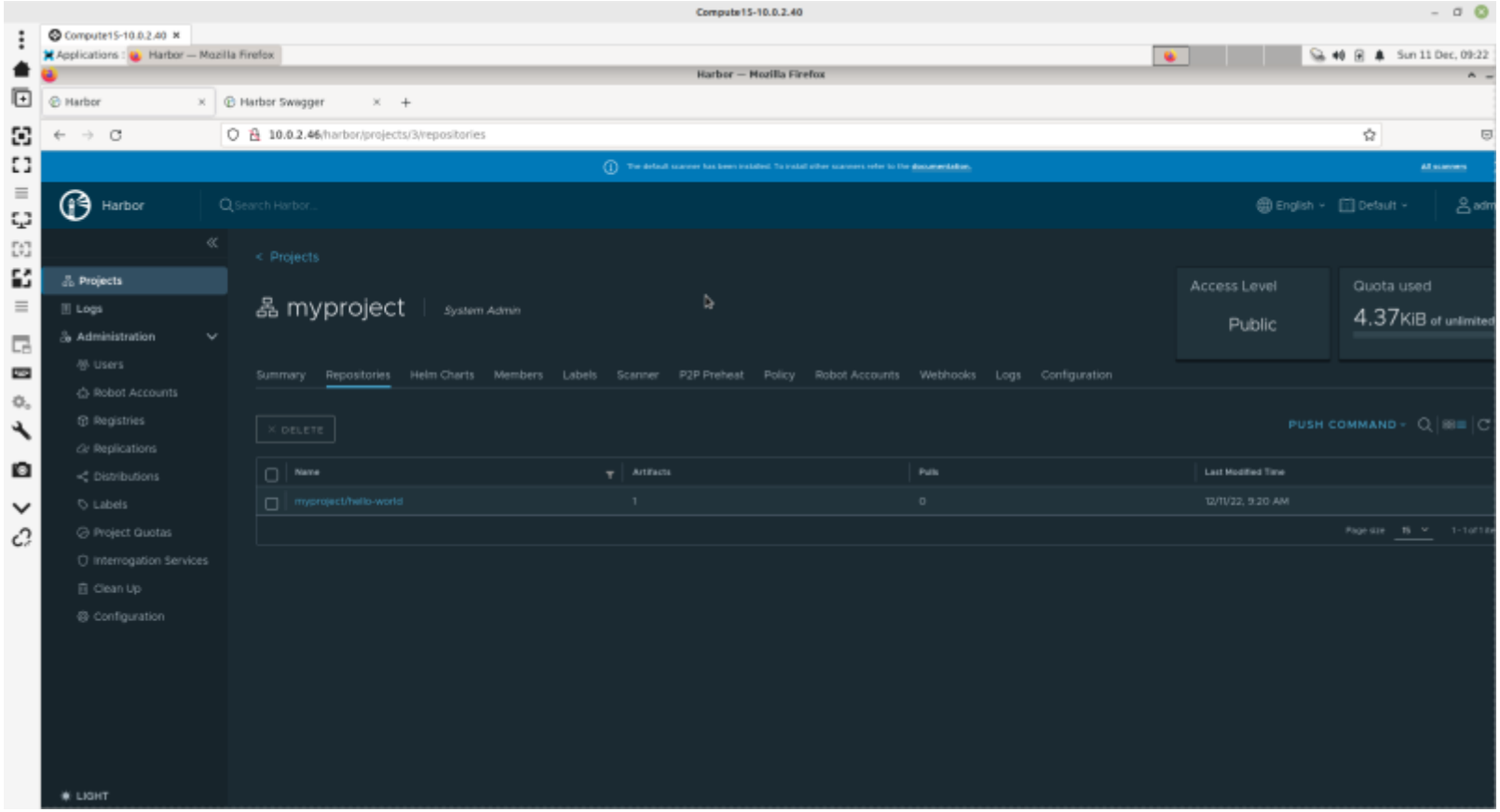
De cette façon, lors des logins futur, docker ne demandera ni le nom de l'utilisateur, ni le mot de passe :

```
root@debian11:~/harbor# docker logout
Removing login credentials for https://index.docker.io/v1/

root@debian11:~/harbor# docker login 10.0.2.46:80
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

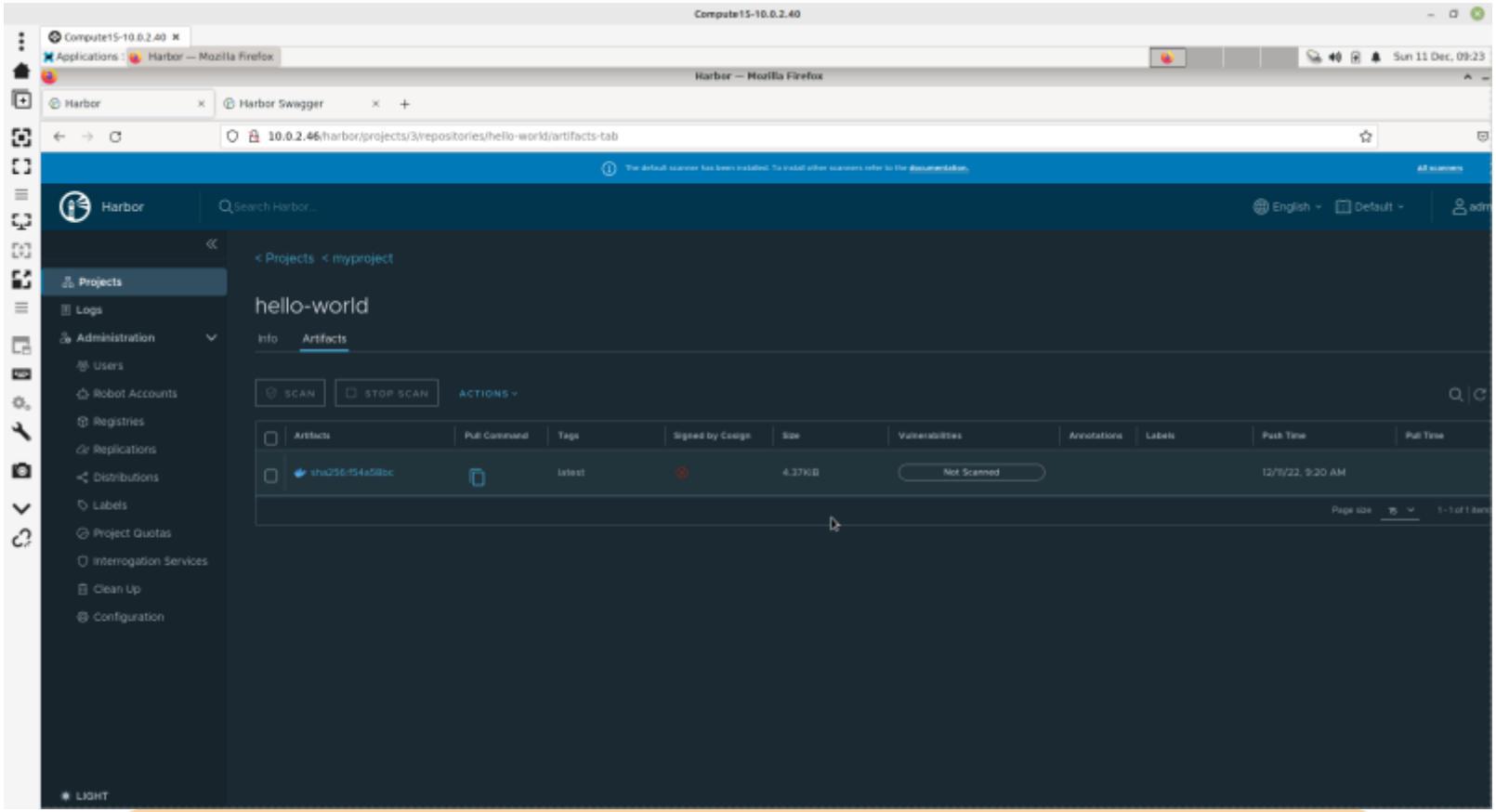
Procédez maintenant au **push** :

```
root@debian11:~/harbor# docker push 10.0.2.46:80/myproject/hello-world
Using default tag: latest
The push refers to repository [10.0.2.46:80/myproject/hello-world]
e07ee1baac5f: Pushed
latest: digest: sha256:f54a58bc1aac5ea1a25d796ae155dc228b3f0e11d046ae276b39c4bf2f13d8c4 size: 525
```

Retournez à Harbor et constatez la présence du dépôt :



Cliquez sur le nom du dépôt et constatez son contenu :



Important : Notez que l'image stocké dans Harbor est appeléé un **Artefact**.

5.5 - Pousser des Images Supplémentaires

Modifiez le tag de l'image hello-world :

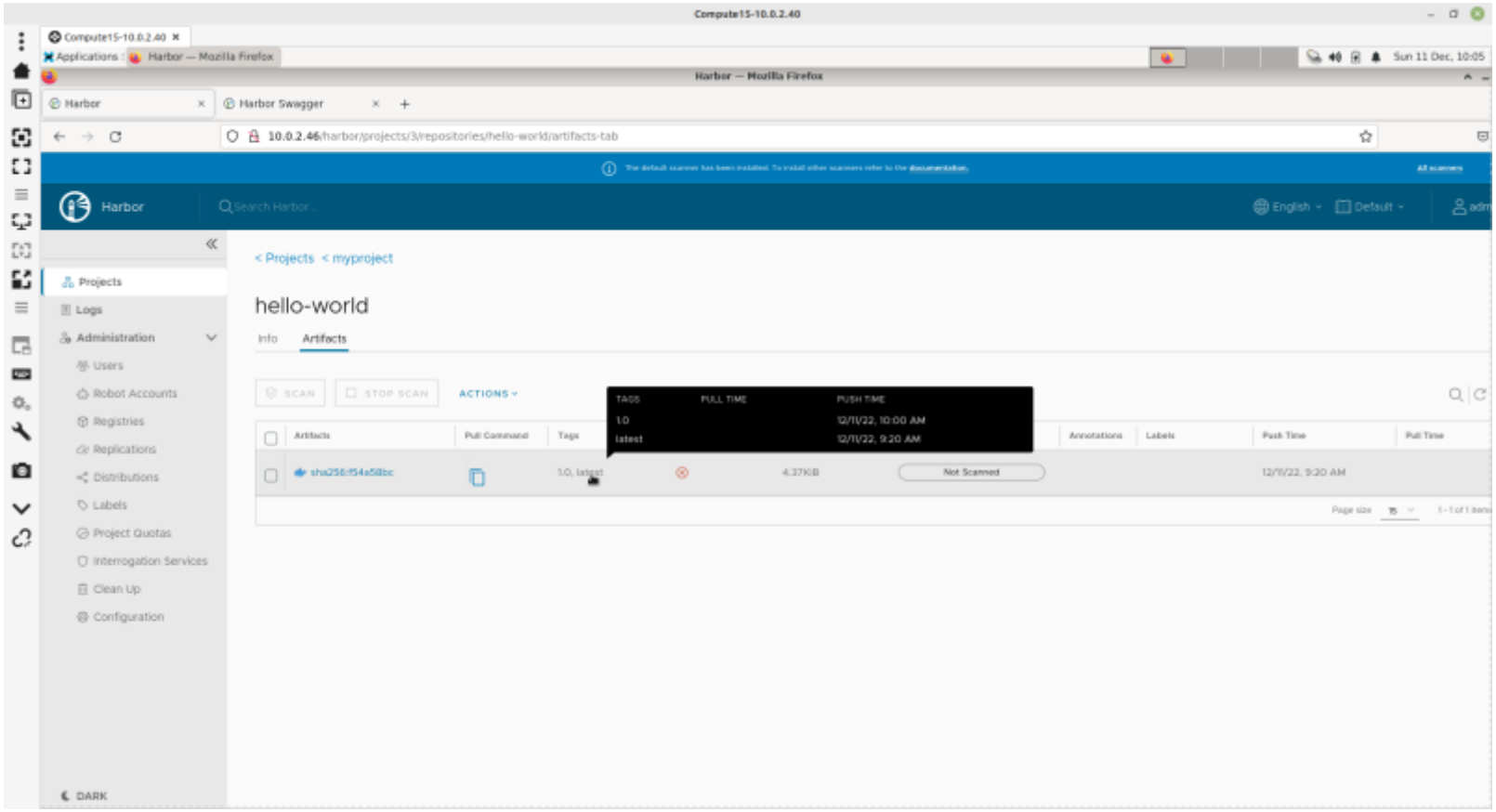
```
root@debian11:~/harbor# docker tag hello-world 10.0.2.46:80/myproject/hello-world:1.0
```

Procédez maintenant au **push** :

```
root@debian11:~/harbor# docker push 10.0.2.46:80/myproject/hello-world:1.0
The push refers to repository [10.0.2.46:80/myproject/hello-world]
e07ee1baac5f: Layer already exists
1.0: digest: sha256:f54a58bc1aac5ea1a25d796ae155dc228b3f0e11d046ae276b39c4bf2f13d8c4 size: 525
```

Important : Notez que la valeur du SHA256 est indentique au push précédent. De cette façon Docker n'a pas poussé l'image mais seulement le nouveau tag.

Constatez le résultat dans Harbor :



Téléchargez maintenant l'image de **alpine** à partir de docker.io :

```
root@debian11:~/harbor# docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
c158987b0551: Pull complete
Digest: sha256:8914eb54f968791faf6a8638949e480fef81e697984fba772b3976835194c6d4
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
```

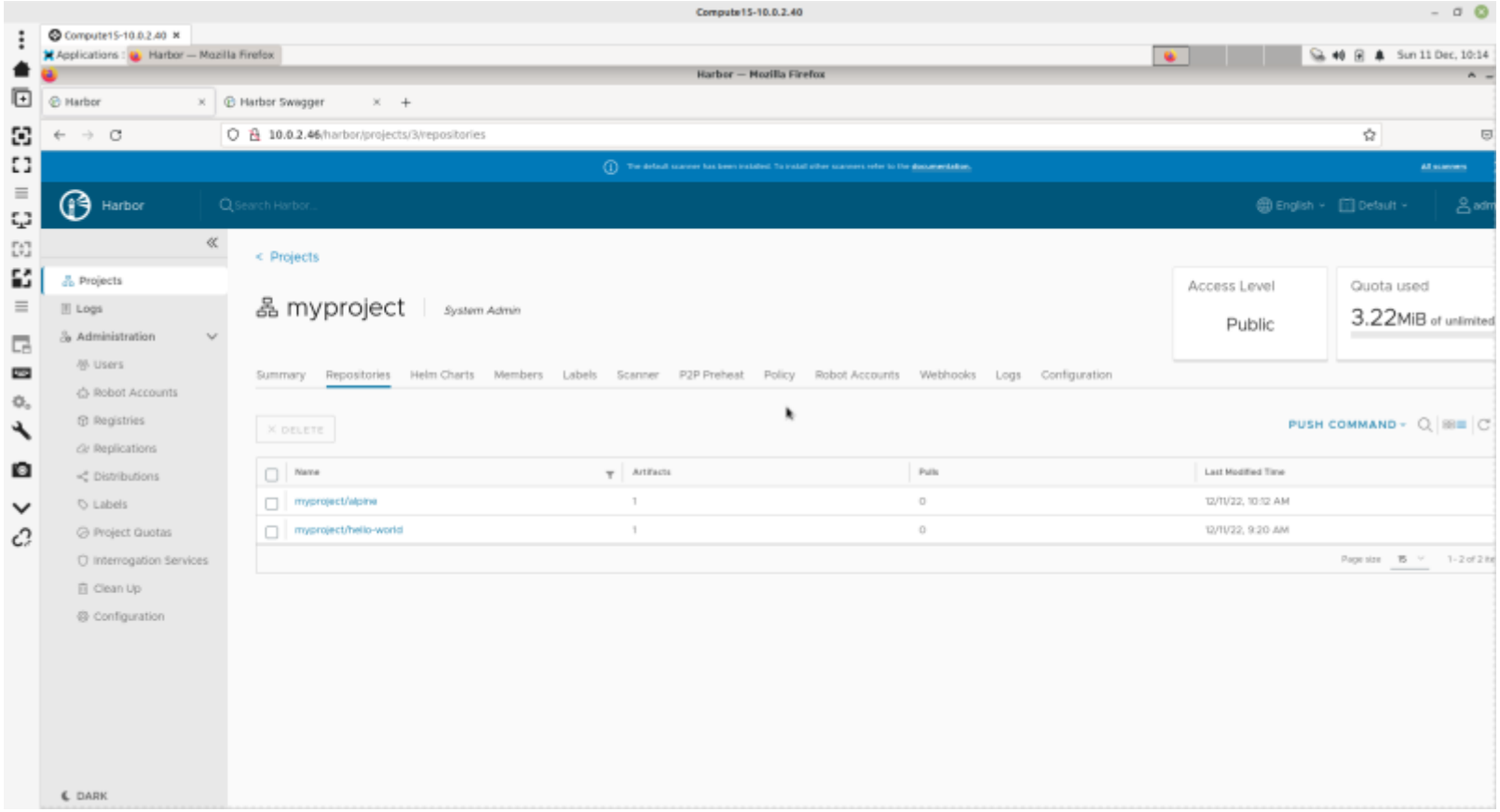
Modifiez le tag de l'image alpine :

```
root@debian11:~/harbor# docker tag alpine 10.0.2.46:80/myproject/alpine
```

Procédez maintenant au **push** :

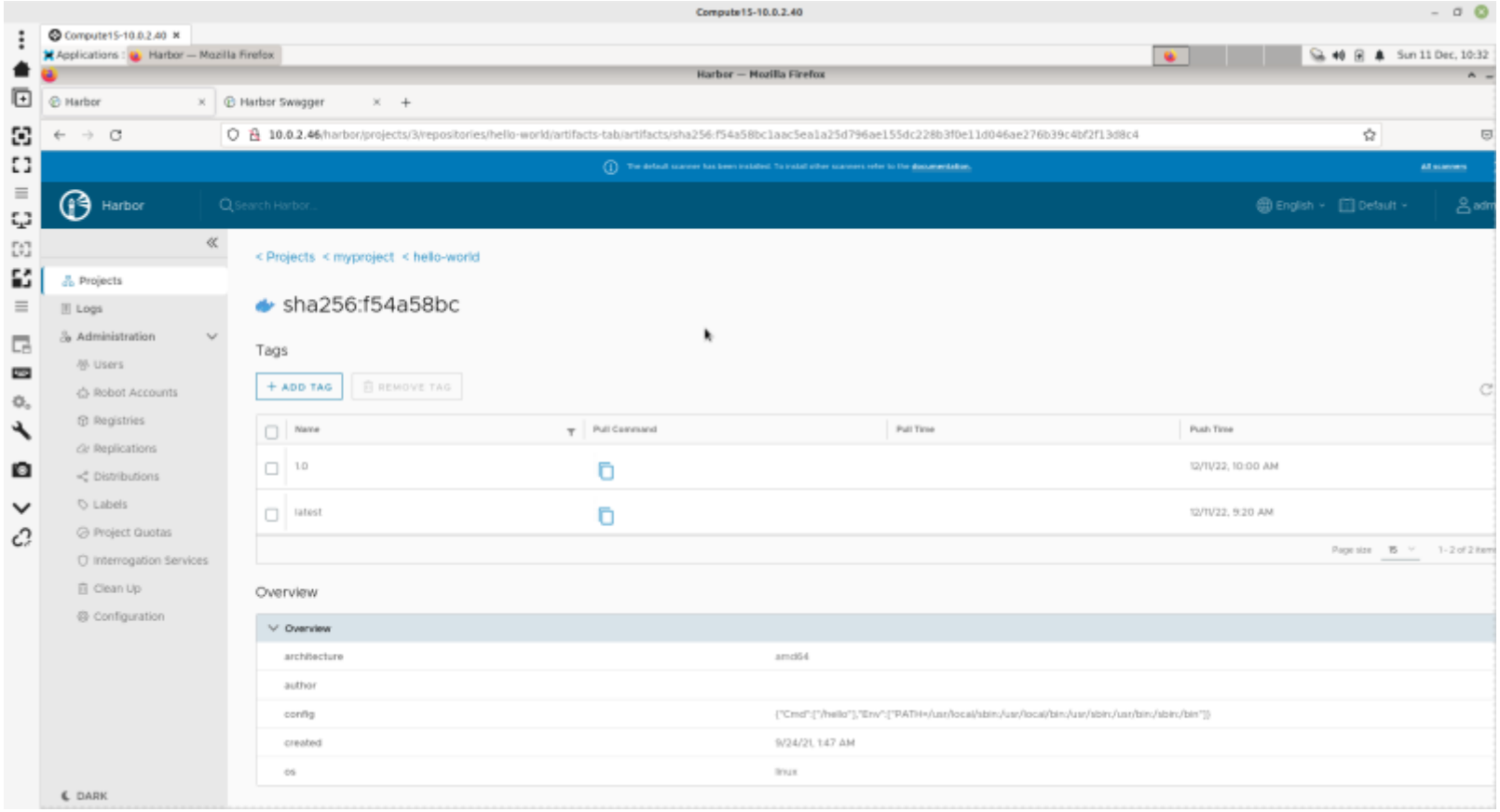
```
root@debian11:~/harbor# docker push 10.0.2.46:80/myproject/alpine
Using default tag: latest
The push refers to repository [10.0.2.46:80/myproject/alpine]
ded7a220bb05: Pushed
latest: digest: sha256:c0d488a800e4127c334ad20d61d7bc21b4097540327217dfab52262adc02380c size: 528
```

Constatez le résultat dans Harbor :

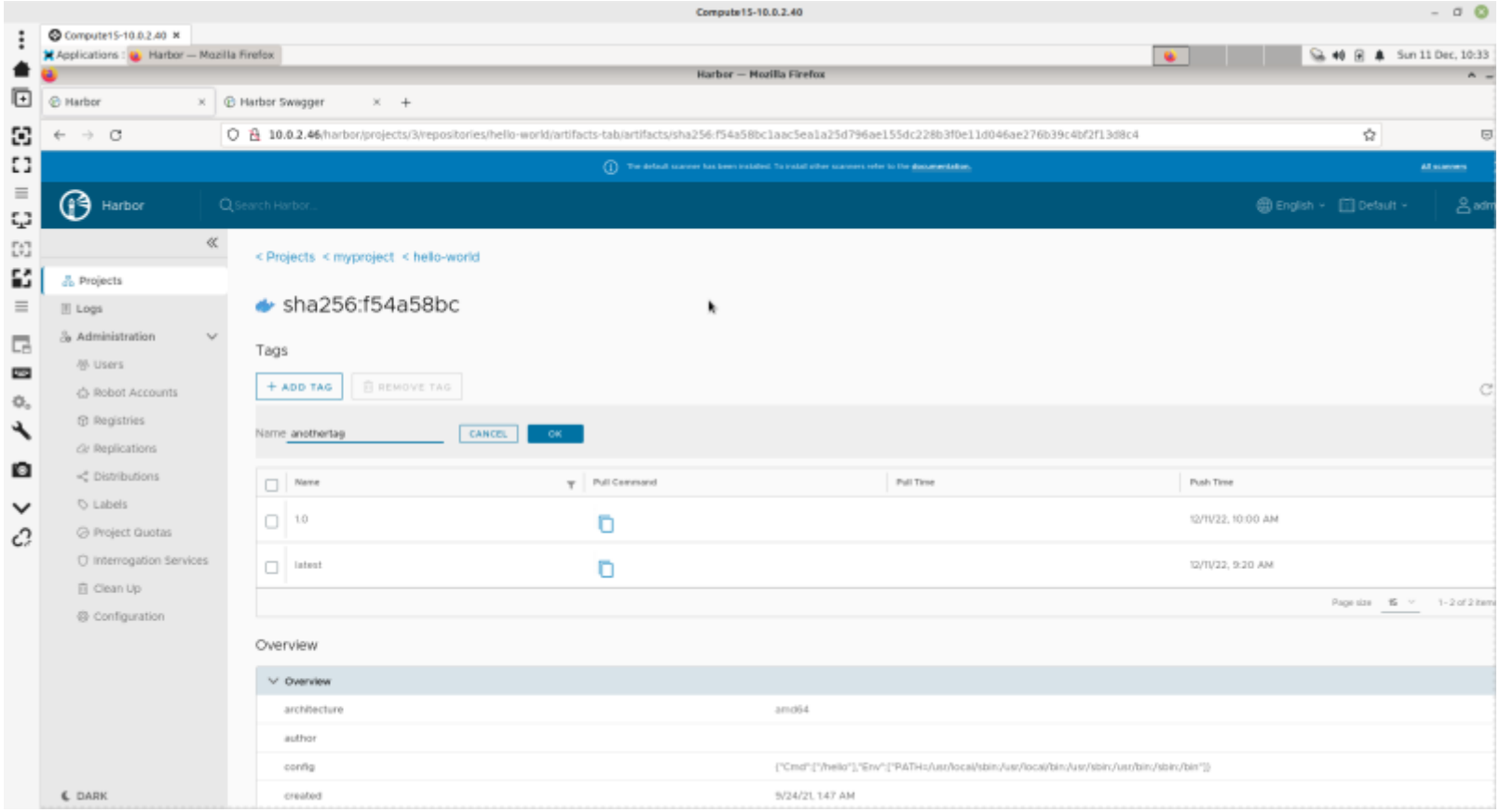


5.6 - Travailler avec des Artefacts

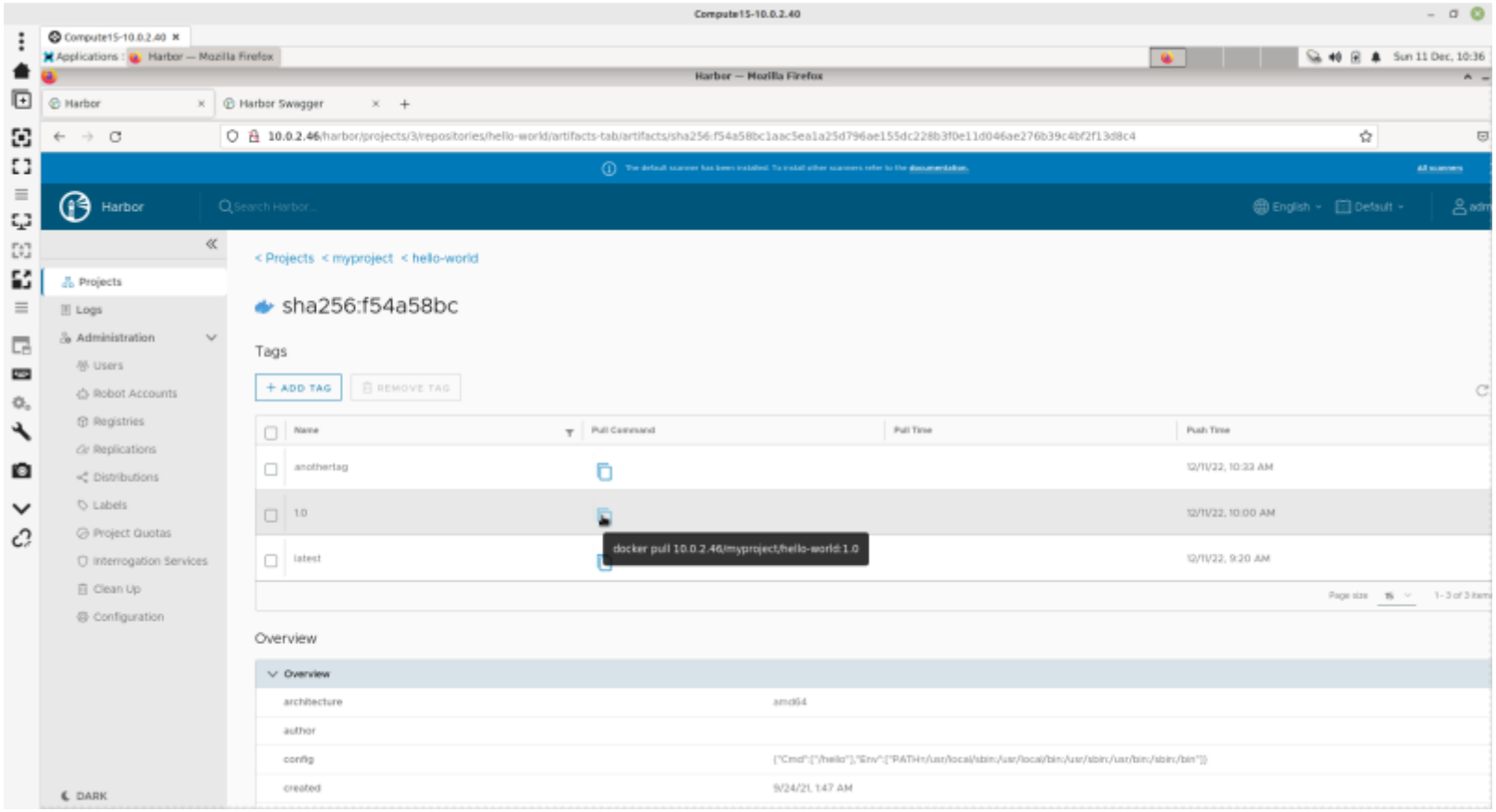
Cliquez sur **Projects > myproject > myproject/hello-world** puis sur le nom de l'Artefact :



Cliquez ensuite sur le bouton **ADD TAG** et ajoutez le tag **anothertag** :



Sur la ligne 1.0 et dans la colonne **Pull Command**, cliquez sur l'icône :



Revenez à l'interface en ligne de commande et consultez la liste des images :

```
REPOSITORY                          TAG       IMAGE ID       CREATED          SIZE
10.0.2.46:80/myproject/alpine       latest    49176f190c7e   2 weeks ago      7.05MB
alpine                              latest    49176f190c7e   2 weeks ago      7.05MB
goharbor/harbor-exporter            v2.6.2    ae230e335853   4 weeks ago      95.4MB
goharbor/chartmuseum-photon         v2.6.2    be361eee8a36   4 weeks ago      225MB
goharbor/redis-photon               v2.6.2    84fa3669bb10   4 weeks ago      154MB
goharbor/trivy-adapter-photon       v2.6.2    beef2527c7b2   4 weeks ago      441MB
goharbor/notary-server-photon       v2.6.2    b43e7c963f06   4 weeks ago      112MB
goharbor/notary-signer-photon       v2.6.2    4e9982286316   4 weeks ago      109MB
```

```
goharbor/harbor-registryctl          v2.6.2    d403d4e0f990   4 weeks ago     137MB
goharbor/registry-photon             v2.6.2    596e445b18f0   4 weeks ago     77.5MB
goharbor/nginx-photon                v2.6.2    cbbdf28947a5   4 weeks ago     154MB
goharbor/harbor-log                  v2.6.2    3ce902023f73   4 weeks ago     161MB
goharbor/harbor-jobservice           v2.6.2    05791a75c8b8   4 weeks ago     250MB
goharbor/harbor-core                 v2.6.2    484e2a64616d   4 weeks ago     213MB
goharbor/harbor-portal               v2.6.2    932297030d4e   4 weeks ago     162MB
goharbor/harbor-db                   v2.6.2    351b19e366cb   4 weeks ago     225MB
goharbor/prepare                     v2.6.2    eecc993c7643   4 weeks ago     164MB
10.0.2.46:80/myproject/hello-world   1.0       feb5d9fea6a5   14 months ago   13.3kB
10.0.2.46:80/myproject/hello-world   latest    feb5d9fea6a5   14 months ago   13.3kB
hello-world                          latest    feb5d9fea6a5   14 months ago   13.3kB
```

Supprimez l'image **10.0.2.46:80/myproject/hello-world:1.0** :

```
root@debian11:~/harbor# docker image rm 10.0.2.46:80/myproject/hello-world:1.0
Untagged: 10.0.2.46:80/myproject/hello-world:1.0
```

Consultez la liste des images de nouveau :

```
root@debian11:~/harbor# docker images
REPOSITORY                           TAG       IMAGE ID       CREATED         SIZE
10.0.2.46:80/myproject/alpine        latest    49176f190c7e   2 weeks ago     7.05MB
alpine                               latest    49176f190c7e   2 weeks ago     7.05MB
goharbor/harbor-exporter             v2.6.2    ae230e335853   4 weeks ago     95.4MB
goharbor/chartmuseum-photon          v2.6.2    be361eee8a36   4 weeks ago     225MB
goharbor/redis-photon                v2.6.2    84fa3669bb10   4 weeks ago     154MB
goharbor/trivy-adapter-photon        v2.6.2    beef2527c7b2   4 weeks ago     441MB
goharbor/notary-server-photon        v2.6.2    b43e7c963f06   4 weeks ago     112MB
goharbor/notary-signer-photon        v2.6.2    4e9982286316   4 weeks ago     109MB
goharbor/harbor-registryctl          v2.6.2    d403d4e0f990   4 weeks ago     137MB
goharbor/registry-photon             v2.6.2    596e445b18f0   4 weeks ago     77.5MB
goharbor/nginx-photon                v2.6.2    cbbdf28947a5   4 weeks ago     154MB
goharbor/harbor-log                  v2.6.2    3ce902023f73   4 weeks ago     161MB
```

```
goharbor/harbor-jobservice           v2.6.2    05791a75c8b8   4 weeks ago     250MB
goharbor/harbor-core                 v2.6.2    484e2a64616d   4 weeks ago     213MB
goharbor/harbor-portal               v2.6.2    932297030d4e   4 weeks ago     162MB
goharbor/harbor-db                   v2.6.2    351b19e366cb   4 weeks ago     225MB
goharbor/prepare                     v2.6.2    eecc993c7643   4 weeks ago     164MB
10.0.2.46:80/myproject/hello-world   latest    feb5d9fea6a5   14 months ago   13.3kB
hello-world                          latest    feb5d9fea6a5   14 months ago   13.3kB
```

Téléchargez maintenant l'image de **10.0.2.46:80/myproject/hello-world:1.0** à partir de Harbor :

```
root@debian11:~/harbor# docker pull 10.0.2.46:80/myproject/hello-world:1.0
1.0: Pulling from myproject/hello-world
Digest: sha256:f54a58bc1aac5ea1a25d796ae155dc228b3f0e11d046ae276b39c4bf2f13d8c4
Status: Downloaded newer image for 10.0.2.46:80/myproject/hello-world:1.0
10.0.2.46:80/myproject/hello-world:1.0
```

Lister maintenant les images :

```
root@debian11:~/harbor# docker images
REPOSITORY                           TAG       IMAGE ID       CREATED         SIZE
10.0.2.46:80/myproject/alpine        latest    49176f190c7e   2 weeks ago     7.05MB
alpine                               latest    49176f190c7e   2 weeks ago     7.05MB
goharbor/harbor-exporter             v2.6.2    ae230e335853   4 weeks ago     95.4MB
goharbor/chartmuseum-photon          v2.6.2    be361eee8a36   4 weeks ago     225MB
goharbor/redis-photon                v2.6.2    84fa3669bb10   4 weeks ago     154MB
goharbor/trivy-adapter-photon        v2.6.2    beef2527c7b2   4 weeks ago     441MB
goharbor/notary-server-photon        v2.6.2    b43e7c963f06   4 weeks ago     112MB
goharbor/notary-signer-photon        v2.6.2    4e9982286316   4 weeks ago     109MB
goharbor/harbor-registryctl          v2.6.2    d403d4e0f990   4 weeks ago     137MB
goharbor/registry-photon             v2.6.2    596e445b18f0   4 weeks ago     77.5MB
goharbor/nginx-photon                v2.6.2    cbbdf28947a5   4 weeks ago     154MB
goharbor/harbor-log                  v2.6.2    3ce902023f73   4 weeks ago     161MB
goharbor/harbor-jobservice           v2.6.2    05791a75c8b8   4 weeks ago     250MB
goharbor/harbor-core                 v2.6.2    484e2a64616d   4 weeks ago     213MB
```

```
goharbor/harbor-portal                v2.6.2    932297030d4e   4 weeks ago     162MB
goharbor/harbor-db                    v2.6.2    351b19e366cb   4 weeks ago     225MB
goharbor/prepare                      v2.6.2    eecc993c7643   4 weeks ago     164MB
hello-world                           latest    feb5d9fea6a5   14 months ago   13.3kB
10.0.2.46:80/myproject/hello-world    1.0       feb5d9fea6a5   14 months ago   13.3kB
10.0.2.46:80/myproject/hello-world    latest    feb5d9fea6a5   14 months ago   13.3kB
```

Modifiez le tag de l'image téléchargée :

```
root@debian11:~/harbor# docker tag hello-world 10.0.2.46:80/myproject/production:latest

root@debian11:~/harbor# docker images
REPOSITORY                            TAG       IMAGE ID       CREATED         SIZE
10.0.2.46:80/myproject/alpine         latest    49176f190c7e   2 weeks ago     7.05MB
alpine                                latest    49176f190c7e   2 weeks ago     7.05MB
goharbor/harbor-exporter              v2.6.2    ae230e335853   4 weeks ago     95.4MB
goharbor/chartmuseum-photon           v2.6.2    be361eee8a36   4 weeks ago     225MB
goharbor/redis-photon                 v2.6.2    84fa3669bb10   4 weeks ago     154MB
goharbor/trivy-adapter-photon         v2.6.2    beef2527c7b2   4 weeks ago     441MB
goharbor/notary-server-photon         v2.6.2    b43e7c963f06   4 weeks ago     112MB
goharbor/notary-signer-photon         v2.6.2    4e9982286316   4 weeks ago     109MB
goharbor/harbor-registryctl           v2.6.2    d403d4e0f990   4 weeks ago     137MB
goharbor/registry-photon              v2.6.2    596e445b18f0   4 weeks ago     77.5MB
goharbor/nginx-photon                 v2.6.2    cbbdf28947a5   4 weeks ago     154MB
goharbor/harbor-log                   v2.6.2    3ce902023f73   4 weeks ago     161MB
goharbor/harbor-jobservice            v2.6.2    05791a75c8b8   4 weeks ago     250MB
goharbor/harbor-core                  v2.6.2    484e2a64616d   4 weeks ago     213MB
goharbor/harbor-portal                v2.6.2    932297030d4e   4 weeks ago     162MB
goharbor/harbor-db                    v2.6.2    351b19e366cb   4 weeks ago     225MB
goharbor/prepare                      v2.6.2    eecc993c7643   4 weeks ago     164MB
10.0.2.46:80/myproject/hello-world    1.0       feb5d9fea6a5   14 months ago   13.3kB
10.0.2.46:80/myproject/hello-world    latest    feb5d9fea6a5   14 months ago   13.3kB
10.0.2.46:80/myproject/production     latest    feb5d9fea6a5   14 months ago   13.3kB
```

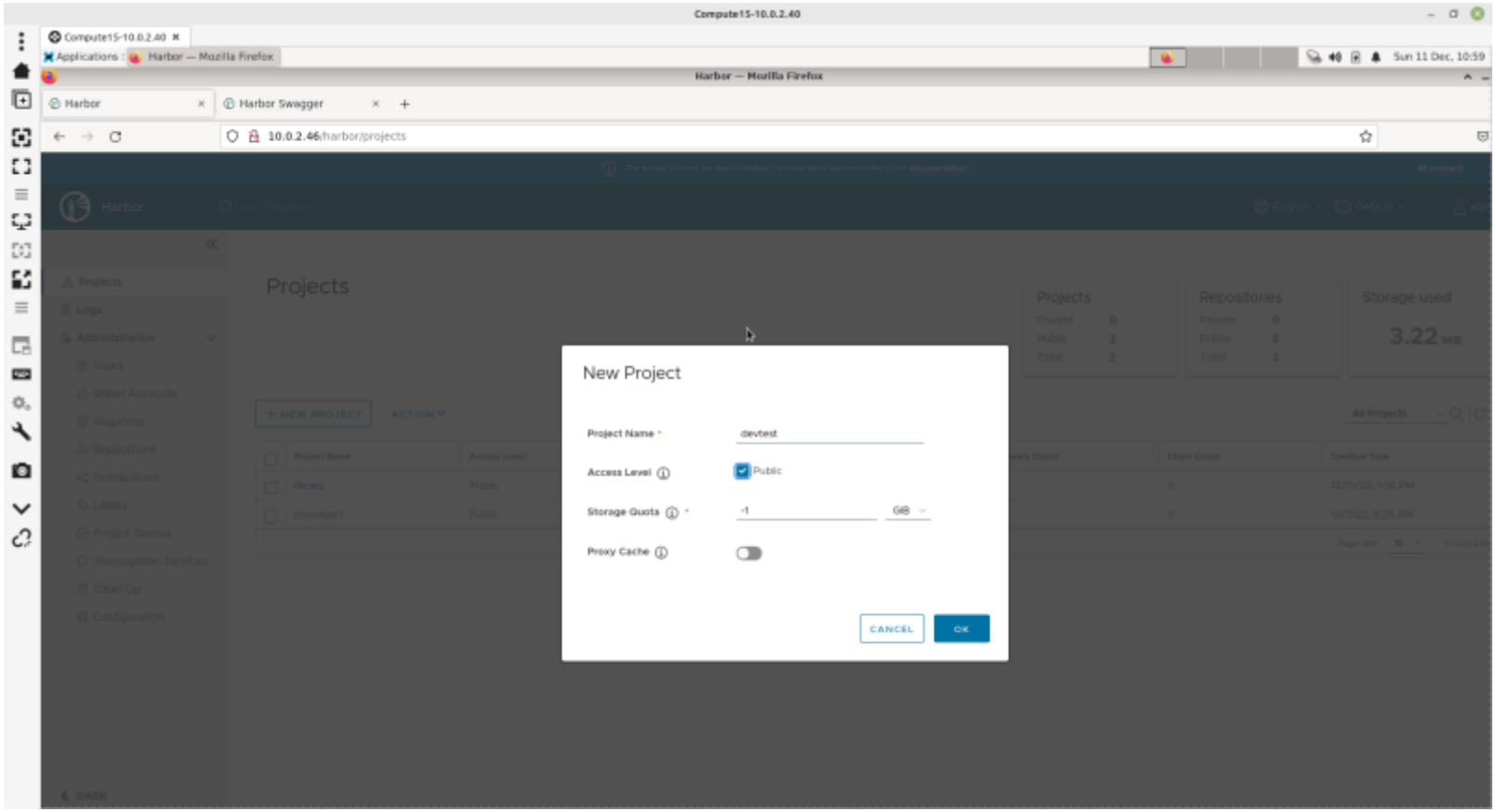
```
hello-world                          latest    feb5d9fea6a5   14 months ago   13.3kB
```

Procédez maintenant au **push** :

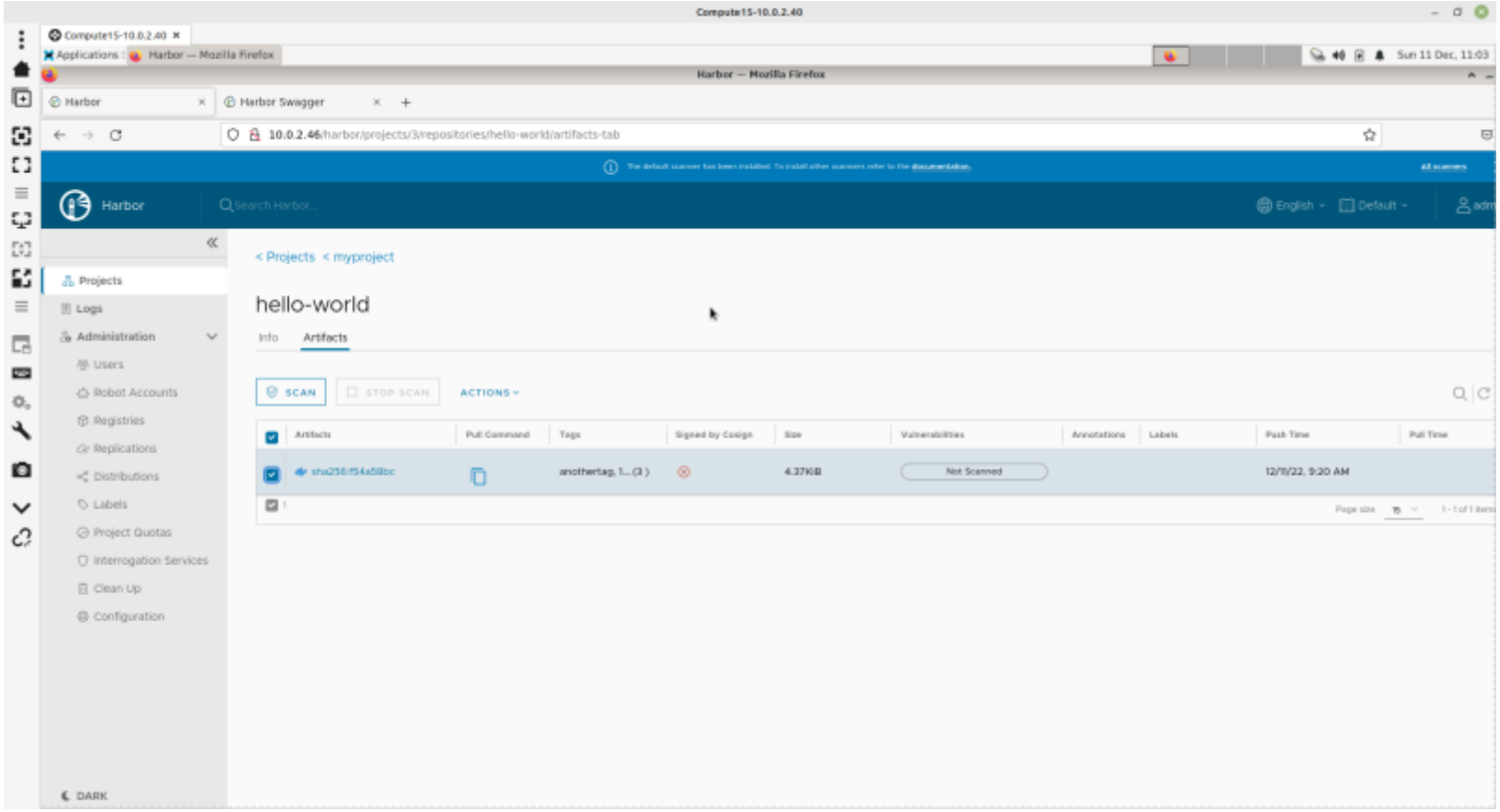
```
root@debian11:~/harbor# docker push 10.0.2.46:80/myproject/production:latest
The push refers to repository [10.0.2.46:80/myproject/production]
e07ee1baac5f: Mounted from myproject/hello-world
latest: digest: sha256:f54a58bc1aac5ea1a25d796ae155dc228b3f0e11d046ae276b39c4bf2f13d8c4 size: 525
```

5.7 - Copier un Artefact d'un Projet vers un autre Projet

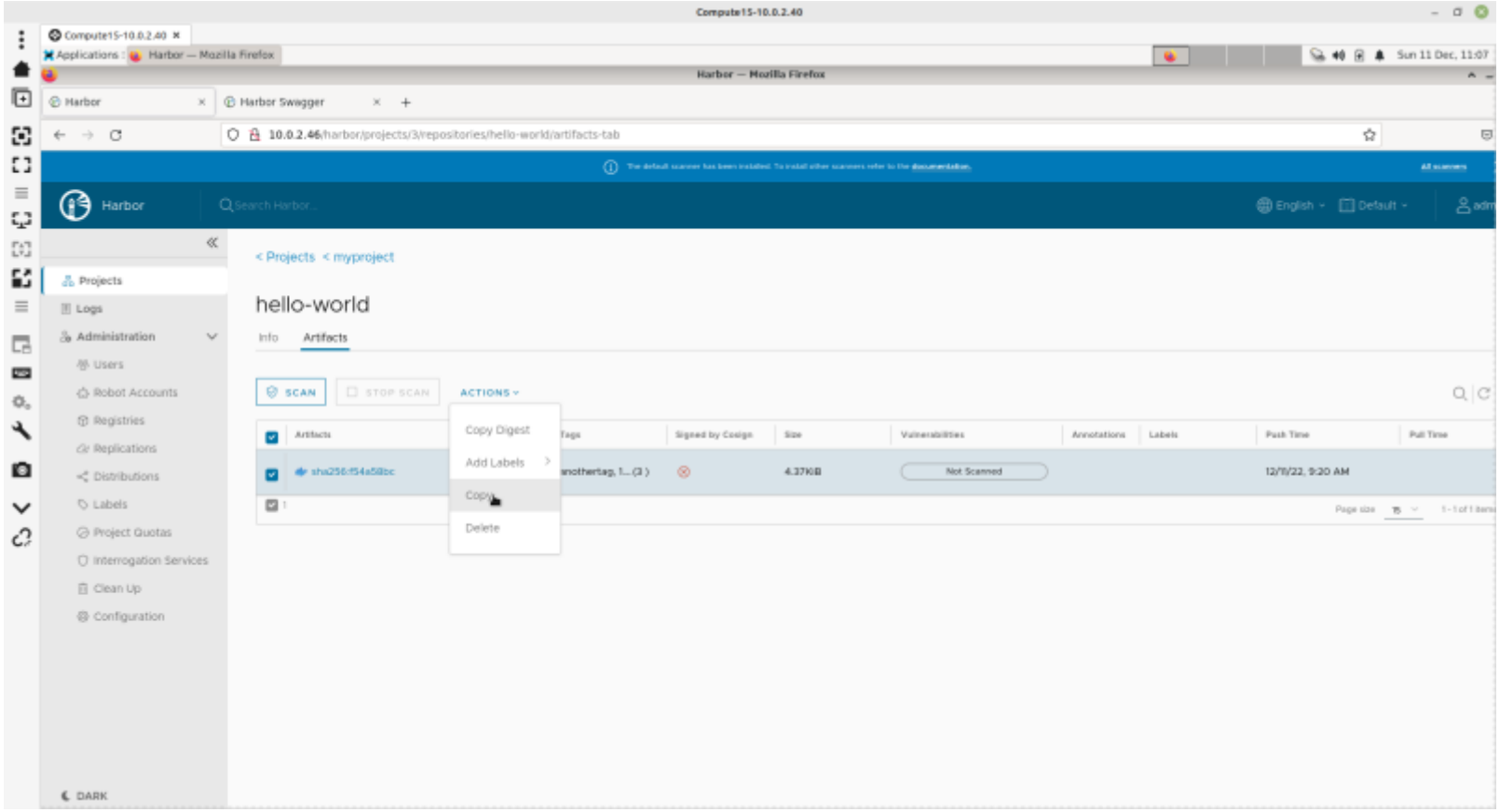
Commencez par créer un autre projet appelé **devtest** :



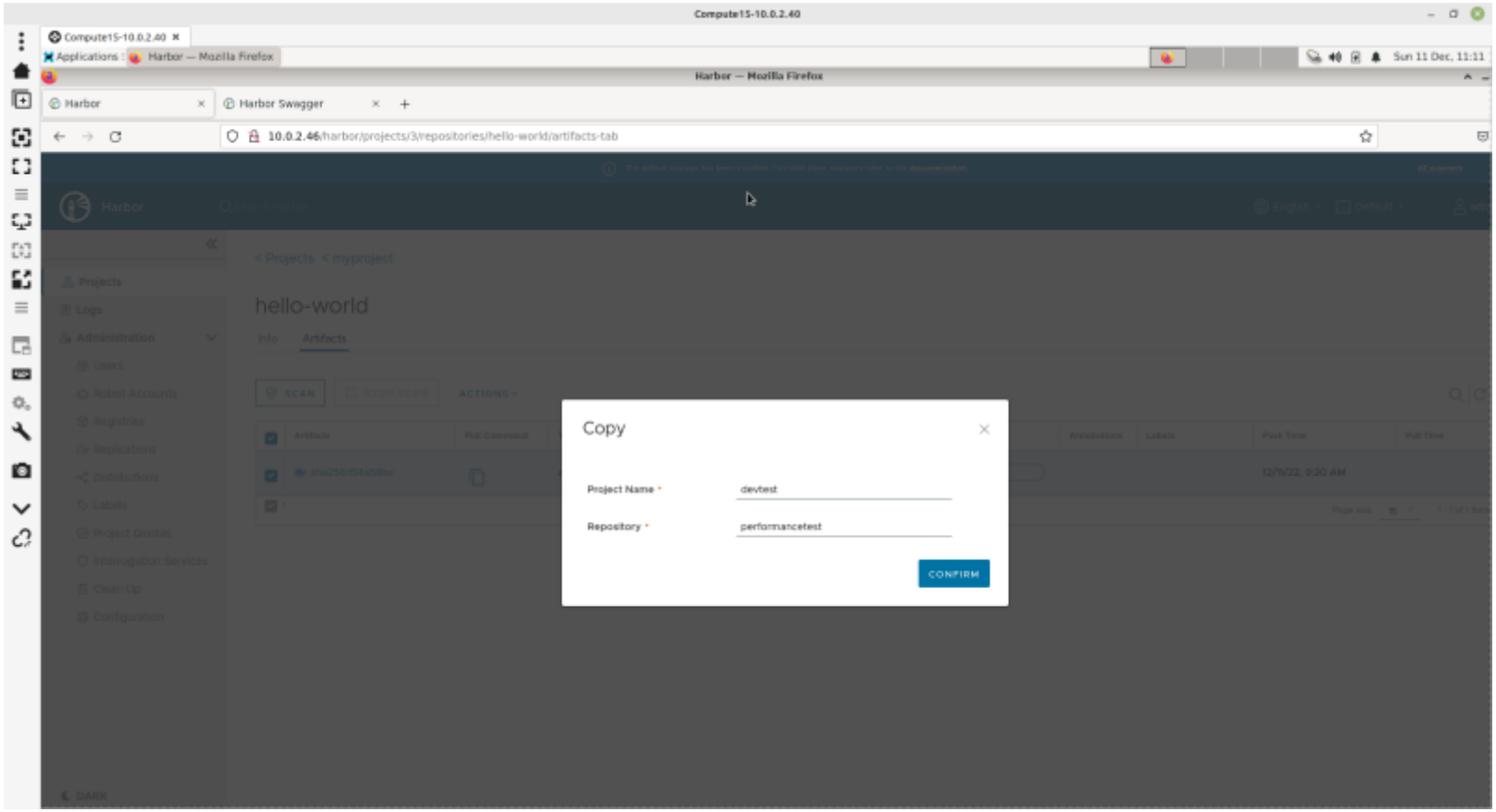
Cliquez sur **Projects > myproject > myproject/hello-world** et cochez l'Artefact :



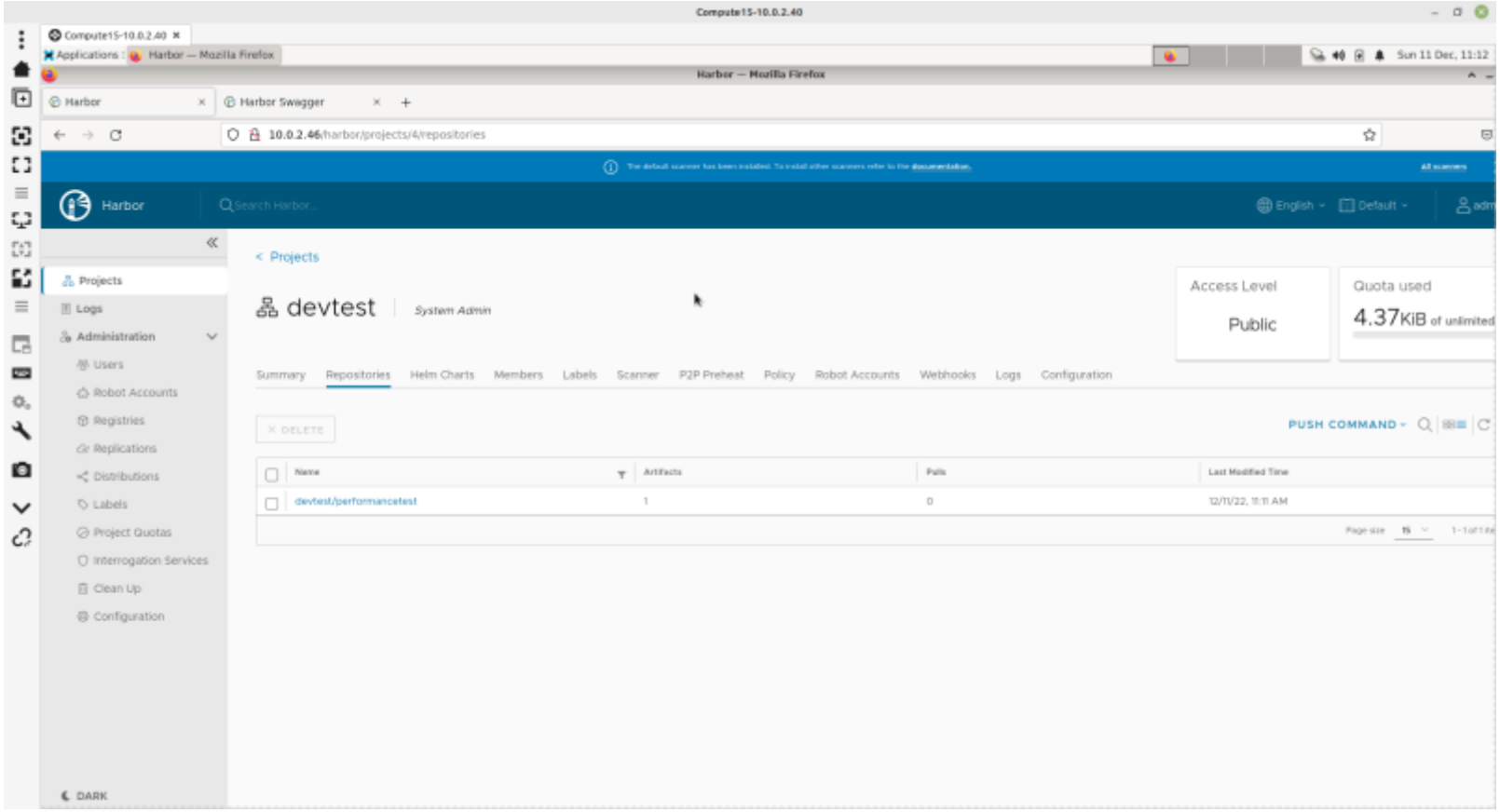
Cliquez ensuite sur **ACTIONS > Copy** :



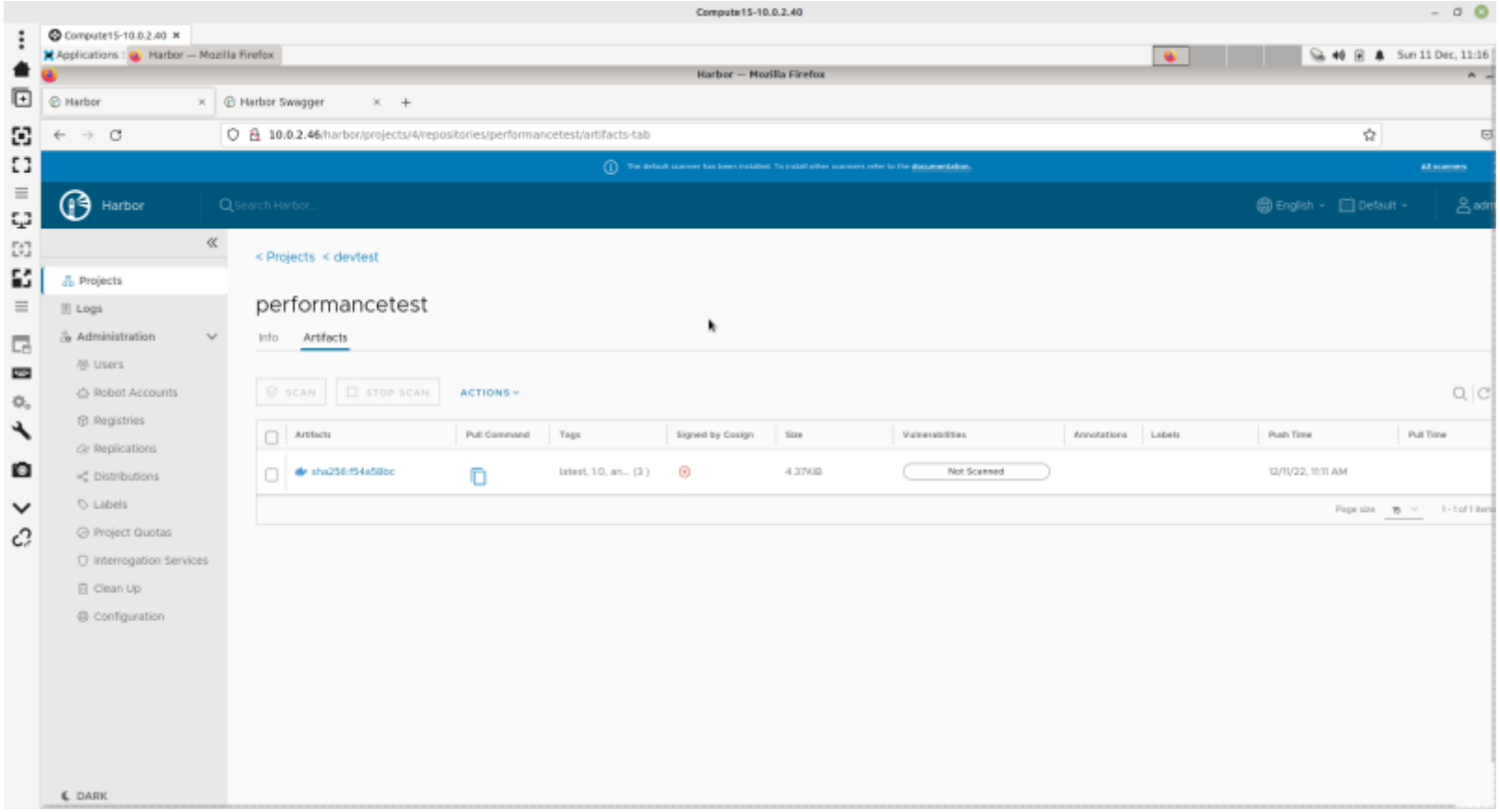
Renseignez les champs et cliquez sur le bouton **CONFIRM** :



Cliquez sur **Projects > devtest** :

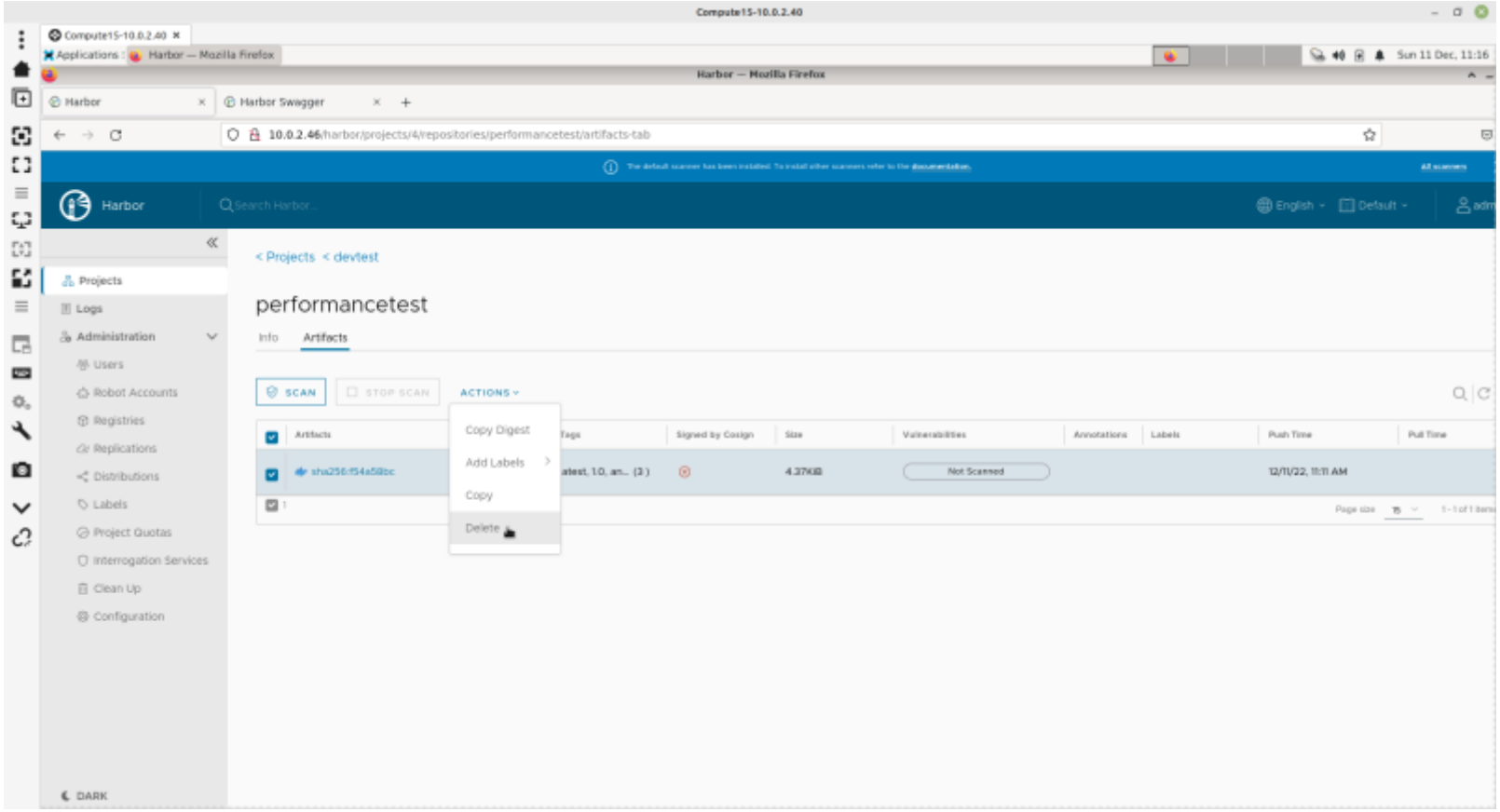


Cliquez ensuite sur **devtest/performancetest** :

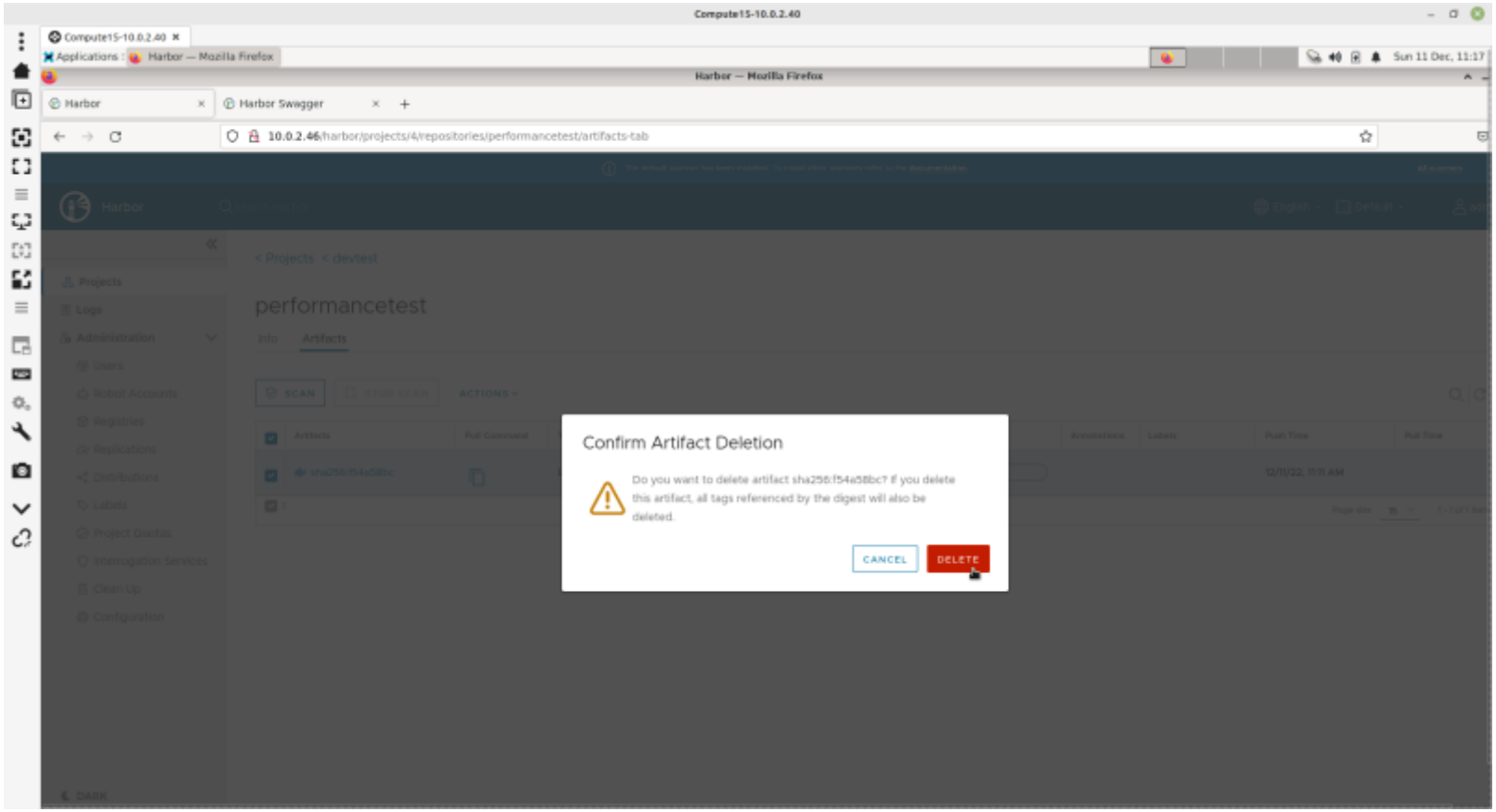


5.8 - Suppression d'un Artefact, d'un Dépôt et d'un Projet

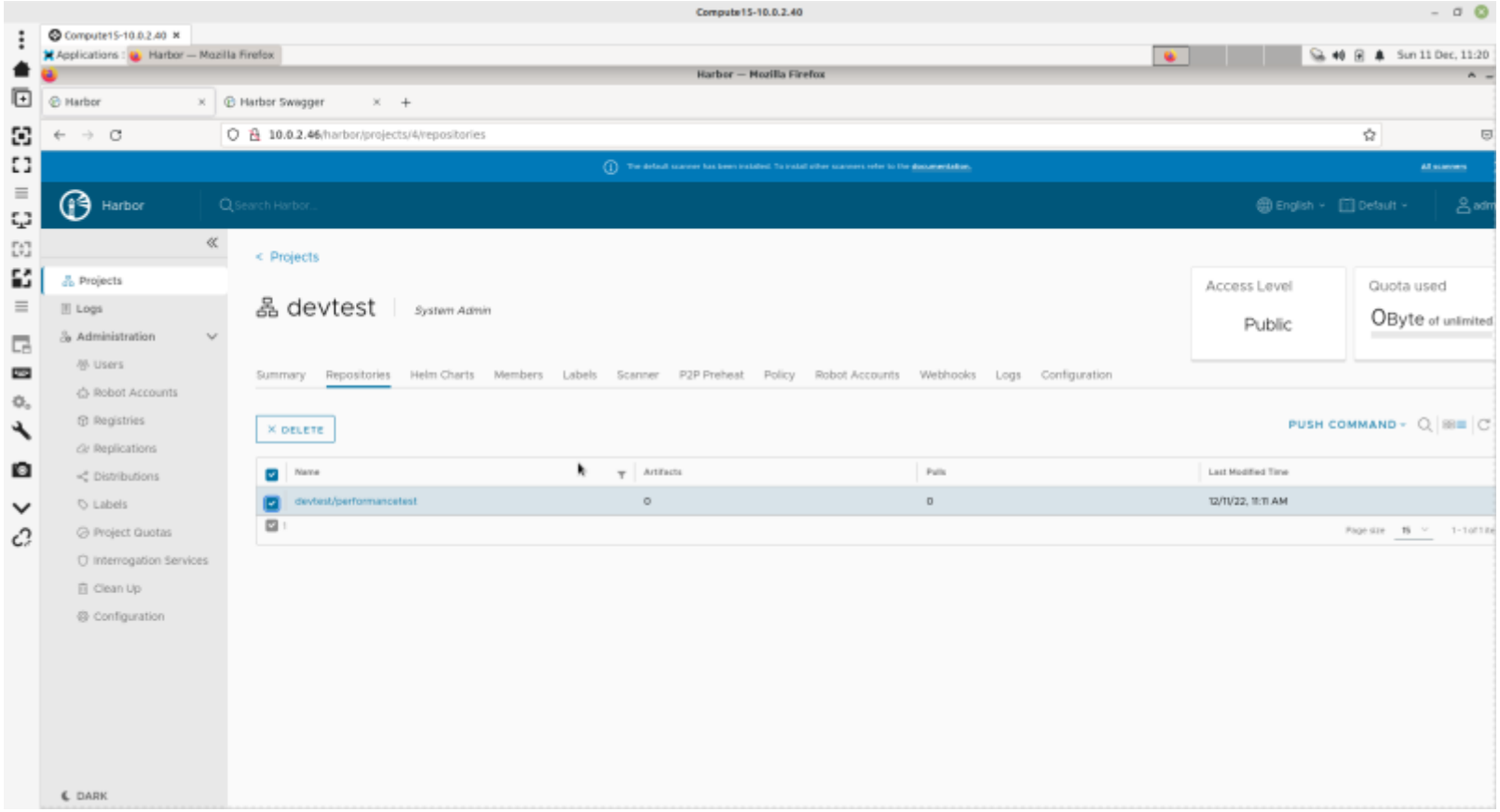
Sélectionnez l'Artefact puis cliquez sur **ACTIONS > Delete** :



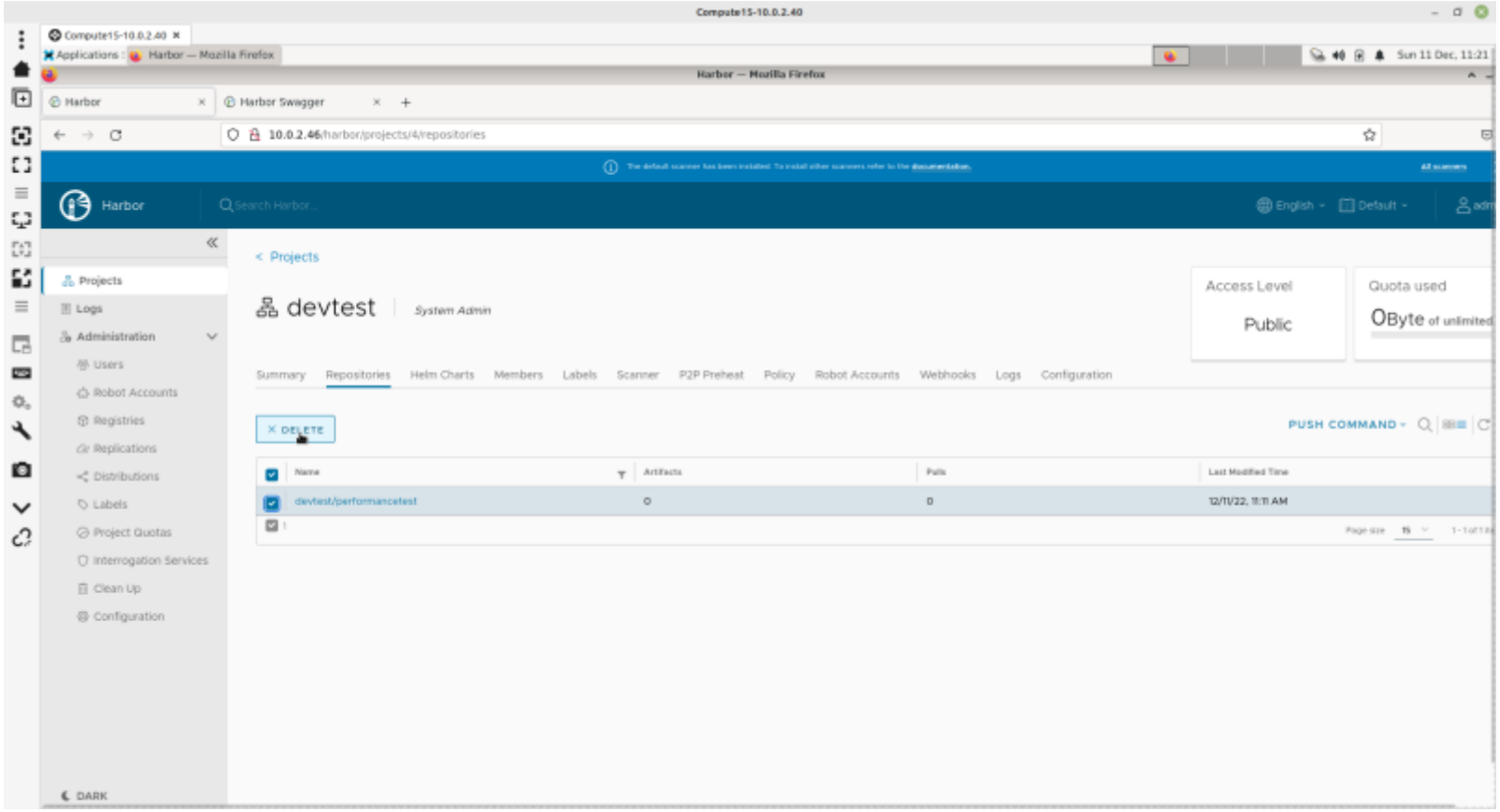
Confirmez la suppression en cliquant sur le bouton **CONFIRM** :



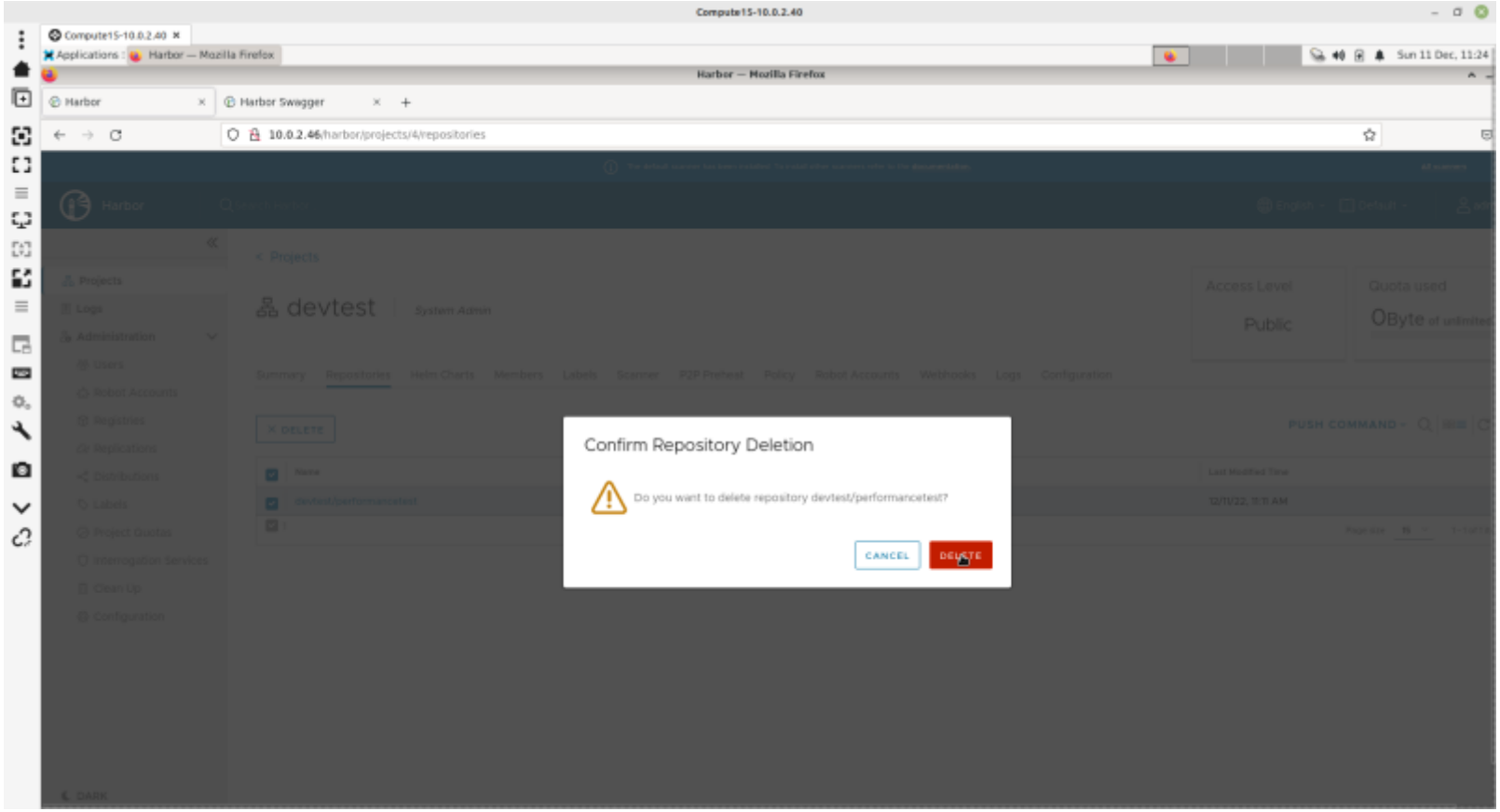
Cliquez sur **devtest** et cochez **devtest/Performancetest** :



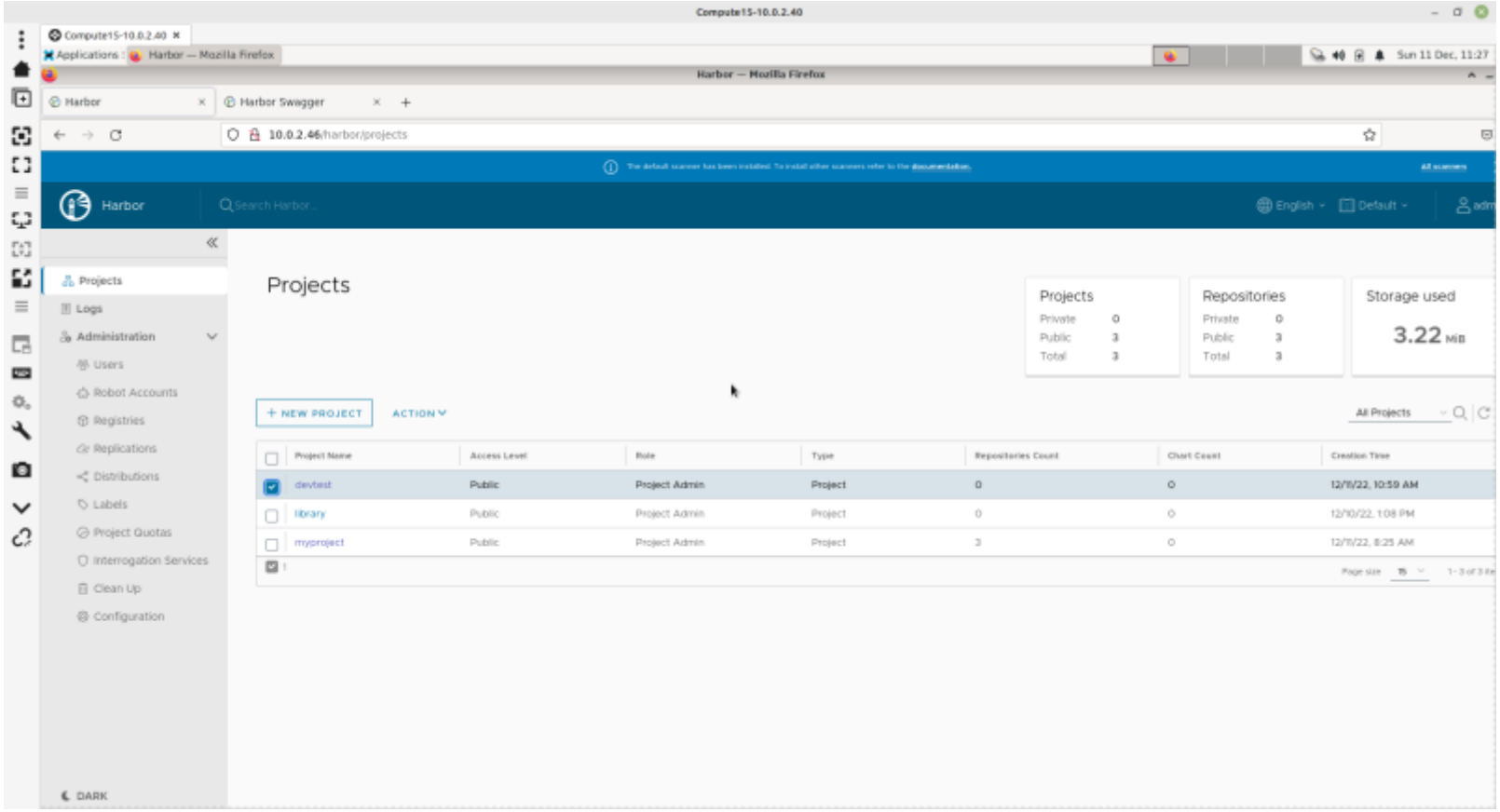
Cliquez sur le bouton **DELETE** :



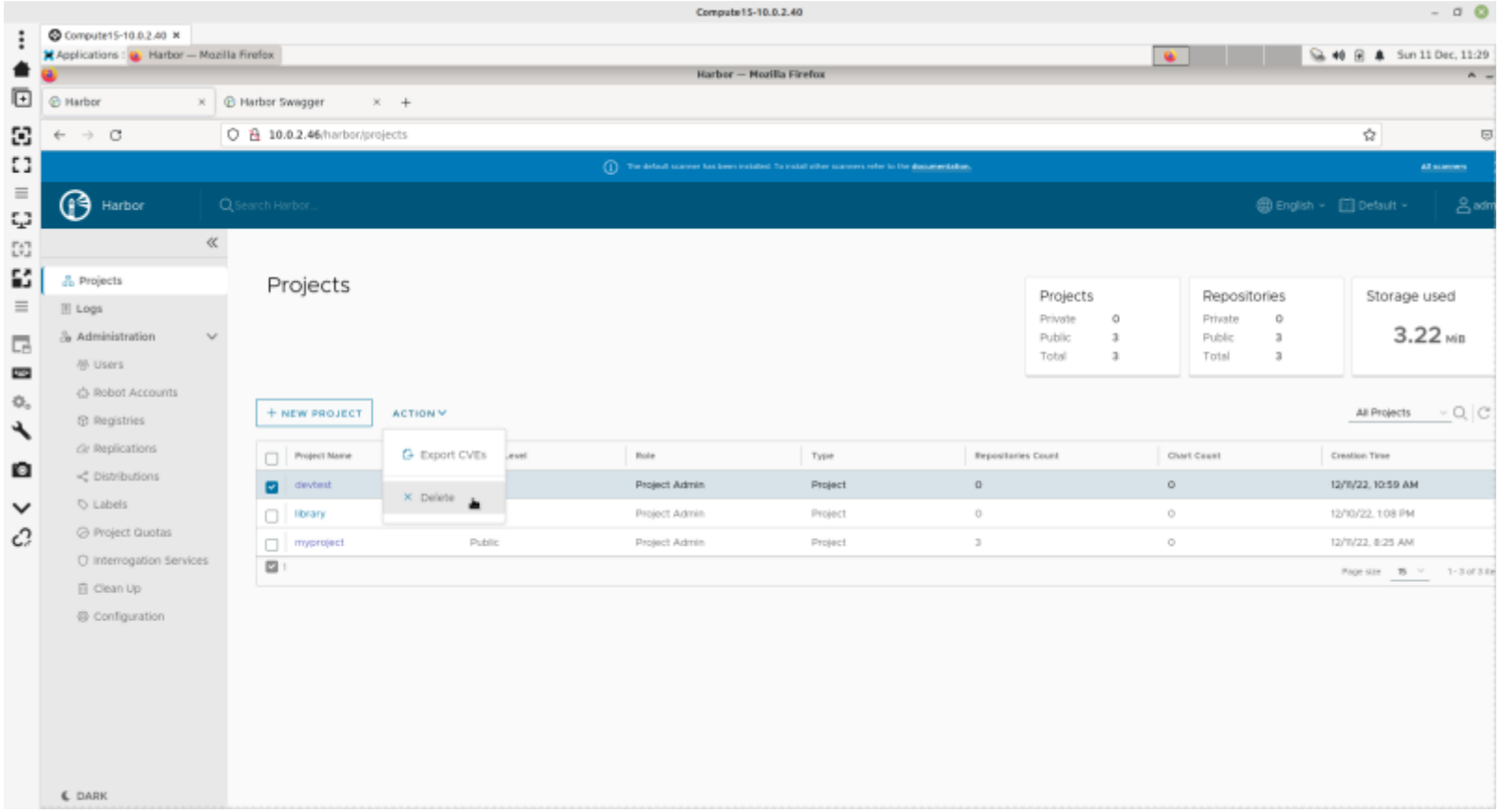
Confirmez la suppression en cliquant sur le bouton **CONFIRM** :



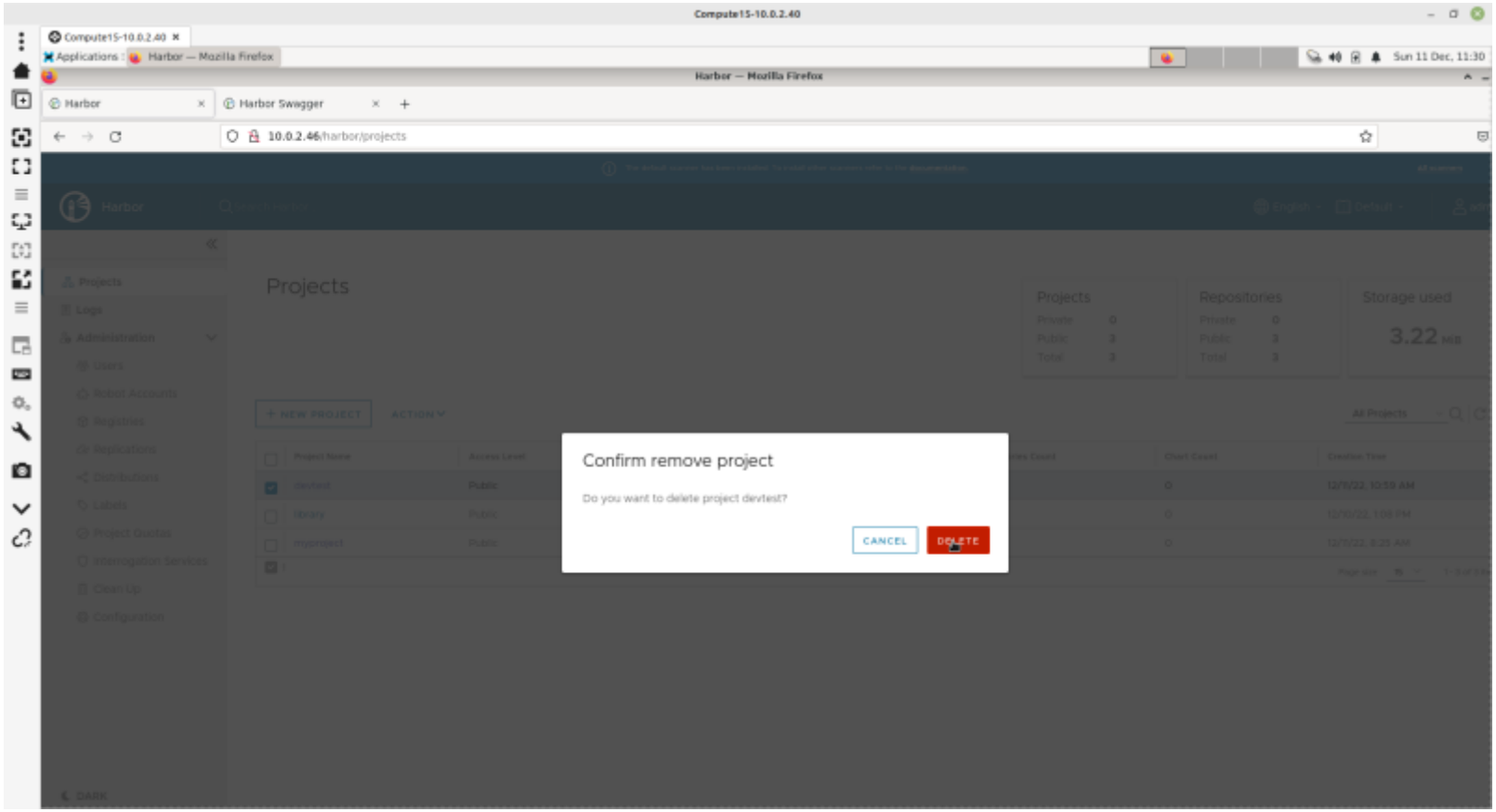
Cliquez sur **Projects** puis cochez **devtest** :



Cliquez sur **ACTION** puis sur **Delete** :



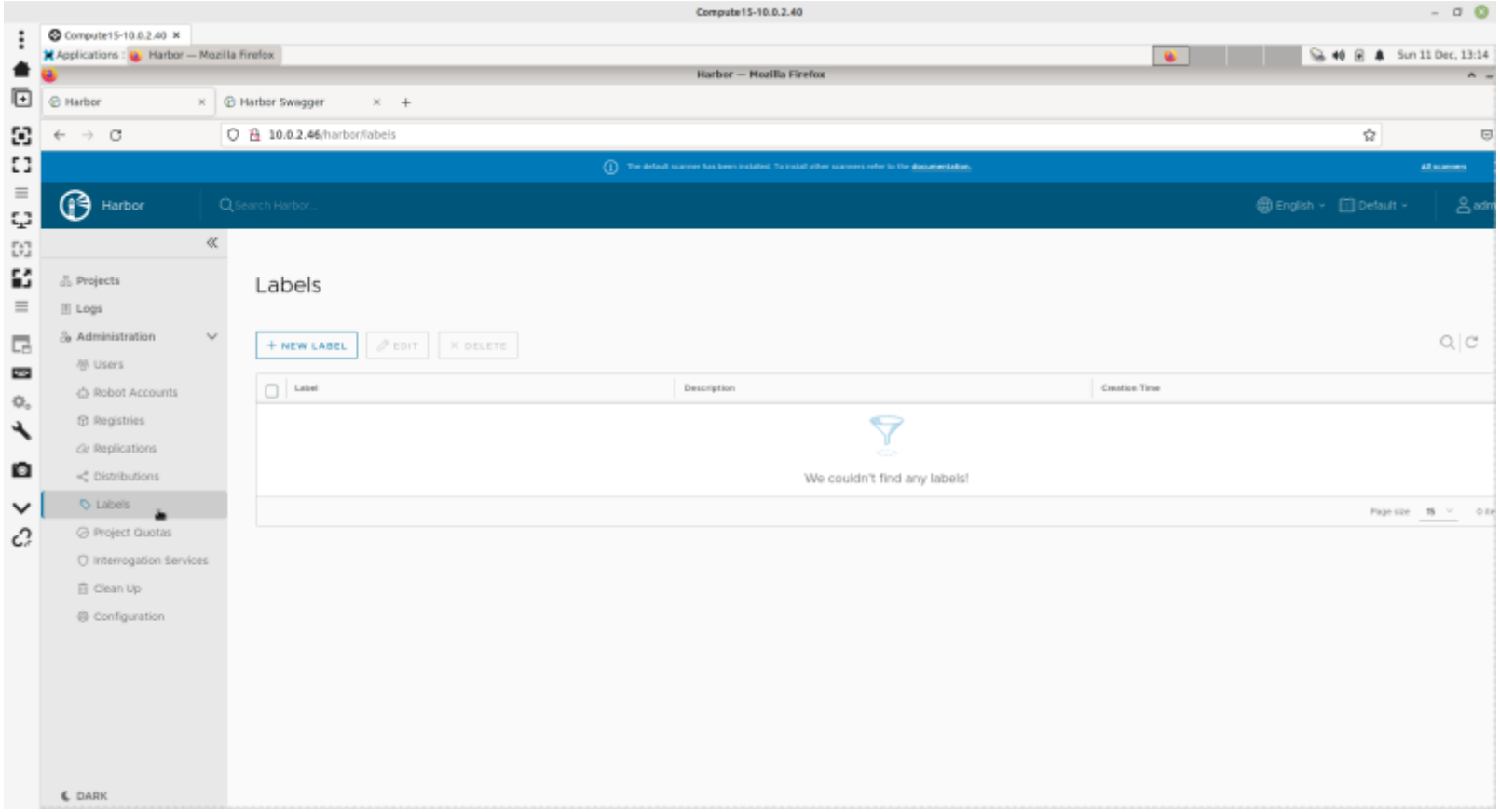
Confirmez la suppression en cliquant sur le bouton **CONFIRM** :



5.8 - Utilisation de Labels

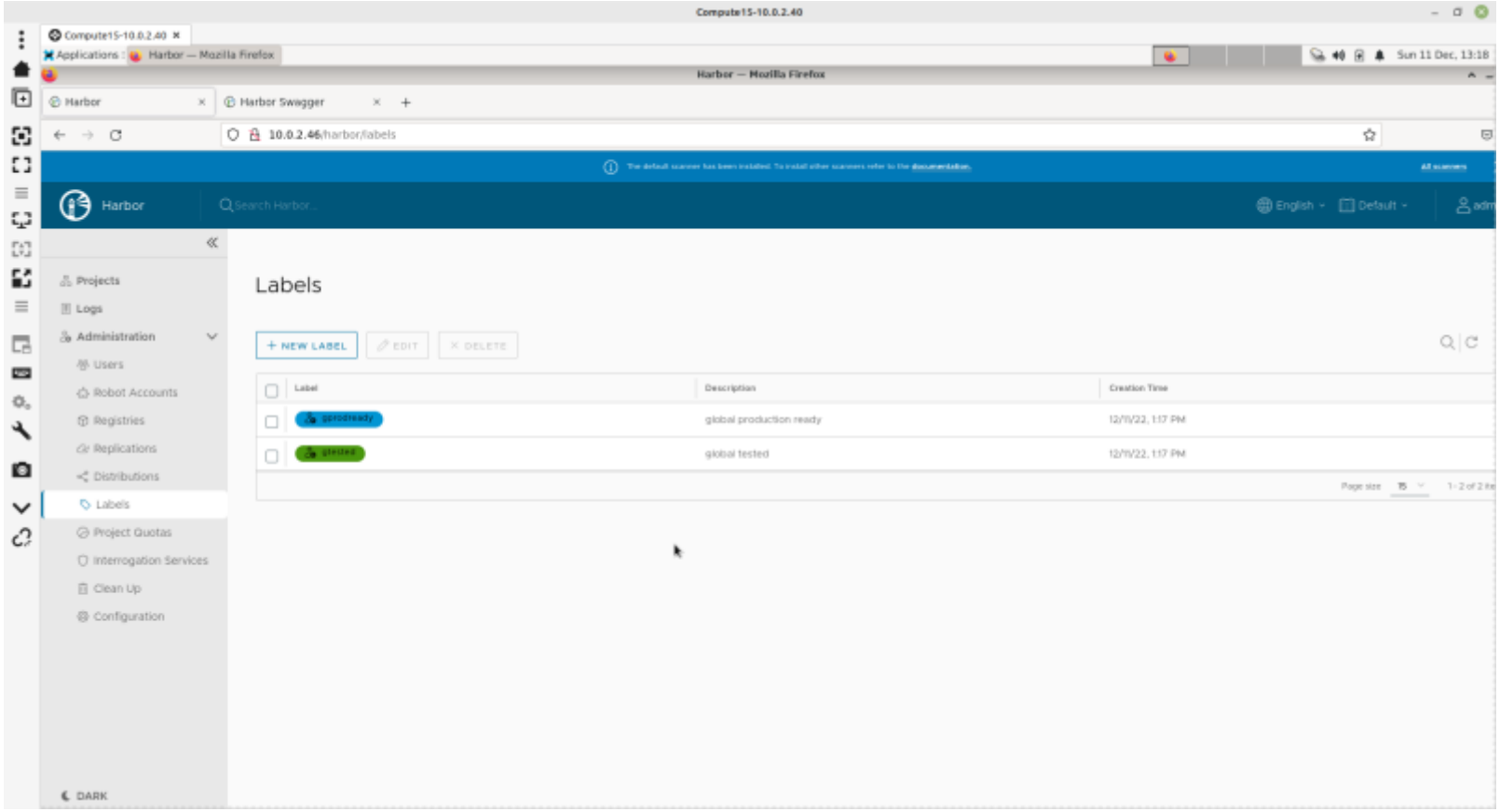
Harbor gère des **Labels** ou étiquettes. Ces étiquettes peuvent être globales ou spécifiques à un Projet. Quand une étiquette est spécifique, celle-ci est utilisée pour marquer les Artefacts pour une utilisation spécifique. Les étiguettes globales ne peuvent être gérées que par l'Administrateur de Harbor.

Cliquez sur **Labels** dans Harbor :

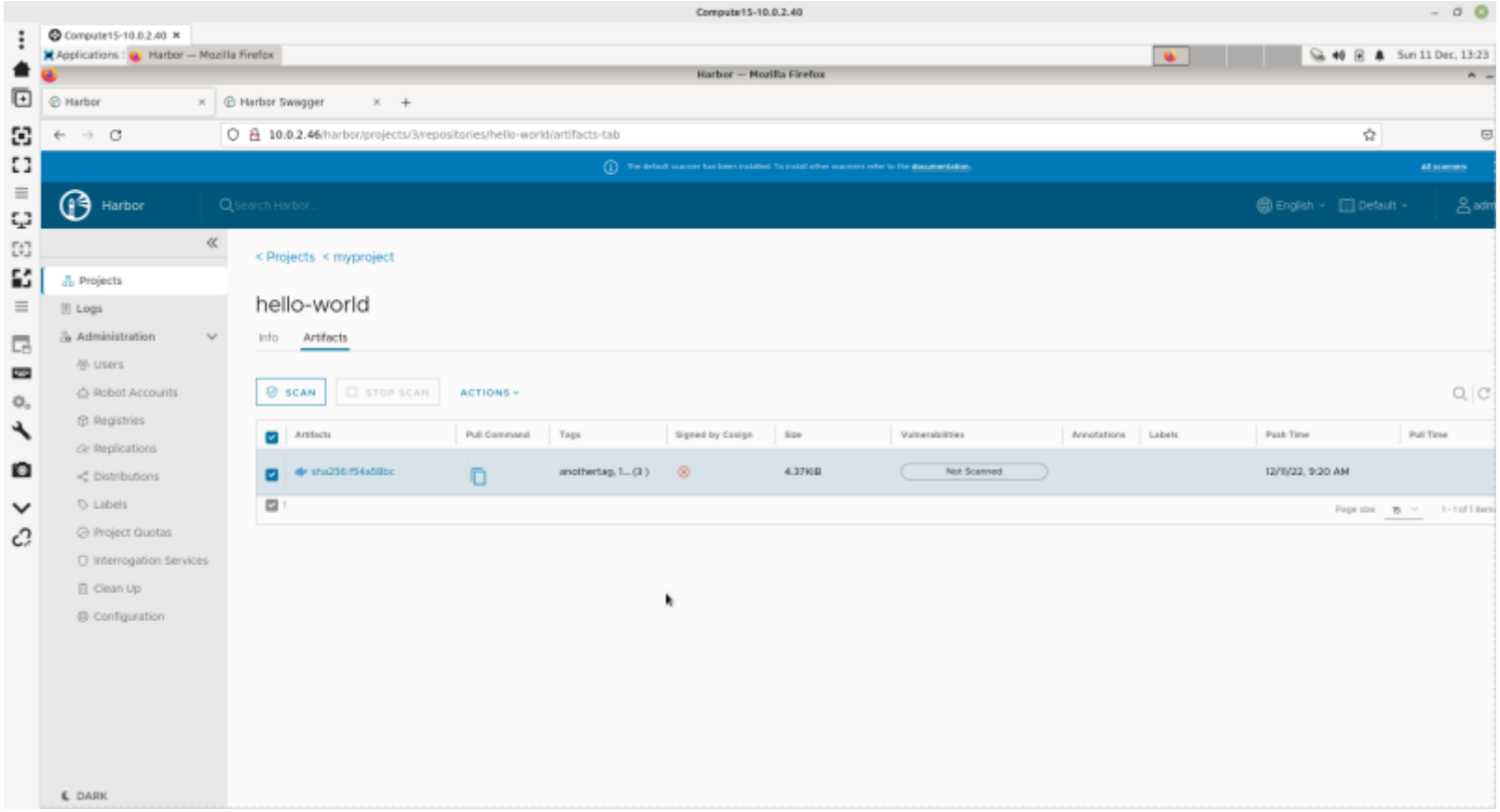


Cliquez sur **NEW LABEL** et créez les deux étiquettes suivantes :

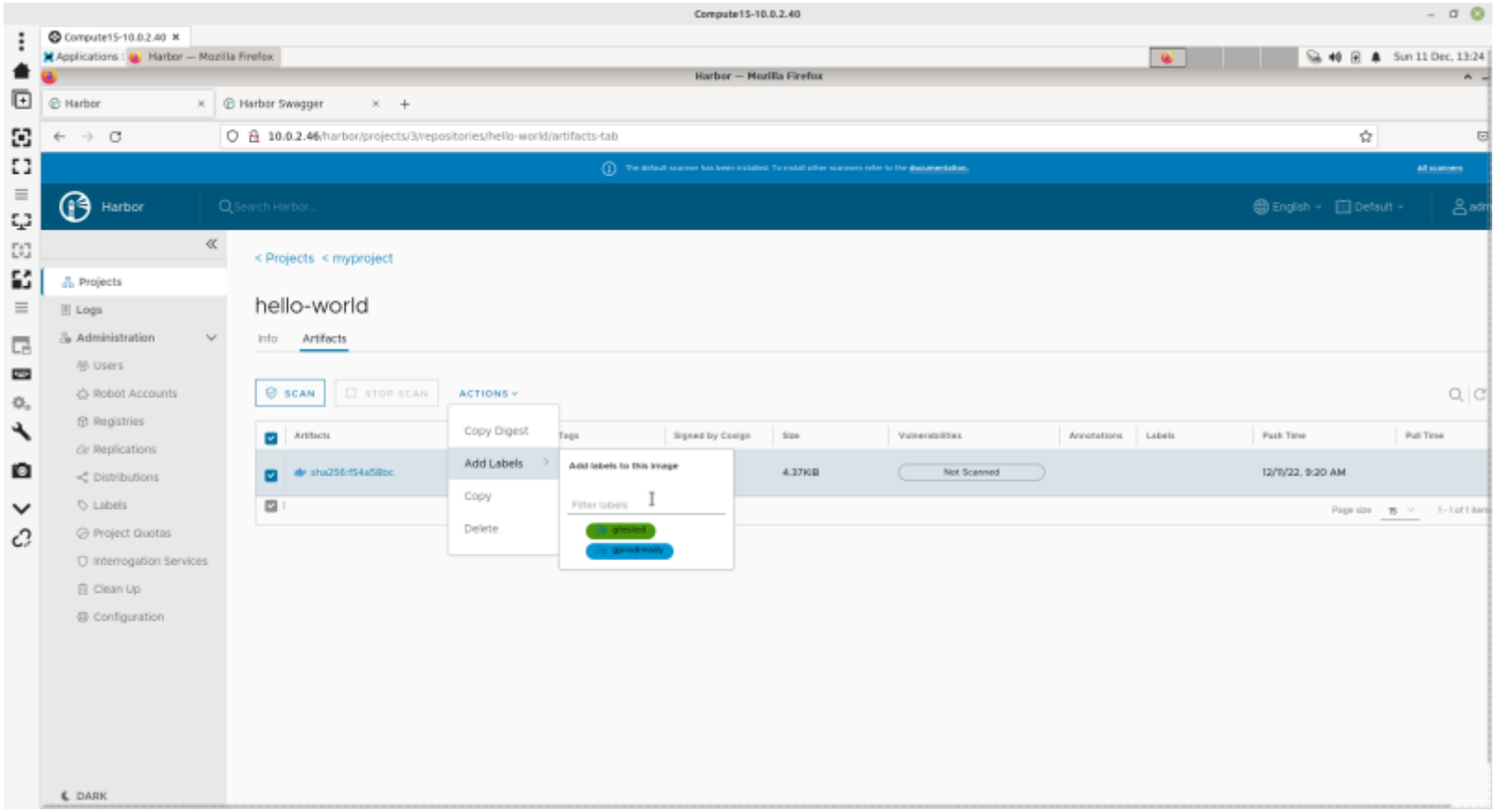
- **gtested** - global tested
- **gprodready** global production ready



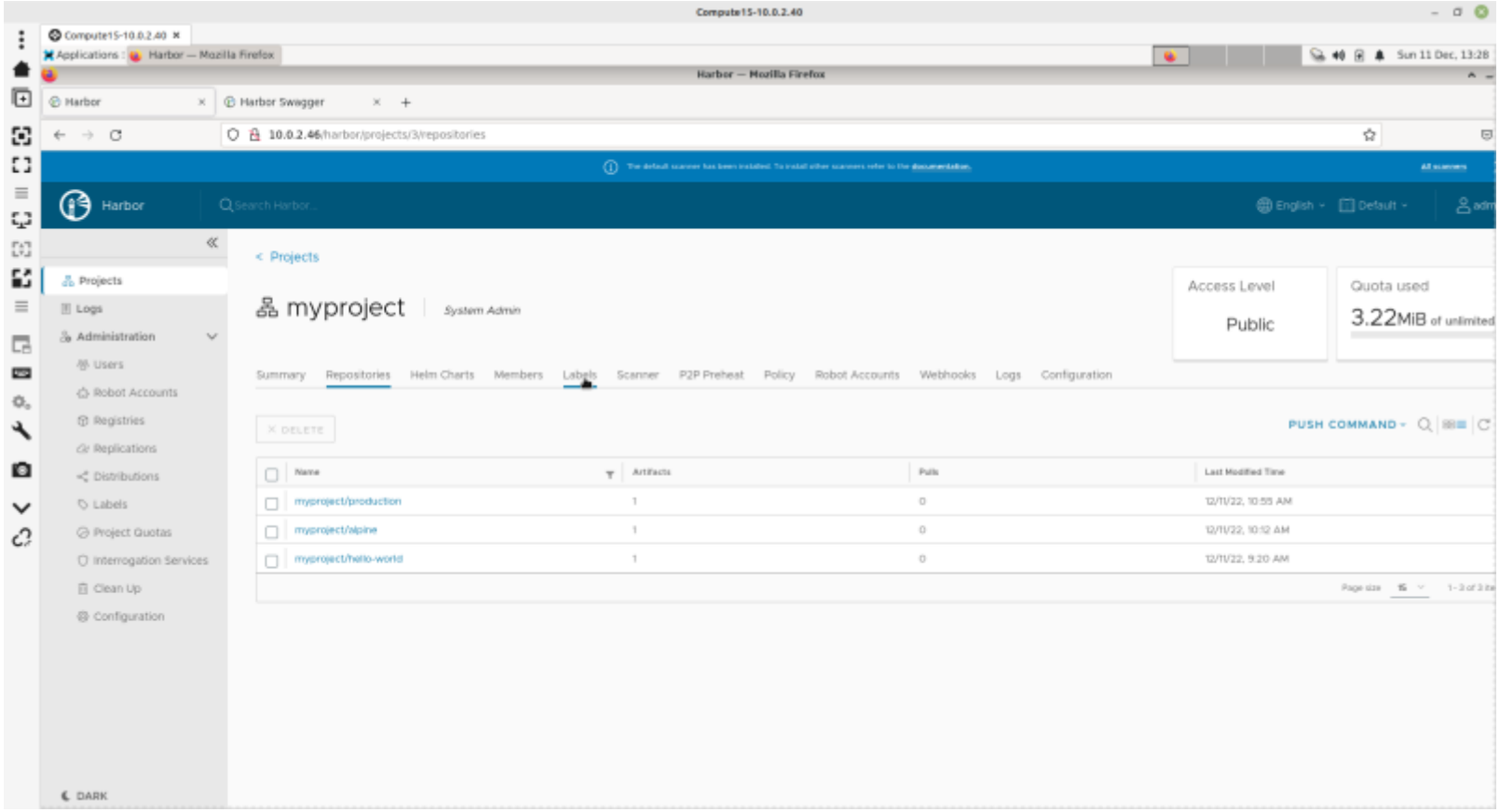
Cliquez sur **Projects > myproject > myproject/hello-world** puis sélectionnez l'Artefact :



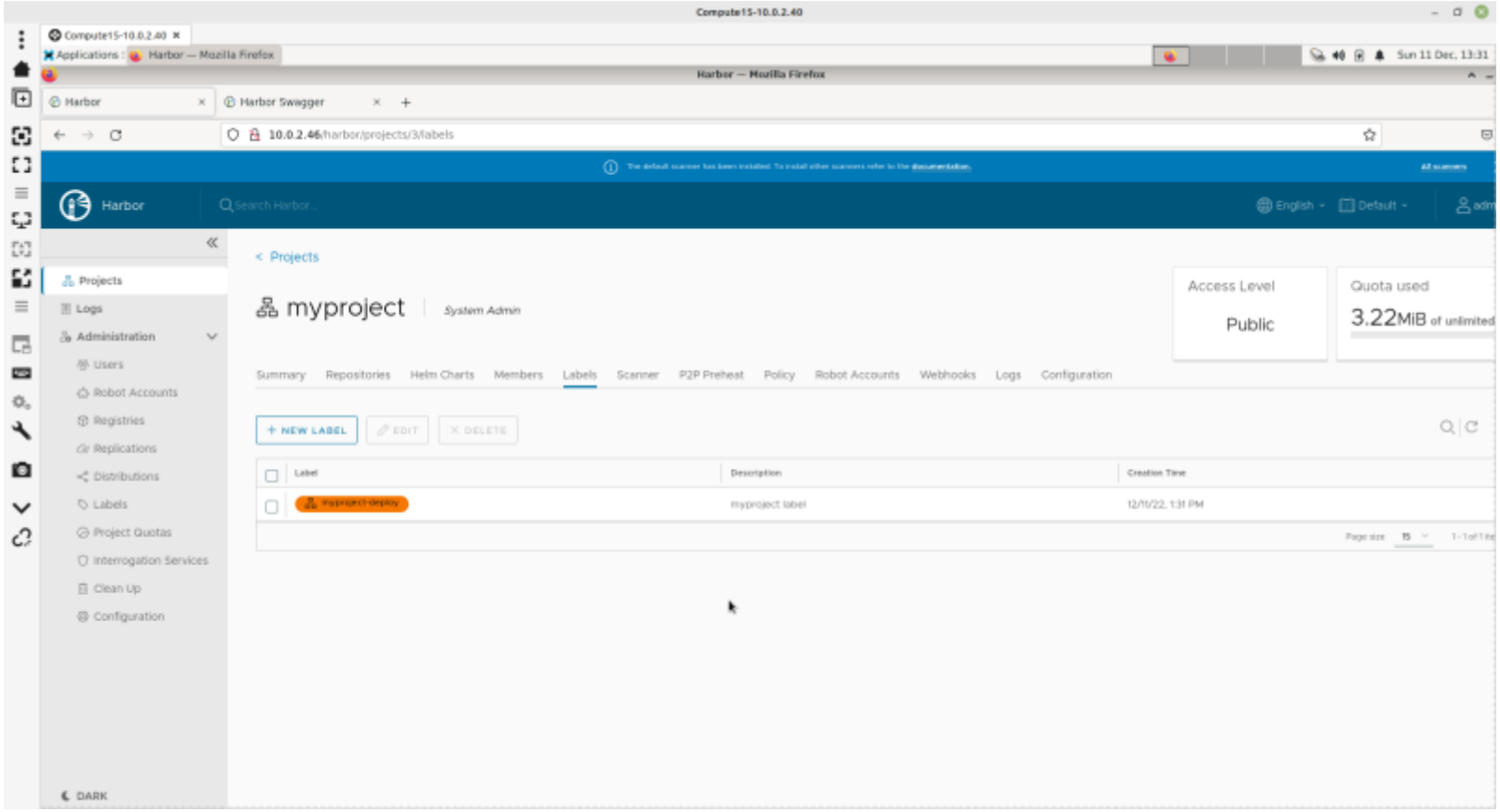
Cliquez sur **ACTIONS** > **Add Labels** puis sélectionnez **gtested** :



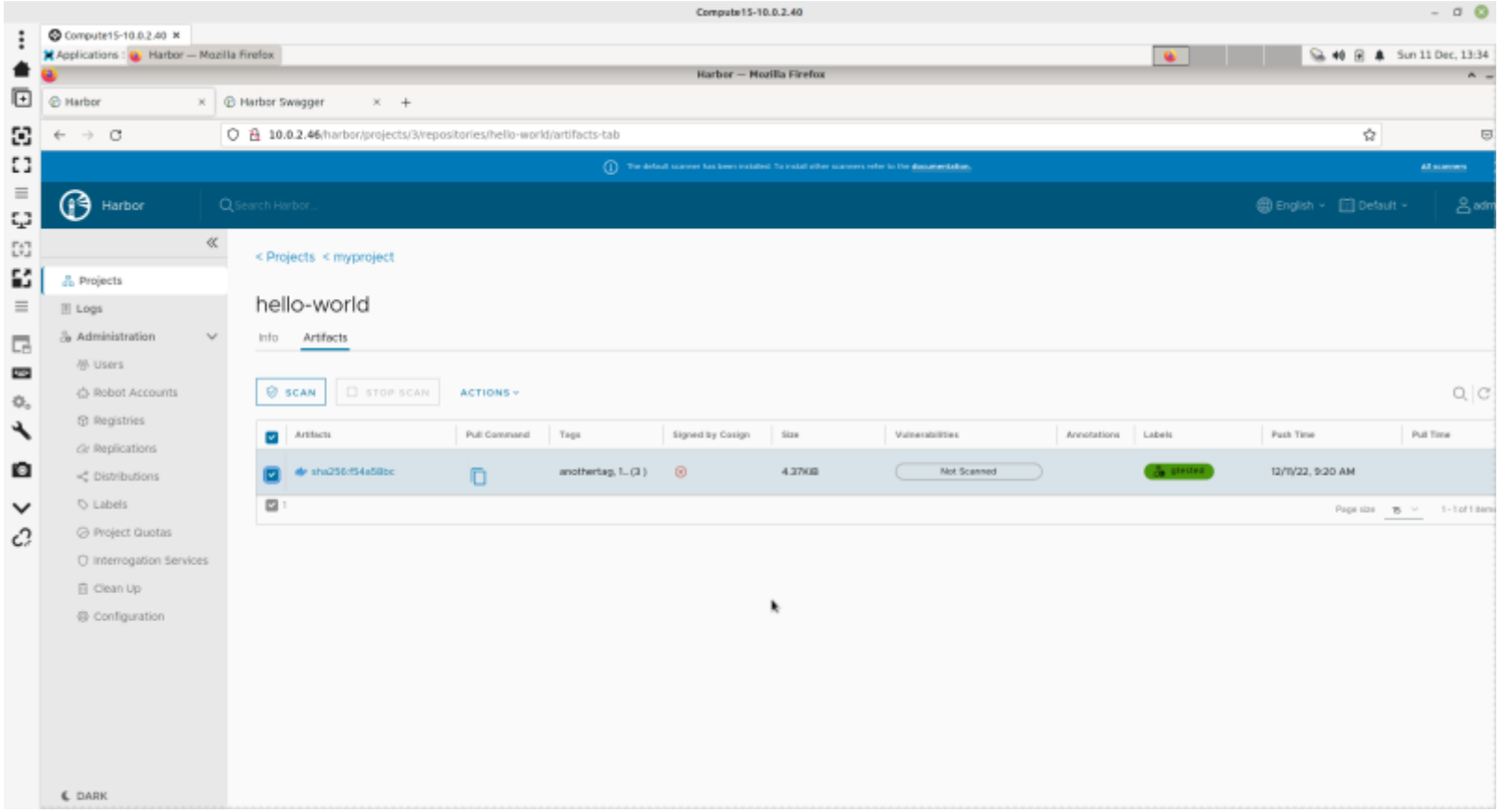
Cliquez maintenant sur **myproject > Labels** :



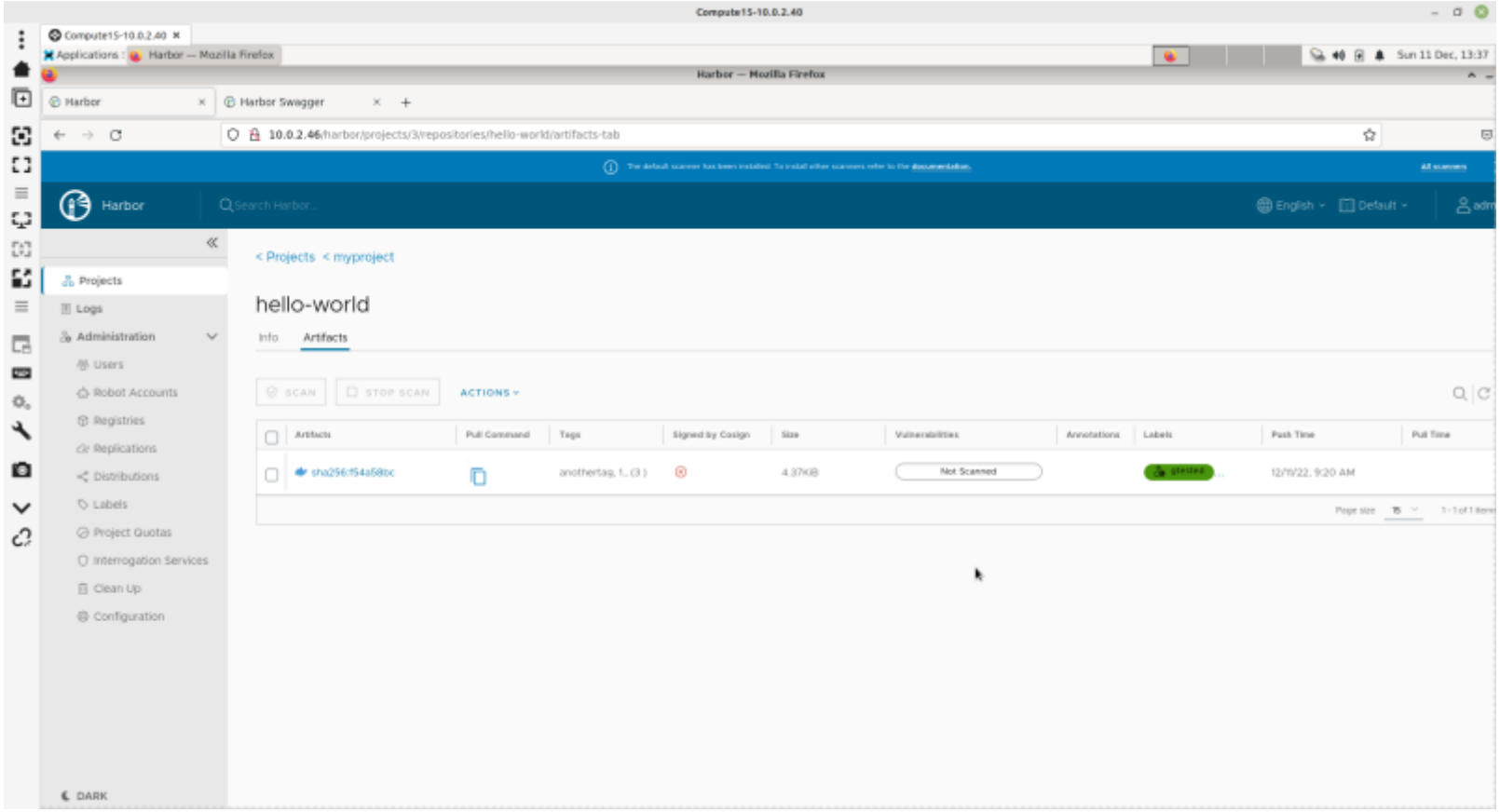
Cliquez ensuite sur **NEW LABEL** et créez ensuite une étiquette spécifique au projet appelé **myproject-deploy** ayant une description **myproject label** :



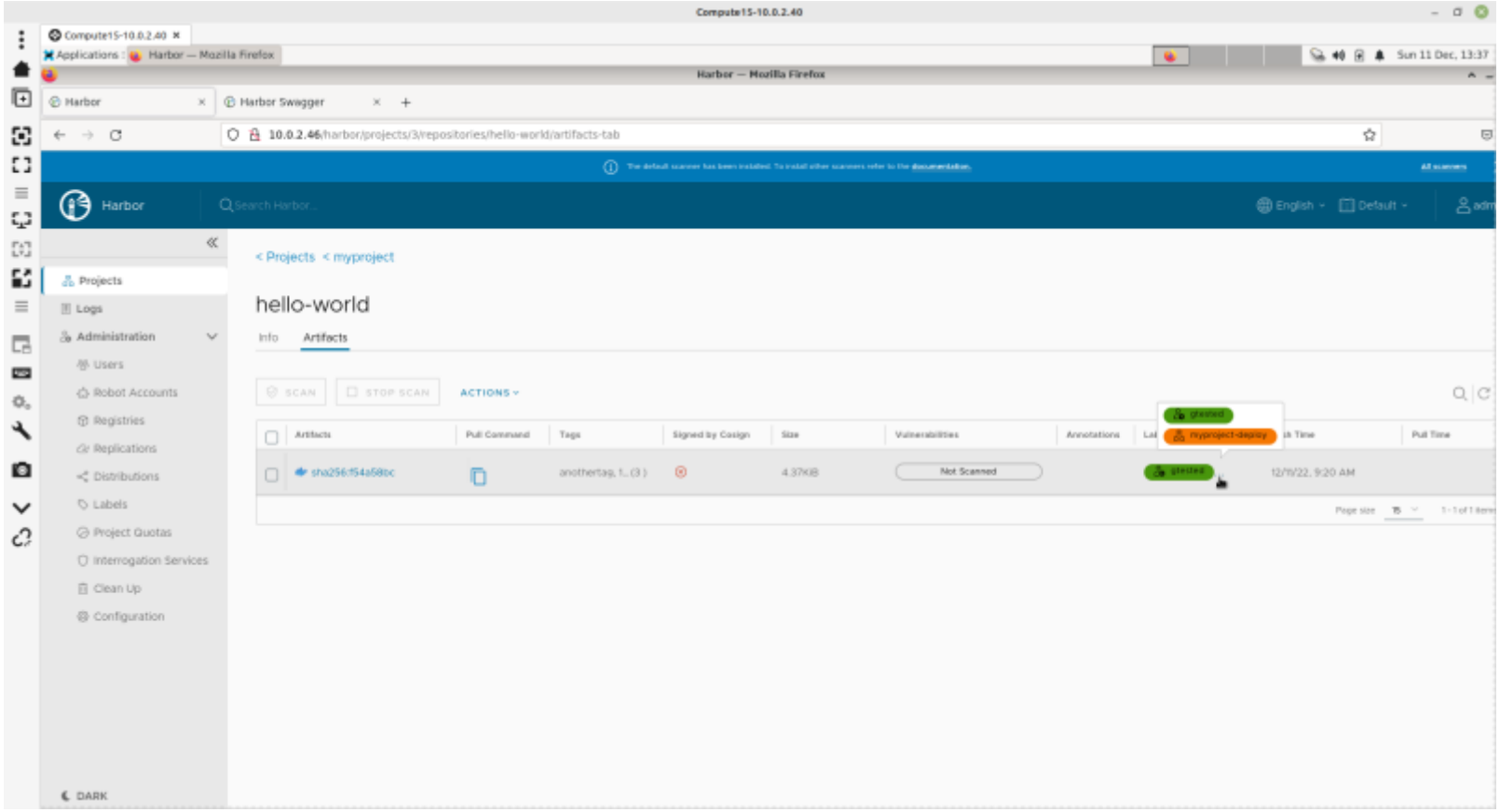
Cliquez sur **Repositories > myproject/hello-world** et sélectionnez l'Artefact :



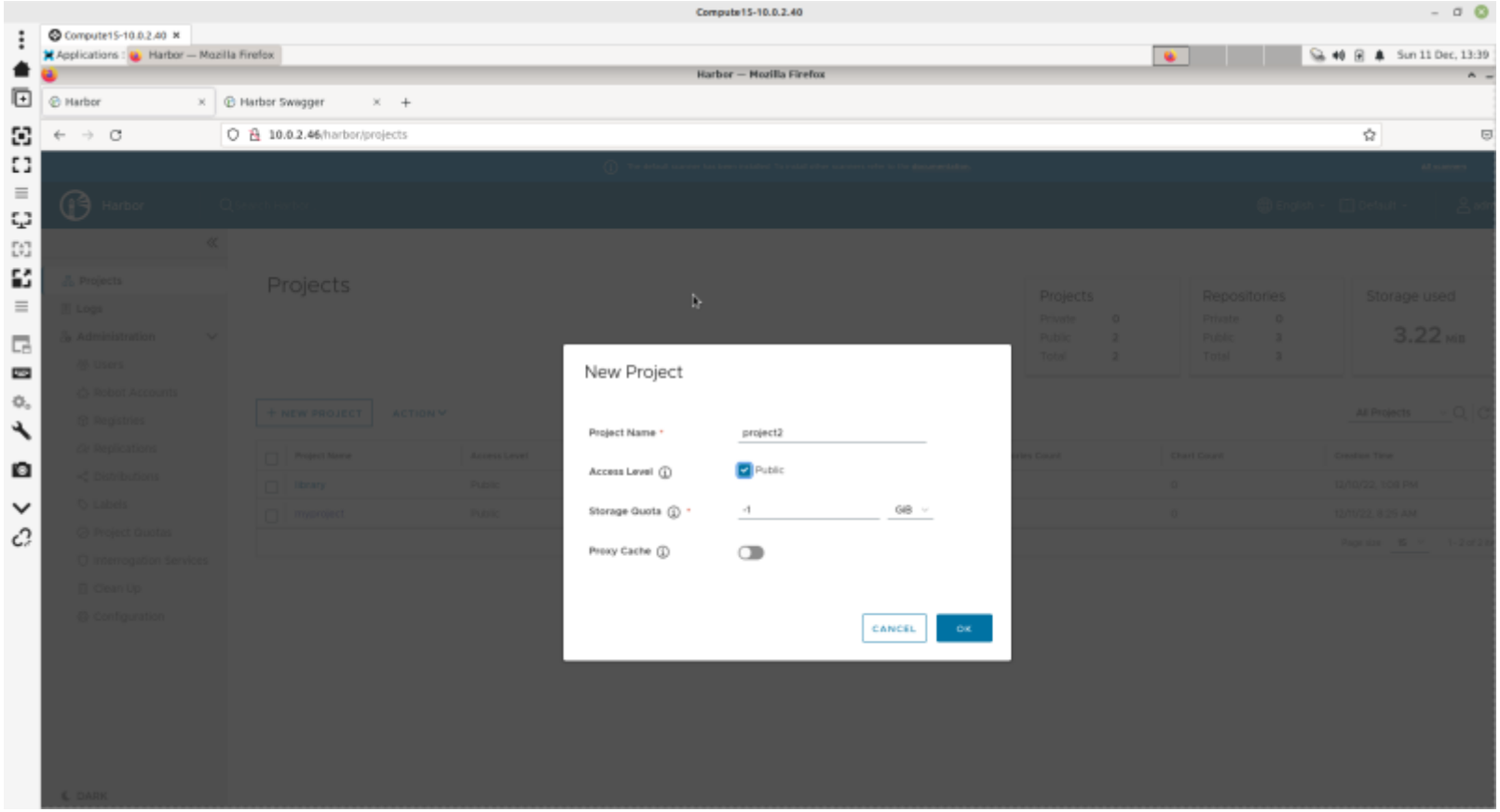
Cliquez sur **ACTIONS** > **Add Labels** et sélectionnez l'étiquette **myproject-deploy** :



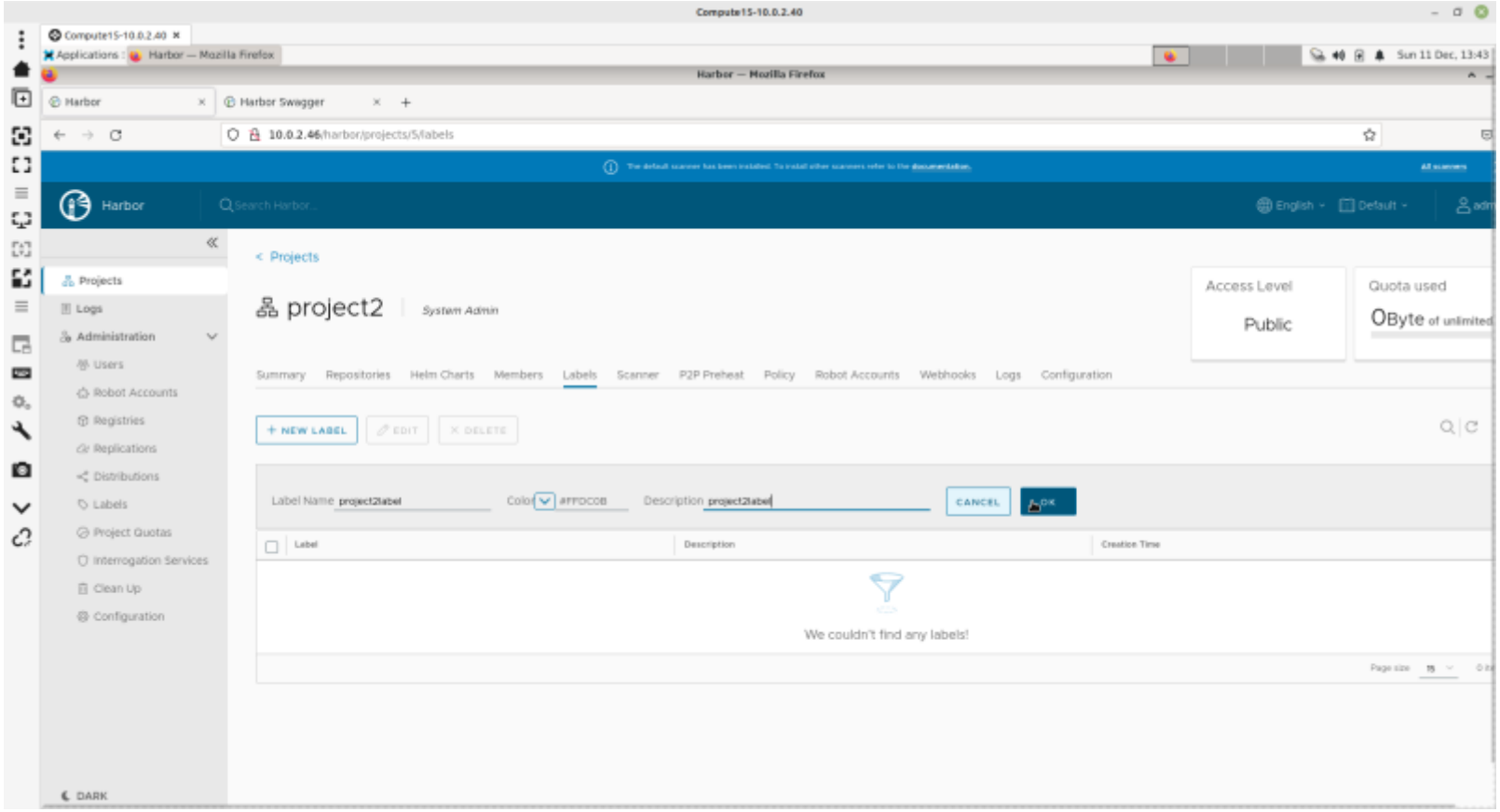
Pour visualiser toutes les étiguettes, cliquez sur les 3 points à droite de **gtested** :



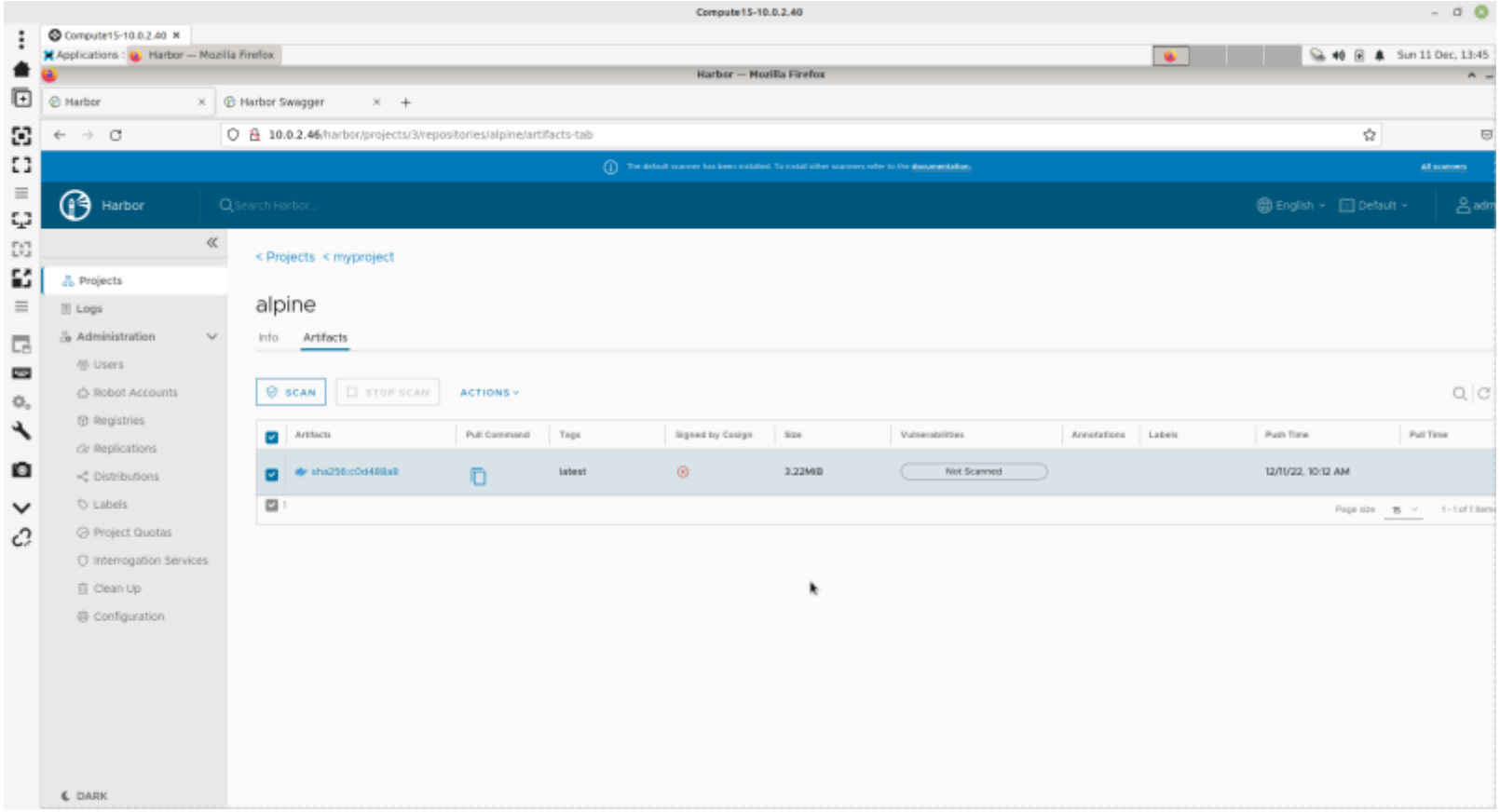
Cliquez sur **Projects > NEW PROJECT** et créez le projet **project2** :



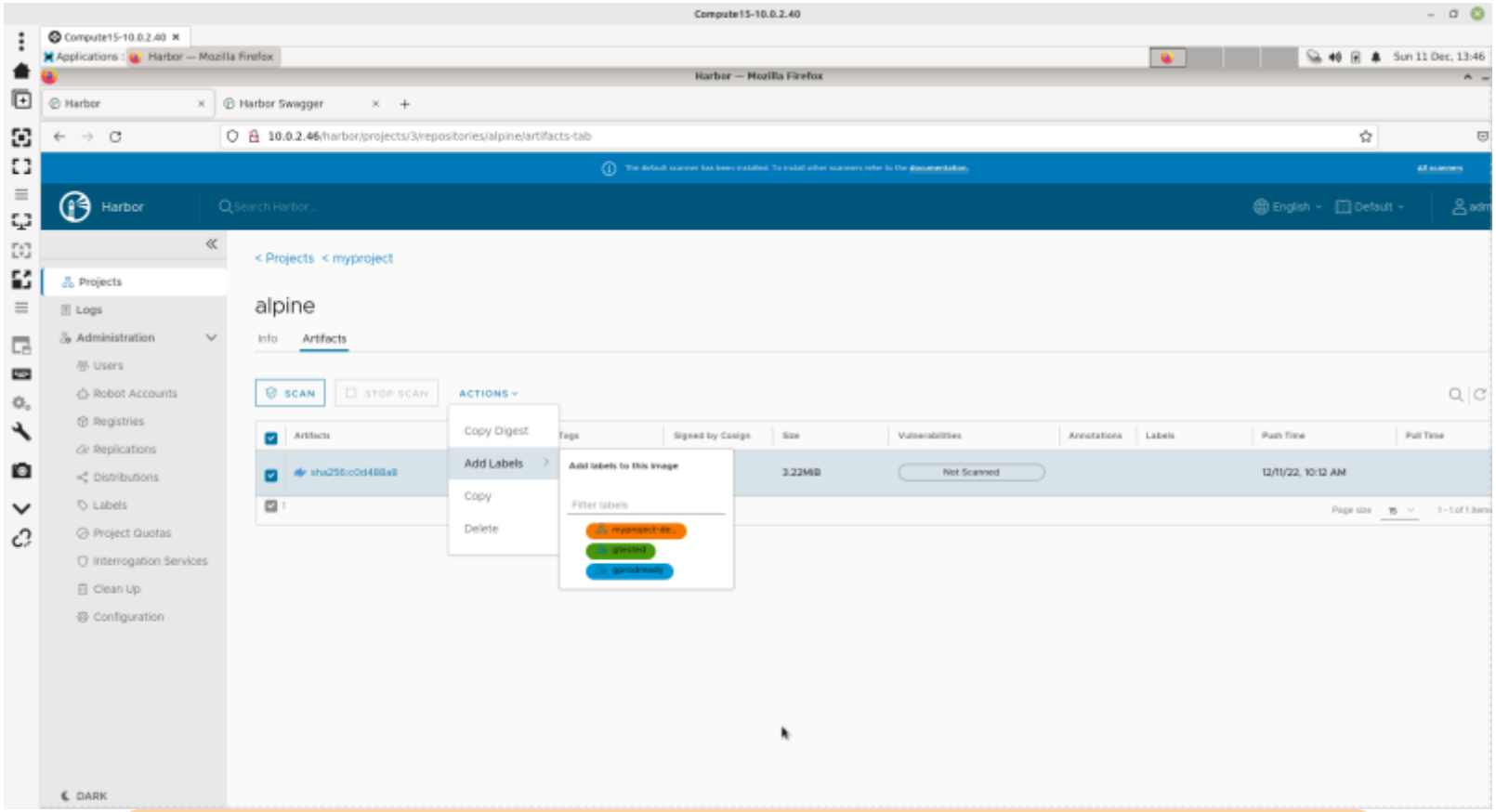
Cliquez sur **project2 > Labels > NEW LABEL** et créez l'étiquette **project2label** :



Cliquez sur **Projects > myproject > myproject/alpine** et sélectionnez l'Artefact :



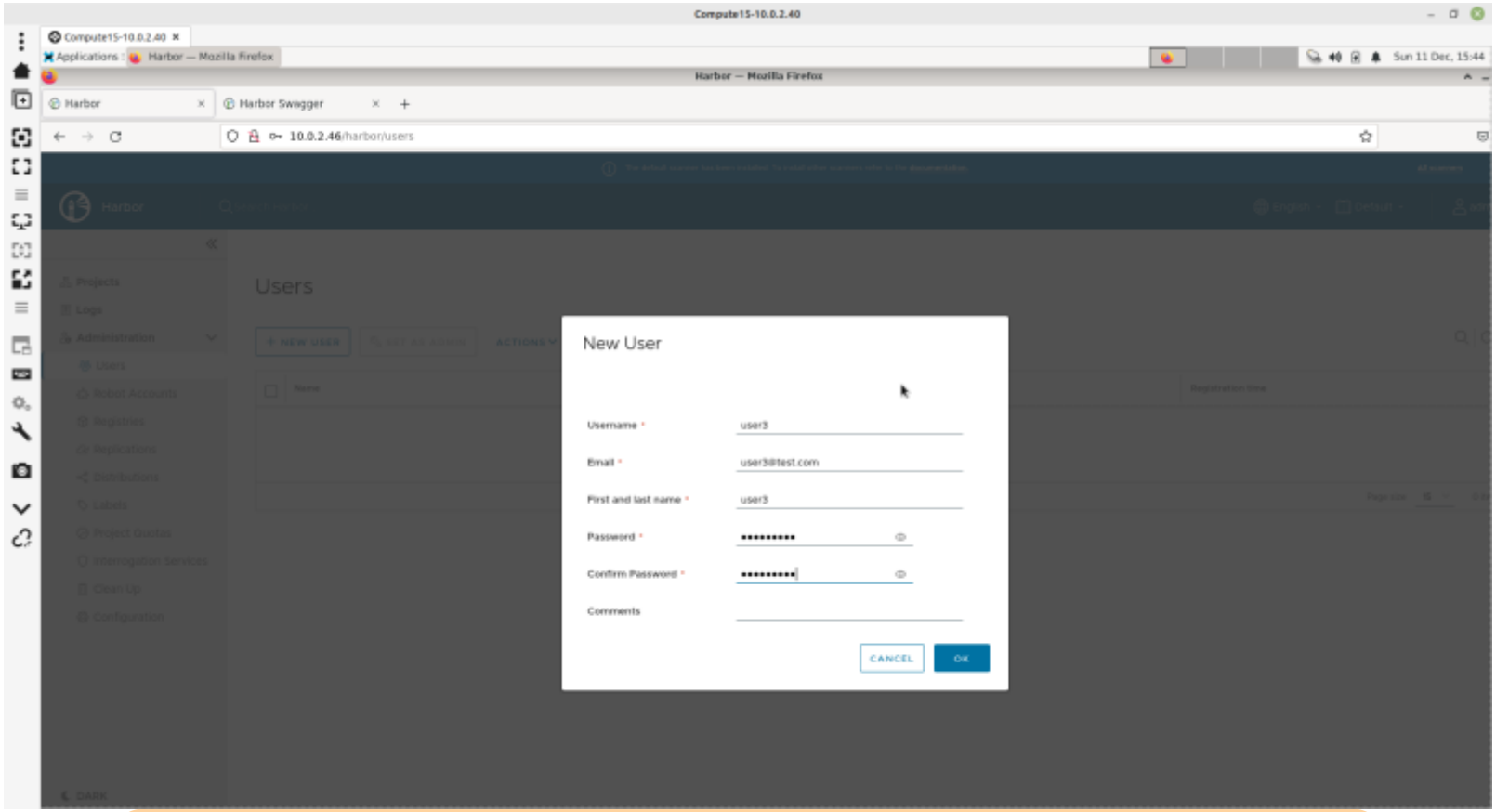
Cliquez sur **ACTIONS** > **Add Labels** :



Important : Notez que l'étiquette **project2label** n'est pas disponible.

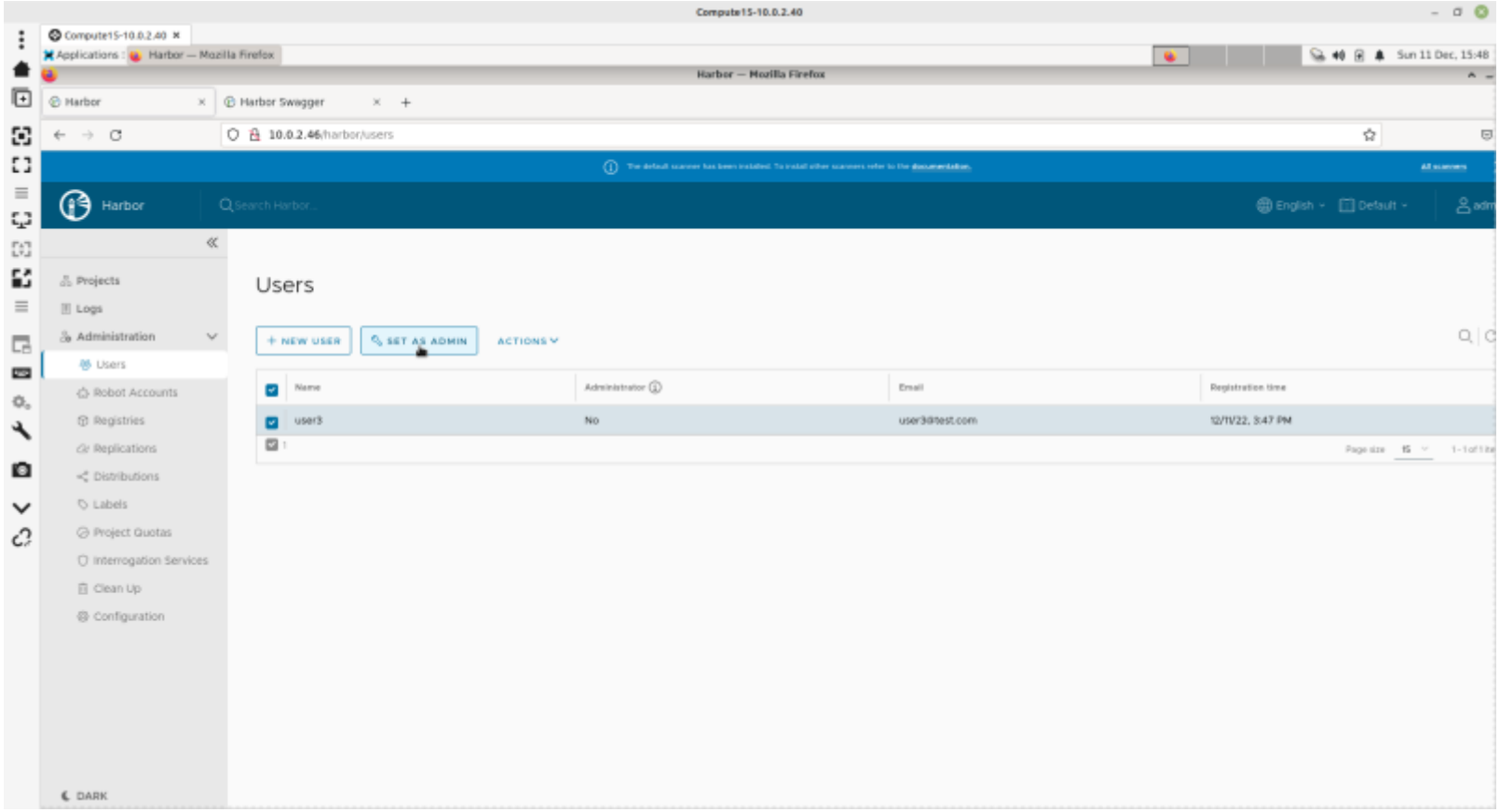
5.9 - Authentification et RBAC

Cliquez sur **Users** puis sur **NEW USER** et creéz l'utilisateur **user3** :

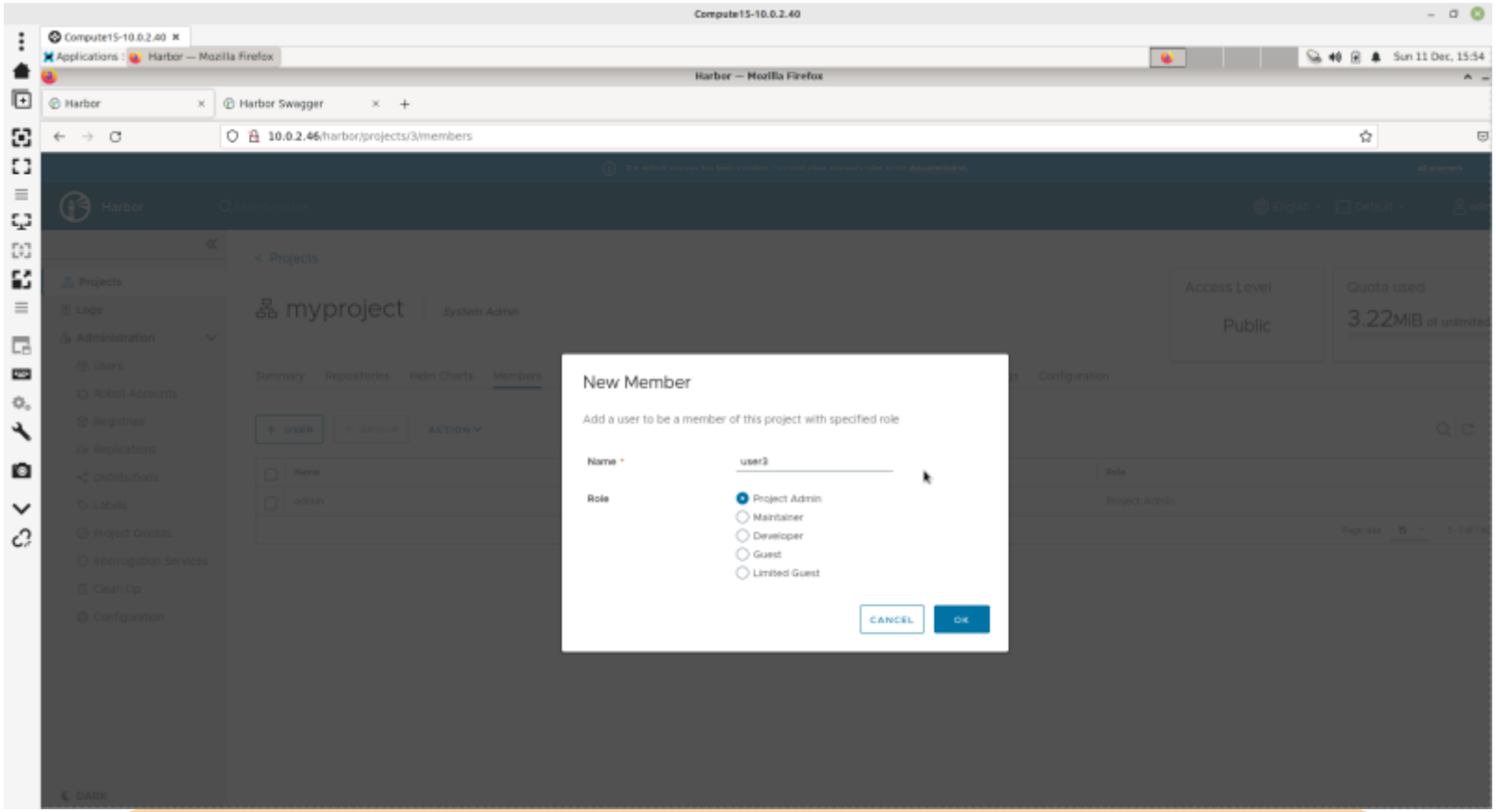


Important : Notez que pour être valide le mot de passe doit comporter une lettre Majuscule et un chiffre.

Sélectionnez l'utilisateur **user3** et cliquez sur le bouton **SET AS ADMIN** :

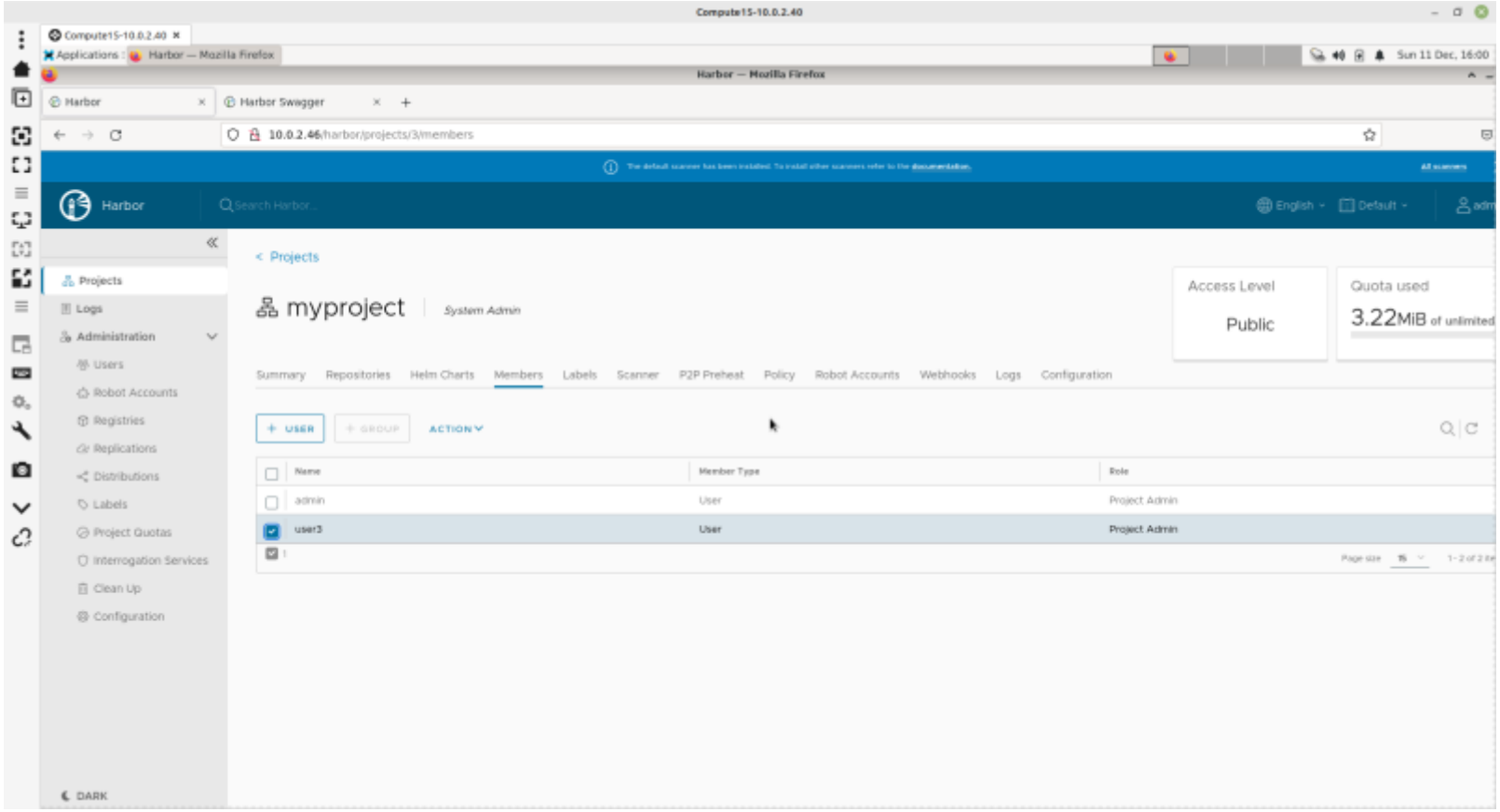


Cliquez sur **Projects > myproject** puis sur **Members** et **+ USER** :

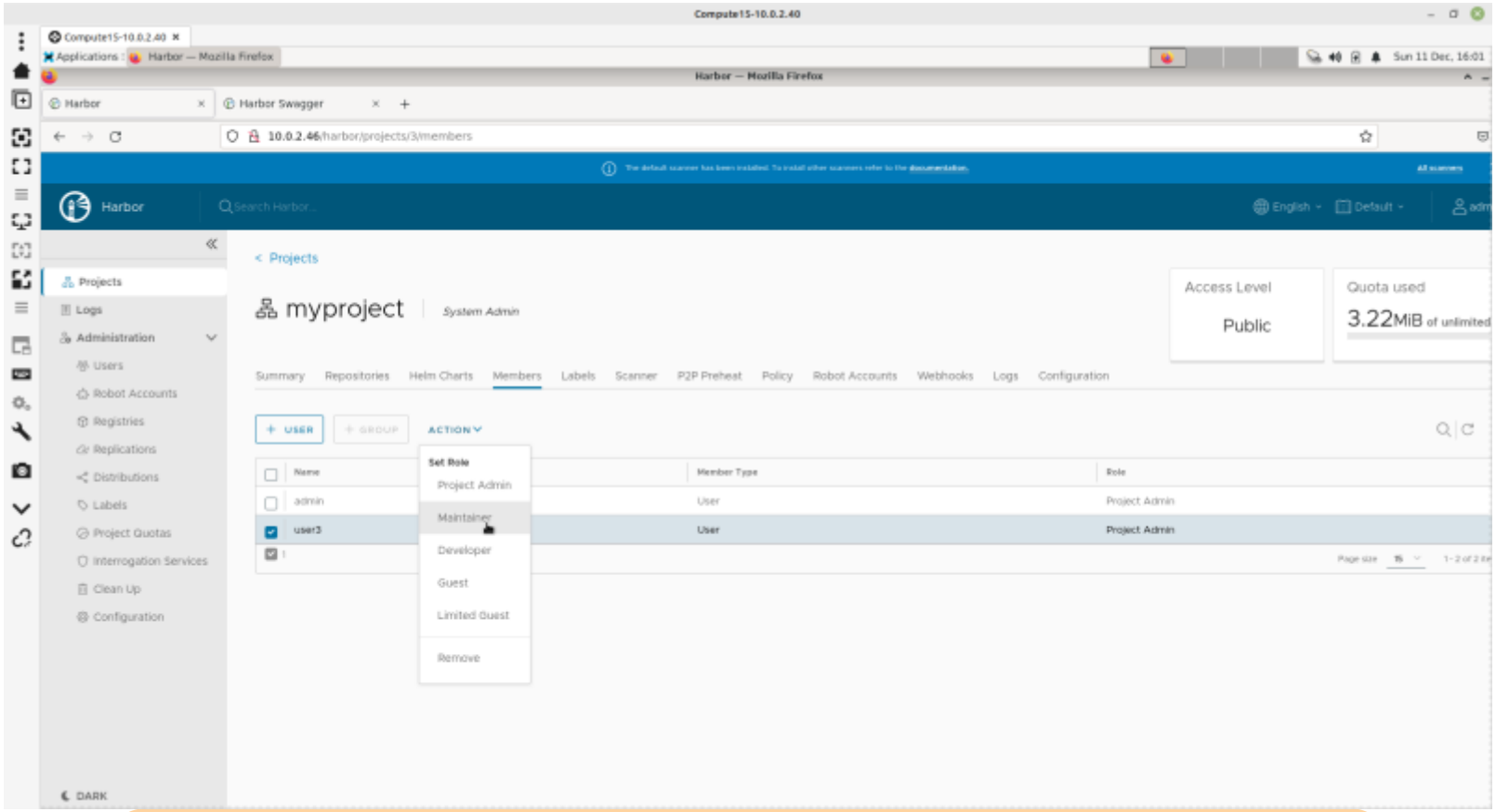


Important : Pour consulter les privilèges de chaque rôle, cliquez sur **ce lien**.

Cliquez sur **OK** et sélectionnez l'utilisateur **user3** :



Pour modifier son rôle, cliquez sur **ACTION** et choisissez le rôle désiré :



Important : Notez que le bouton **+ GROUP** est grisé. Si un Active Directory ou bien LDAP avait été configuré, Harbor pourrait lire la liste des groupes et n ous permettrait d'ajouter des utilisateurs à un des groupes.

5.10 - Utilisation d'Images Docker Personalisées

Créez un répertoire nommé myDocker :

```
root@debian11:~/harbor# cd ..
root@debian11:~# mkdir ~/myDocker
root@debian11:~# cd ~/myDocker
root@debian11:~/myDocker#
```

Créez le fichier myEntrypoint.sh :

```
root@debian11:~/myDocker# vi myEntrypoint.sh
root@debian11:~/myDocker# cat myEntrypoint.sh
#!/bin/bash
if [ -z "$myVariable" ]; then
        echo "La variable myVariable doit être renseignée"
        return 1
fi

while true;
do
        echo $1 \($(date +%H:%M:%S)\);
        sleep "$myVariable";
done
```

Testez ce script :

```
root@debian11:~/myDocker# myVariable=3 . ./myEntrypoint.sh salut
salut (16:12:59)
salut (16:13:02)
salut (16:13:05)
^C
root@debian11:~/myDocker#
```

Rendez ce script exécutable :

```
root@debian11:~/myDocker# chmod u+x myEntrypoint.sh
```

Créez maintenant le fichier **Dockerfile** dans le répertoire **~/myDocker** :

```
root@debian11:~/myDocker# cat Dockerfile
FROM centos:latest
MAINTAINER ittraining "infos@ittraining.io"
COPY myEntrypoint.sh /entrypoint.sh
ENV myVariable 3
ENTRYPOINT ["/entrypoint.sh"]
CMD ["mycommand"]
```

Connectez-vous à Harbor :

```
root@debian11:~/myDocker# docker login 10.0.2.46:80
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

Générez maintenant l'image :

```
root@debian11:~/myDocker# docker build --tag "10.0.2.46:80/myproject/mydocker:latest" .
Sending build context to Docker daemon  3.072kB
Step 1/6 : FROM centos:latest
latest: Pulling from library/centos
a1d0c7532777: Pull complete
Digest: sha256:a27fd8080b517143cbbbab9dfb7c8571c40d67d534bbdee55bd6c473f432b177
Status: Downloaded newer image for centos:latest
 ---> 5d0da3dc9764
Step 2/6 : MAINTAINER ittraining "infos@ittraining.io"
 ---> Running in 7e359420ea95
Removing intermediate container 7e359420ea95
 ---> 7ce8d150d72d
```

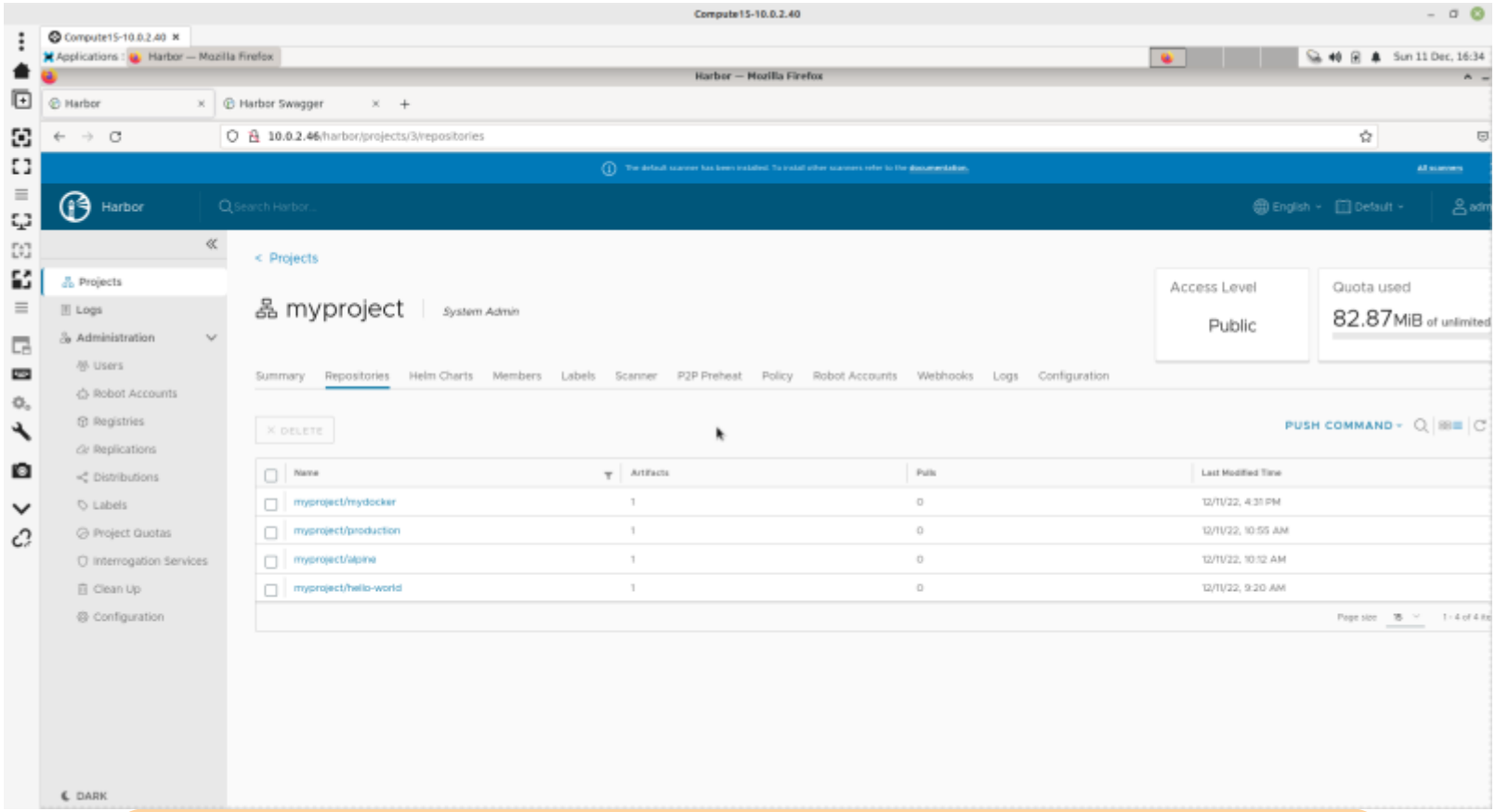
```
Step 3/6 : COPY myEntrypoint.sh /entrypoint.sh
 ---> c1fda8d1e43c
Step 4/6 : ENV myVariable 3
 ---> Running in edf989144c38
Removing intermediate container edf989144c38
 ---> 628615e7003a
Step 5/6 : ENTRYPOINT ["/entrypoint.sh"]
 ---> Running in 0490b6a50bbc
Removing intermediate container 0490b6a50bbc
 ---> c2ed329815ef
Step 6/6 : CMD ["mycommand"]
 ---> Running in e2ecfea5be53
Removing intermediate container e2ecfea5be53
 ---> ccb27534d7f2
Successfully built ccb27534d7f2
Successfully tagged 10.0.2.46:80/myproject/mydocker:latest

root@debian11:~/myDocker# docker images
REPOSITORY                            TAG       IMAGE ID       CREATED         SIZE
10.0.2.46:80/myproject/mydocker       latest    ccb27534d7f2   2 minutes ago   231MB
10.0.2.46:80/myproject/alpine         latest    49176f190c7e   2 weeks ago     7.05MB
alpine                                latest    49176f190c7e   2 weeks ago     7.05MB
goharbor/harbor-exporter              v2.6.2    ae230e335853   4 weeks ago     95.4MB
goharbor/chartmuseum-photon           v2.6.2    be361eee8a36   4 weeks ago     225MB
goharbor/redis-photon                 v2.6.2    84fa3669bb10   4 weeks ago     154MB
goharbor/trivy-adapter-photon         v2.6.2    beef2527c7b2   4 weeks ago     441MB
goharbor/notary-server-photon         v2.6.2    b43e7c963f06   4 weeks ago     112MB
goharbor/notary-signer-photon         v2.6.2    4e9982286316   4 weeks ago     109MB
goharbor/harbor-registryctl           v2.6.2    d403d4e0f990   4 weeks ago     137MB
goharbor/registry-photon              v2.6.2    596e445b18f0   4 weeks ago     77.5MB
goharbor/nginx-photon                 v2.6.2    cbbdf28947a5   4 weeks ago     154MB
goharbor/harbor-log                   v2.6.2    3ce902023f73   4 weeks ago     161MB
goharbor/harbor-jobservice            v2.6.2    05791a75c8b8   4 weeks ago     250MB
goharbor/harbor-core                  v2.6.2    484e2a64616d   4 weeks ago     213MB
```

```
goharbor/harbor-portal                v2.6.2    932297030d4e   4 weeks ago     162MB
goharbor/harbor-db                    v2.6.2    351b19e366cb   4 weeks ago     225MB
goharbor/prepare                      v2.6.2    eecc993c7643   4 weeks ago     164MB
10.0.2.46:80/myproject/production     latest    feb5d9fea6a5   14 months ago   13.3kB
hello-world                           latest    feb5d9fea6a5   14 months ago   13.3kB
10.0.2.46:80/myproject/hello-world    1.0       feb5d9fea6a5   14 months ago   13.3kB
10.0.2.46:80/myproject/hello-world    latest    feb5d9fea6a5   14 months ago   13.3kB
centos                                latest    5d0da3dc9764   15 months ago   231MB
```

Procédez maintenant au **push** :

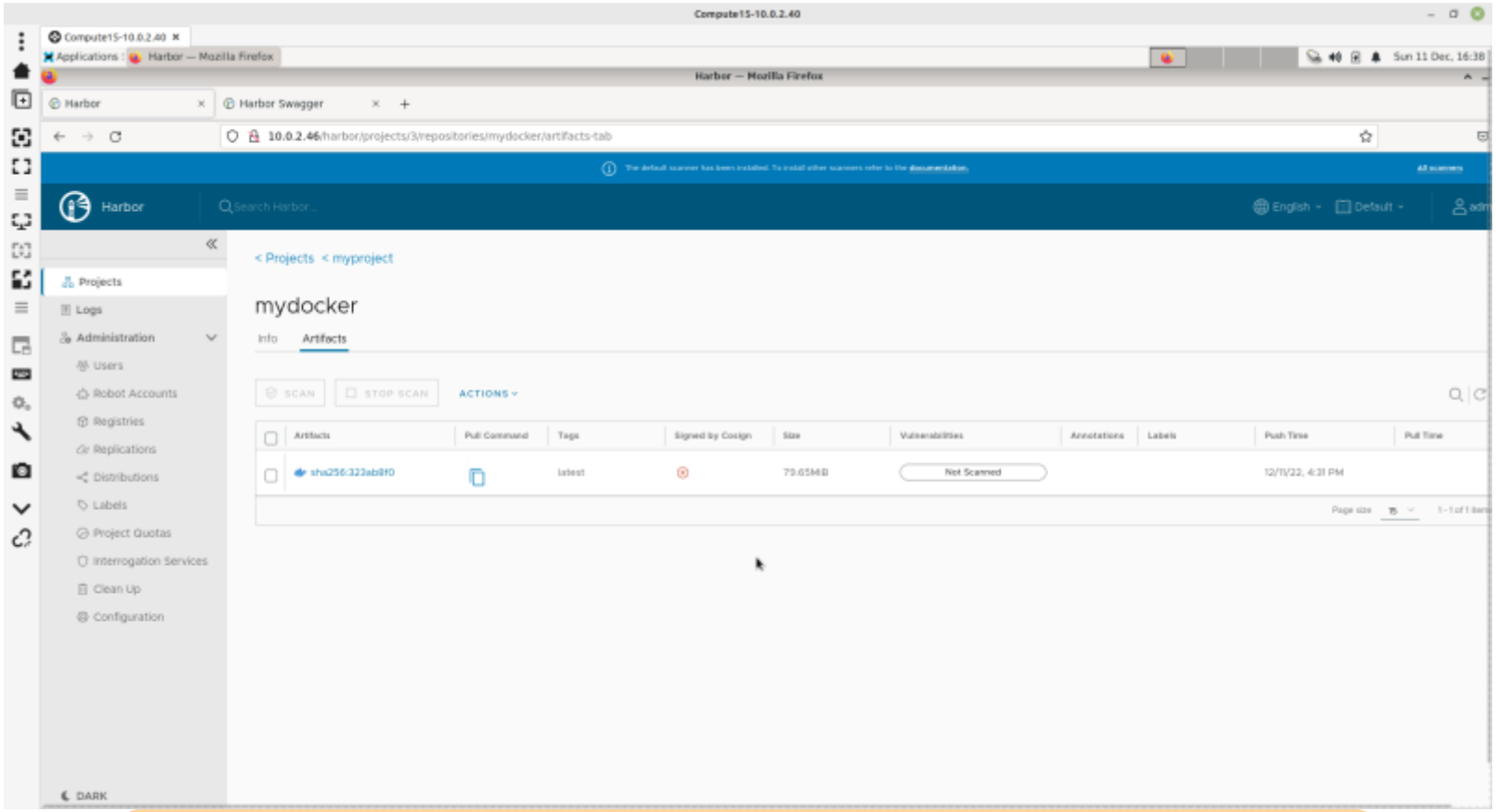
```
root@debian11:~/myDocker# docker push 10.0.2.46:80/myproject/mydocker:latest
The push refers to repository [10.0.2.46:80/myproject/mydocker]
7c7558b01004: Pushed
74ddd0ec08fa: Pushed
latest: digest: sha256:323ab8f05733ac369963b338bb8fe81bd5cb2d77b46ac3650b7924e7da3aa9cf size: 736
```

Retournez à l'interface de Harbor et cliquez sur **Projects > myproject** :



Important : Notez que le dépôt **myproject/mydocker** a été créé.

Cliquez sur **myproject/mydocker** :



Important : Notez que l'Artefact est présent et qu'il possède un tag **latest**.

Refaites un build de l'image en changant le tag :

```
root@debian11:~/myDocker# docker build --tag "10.0.2.46:80/myproject/mydocker:rel1" .
Sending build context to Docker daemon  3.072kB
Step 1/6 : FROM centos:latest
 ---> 5d0da3dc9764
Step 2/6 : MAINTAINER ittraining "infos@ittraining.io"
```

```
 ---> Using cache
 ---> 7ce8d150d72d
Step 3/6 : COPY myEntrypoint.sh /entrypoint.sh
 ---> Using cache
 ---> c1fda8d1e43c
Step 4/6 : ENV myVariable 3
 ---> Using cache
 ---> 628615e7003a
Step 5/6 : ENTRYPOINT ["/entrypoint.sh"]
 ---> Using cache
 ---> c2ed329815ef
Step 6/6 : CMD ["mycommand"]
 ---> Using cache
 ---> ccb27534d7f2
Successfully built ccb27534d7f2
Successfully tagged 10.0.2.46:80/myproject/mydocker:rel1
```

Important : Notez l'utilisation des images intermédiaires issues du cache.

Procédez maintenant au **push** :

```
root@debian11:~/myDocker# docker push 10.0.2.46:80/myproject/mydocker:rel1
The push refers to repository [10.0.2.46:80/myproject/mydocker]
7c7558b01004: Layer already exists
74ddd0ec08fa: Layer already exists
rel1: digest: sha256:323ab8f05733ac369963b338bb8fe81bd5cb2d77b46ac3650b7924e7da3aa9cf size: 736
```

Important : Notez que seul le nouveau tag a été poussé.

Editez le Dockerfile et changez la valeur de **myVariable** de 3 à **5** :

```
root@debian11:~/myDocker# vi Dockerfile
root@debian11:~/myDocker# cat Dockerfile
FROM centos:latest
MAINTAINER ittraining "infos@ittraining.io"
COPY myEntrypoint.sh /entrypoint.sh
ENV myVariable 5
ENTRYPOINT ["/entrypoint.sh"]
CMD ["mycommand"]
```

Compilez de nouveau l'image :

```
root@debian11:~/myDocker# docker build --tag "10.0.2.46:80/myproject/mydocker:rel2" .
Sending build context to Docker daemon  3.072kB
Step 1/6 : FROM centos:latest
 ---> 5d0da3dc9764
Step 2/6 : MAINTAINER ittraining "infos@ittraining.io"
 ---> Using cache
 ---> 7ce8d150d72d
Step 3/6 : COPY myEntrypoint.sh /entrypoint.sh
 ---> Using cache
 ---> c1fda8d1e43c
Step 4/6 : ENV myVariable 5
 ---> Running in c90f60beee6f
Removing intermediate container c90f60beee6f
 ---> d9a704d55c69
Step 5/6 : ENTRYPOINT ["/entrypoint.sh"]
 ---> Running in 8aabb88b172d
Removing intermediate container 8aabb88b172d
 ---> 6ca7239b3489
Step 6/6 : CMD ["mycommand"]
 ---> Running in 8e1bd939a7c2
Removing intermediate container 8e1bd939a7c2
 ---> 9a12b543b94b
Successfully built 9a12b543b94b
```

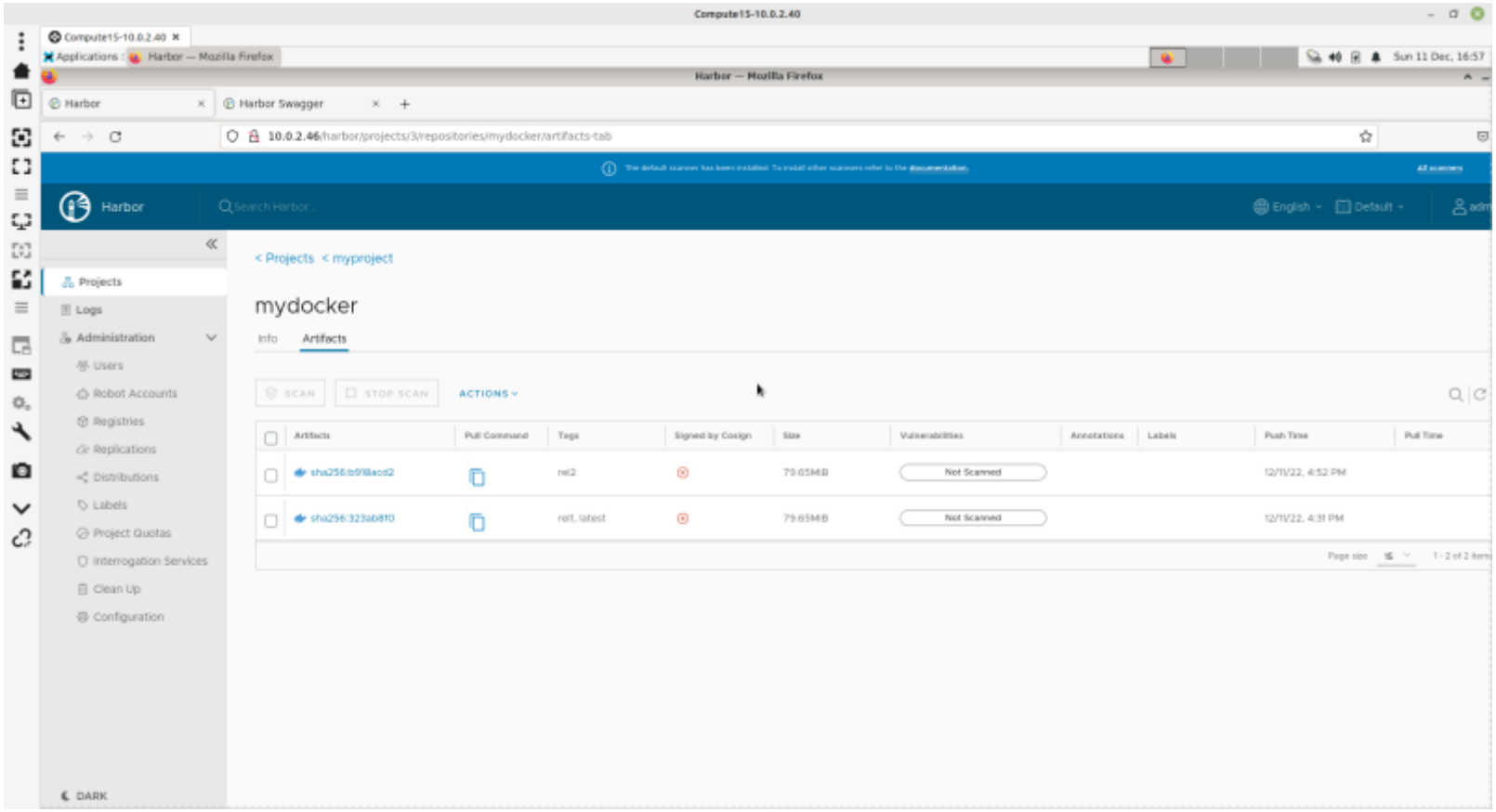
```
Successfully tagged 10.0.2.46:80/myproject/mydocker:rel2
```

Procédez maintenant au **push** :

```
root@debian11:~/myDocker# docker push 10.0.2.46:80/myproject/mydocker:rel2
The push refers to repository [10.0.2.46:80/myproject/mydocker]
7c7558b01004: Layer already exists
74ddd0ec08fa: Layer already exists
rel2: digest: sha256:b918acd244c70cf528d8e96584d8d34099d2a8eefc1edd86ba972dd975934f52 size: 736
```

Important : Notez que la valeur du SHA256 de rel2 est différente - **b918acd244c70cf528d8e96584d8d34099d2a8eefc1edd86ba972dd975934f52** par rapport à **323ab8f05733ac369963b338bb8fe81bd5cb2d77b46ac3650b7924e7da3aa9cf**.

De ce fait un nouvel Artefact a été créé :



Editez le Dockerfile et changez la valeur de **CMD** de “mycommand” à **“mycommand modified”** :

```
root@debian11:~/myDocker# cat Dockerfile
FROM centos:latest
MAINTAINER ittraining "infos@ittraining.io"
COPY myEntrypoint.sh /entrypoint.sh
ENV myVariable 5
ENTRYPOINT ["/entrypoint.sh"]
CMD ["mycommand modified"]
```

Compilez de nouveau l'image en utilisant le même tag qu'avant :

```
root@debian11:~/myDocker# docker build --tag "10.0.2.46:80/myproject/mydocker:rel2" .
Sending build context to Docker daemon  3.072kB
Step 1/6 : FROM centos:latest
 ---> 5d0da3dc9764
Step 2/6 : MAINTAINER ittraining "infos@ittraining.io"
 ---> Using cache
 ---> 7ce8d150d72d
Step 3/6 : COPY myEntrypoint.sh /entrypoint.sh
 ---> Using cache
 ---> c1fda8d1e43c
Step 4/6 : ENV myVariable 5
 ---> Using cache
 ---> d9a704d55c69
Step 5/6 : ENTRYPOINT ["/entrypoint.sh"]
 ---> Using cache
 ---> 6ca7239b3489
Step 6/6 : CMD ["mycommand modified"]
 ---> Running in ac174f91145b
Removing intermediate container ac174f91145b
 ---> 56cc6362dc99
Successfully built 56cc6362dc99
Successfully tagged 10.0.2.46:80/myproject/mydocker:rel2
```

Procédez maintenant au **push** :

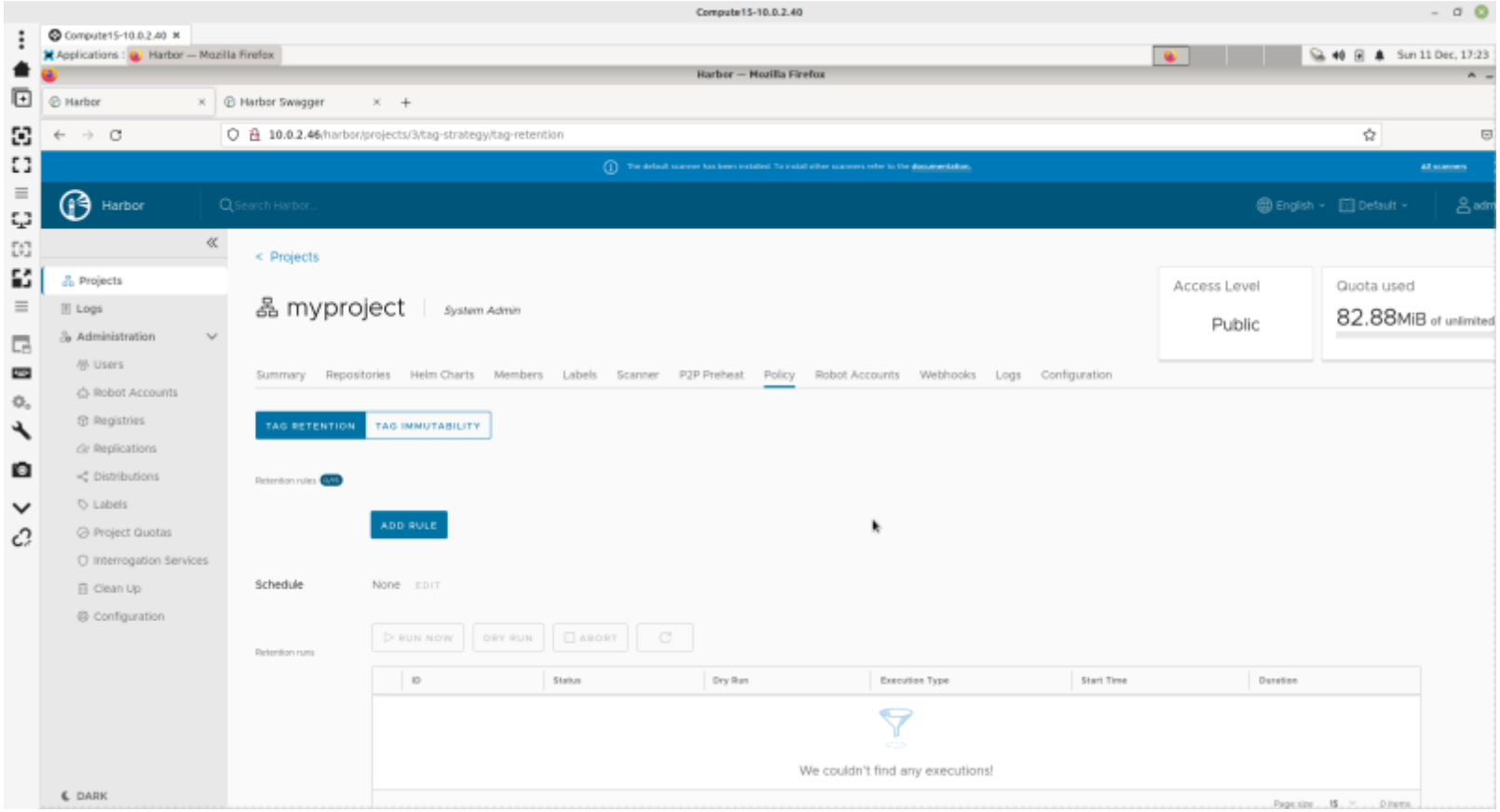
```
root@debian11:~/myDocker# docker push 10.0.2.46:80/myproject/mydocker:rel2
The push refers to repository [10.0.2.46:80/myproject/mydocker]
7c7558b01004: Layer already exists
74ddd0ec08fa: Layer already exists
rel2: digest: sha256:fcf710f98e14ac3df4d3cf365f04423da8ceb42325eca9efe87f67082904728e size: 736
```

Important : Notez que la valeur du SHA256 de rel2 est différente -

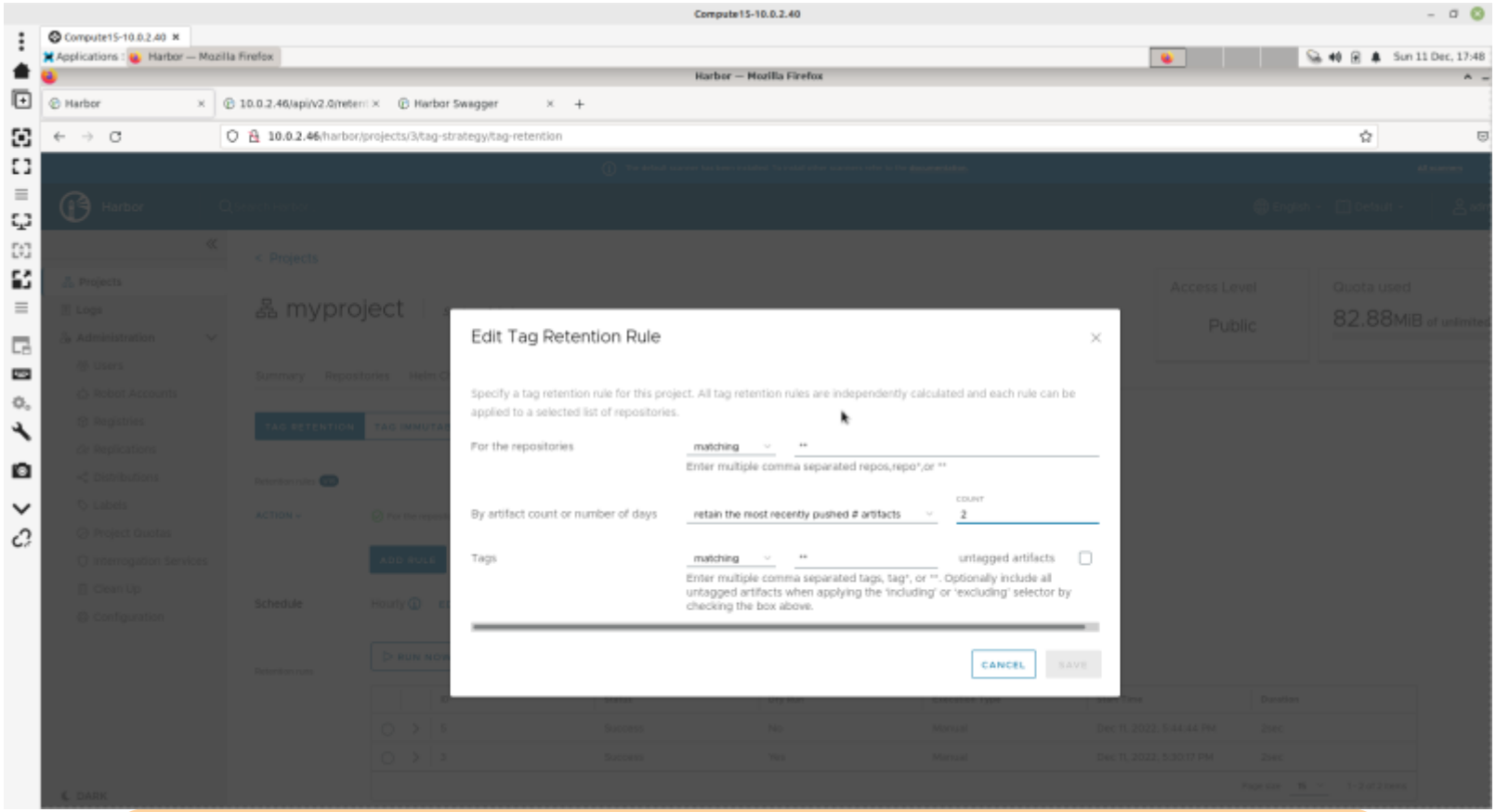
fcf710f98e14ac3df4d3cf365f04423da8ceb42325eca9efe87f67082904728e par rapport à **b918acd244c70cf528d8e96584d8d34099d2a8eefc1edd86ba972dd975934f52**. Par contre, puisque le même tag a été utilisé, le tag **rel2** a été attribué au dernier Artefact créé et le même tag a été supprimé de l'Artefact dont le SHA256 est **b918acd244c70cf528d8e96584d8d34099d2a8eefc1edd86ba972dd975934f52**

5.11 - Rétention de Tags

Pour gérer ce comportement, il est possible de créer des règles. Cliquez sur **myproject > Policy** :

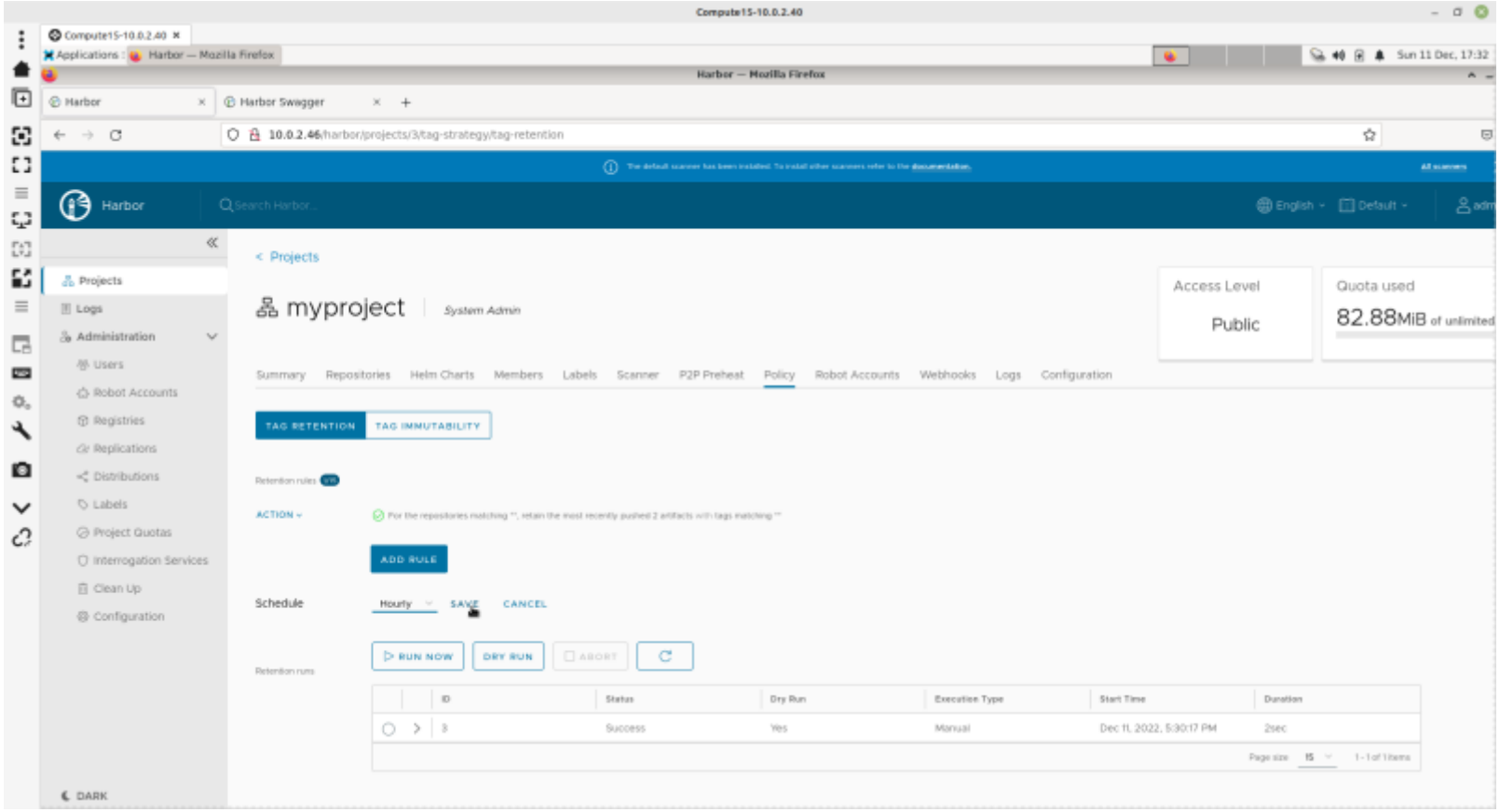


Cliquez sur **ADD RULE** et remplissez le formulaire ainsi :

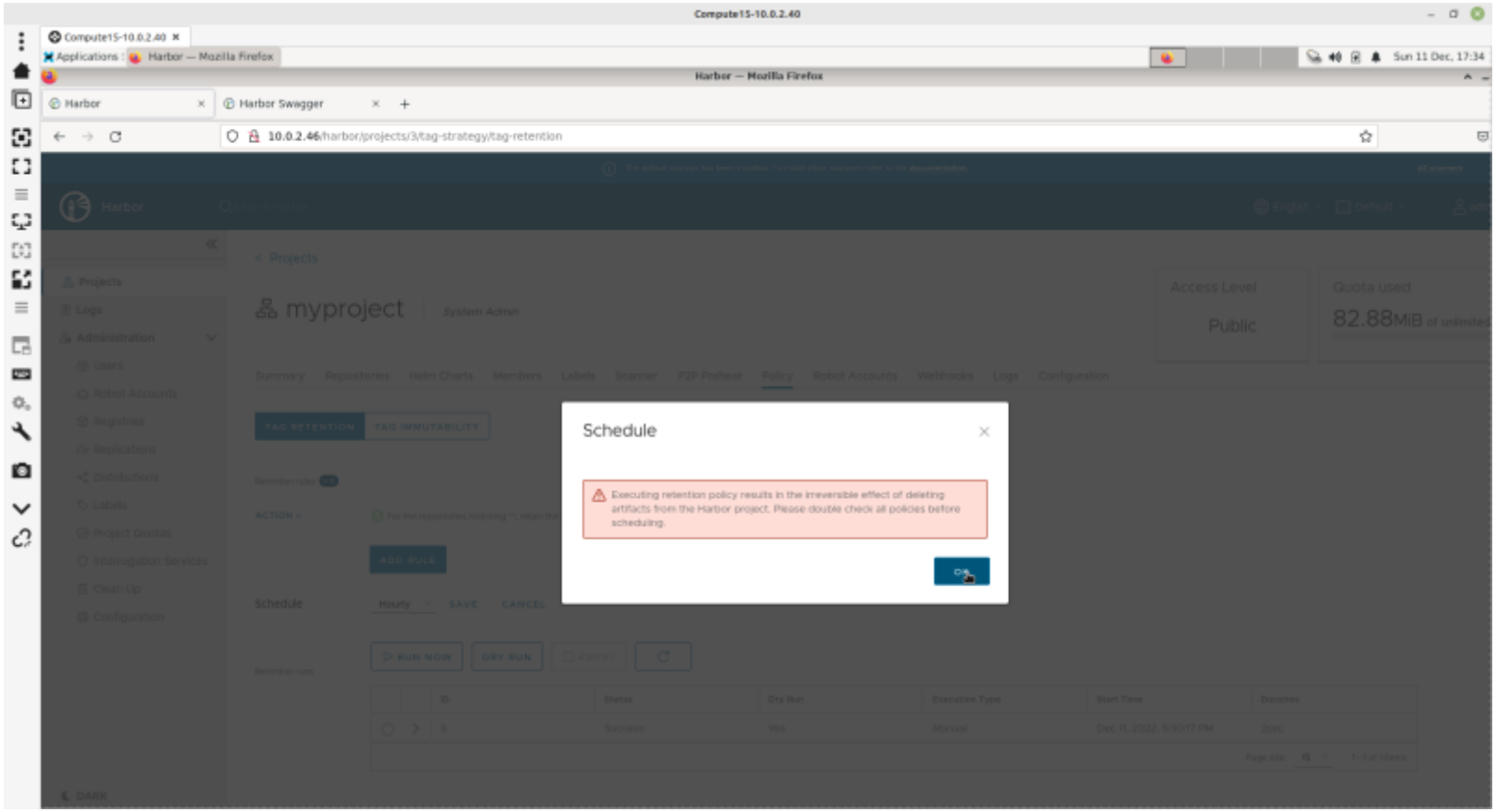


Important : Cette règle s'appliquera à tous les dépôts et concernera tous les tags. La règle retiendra les 2 Artefacts les plus recemment poussés vers le dépôt.

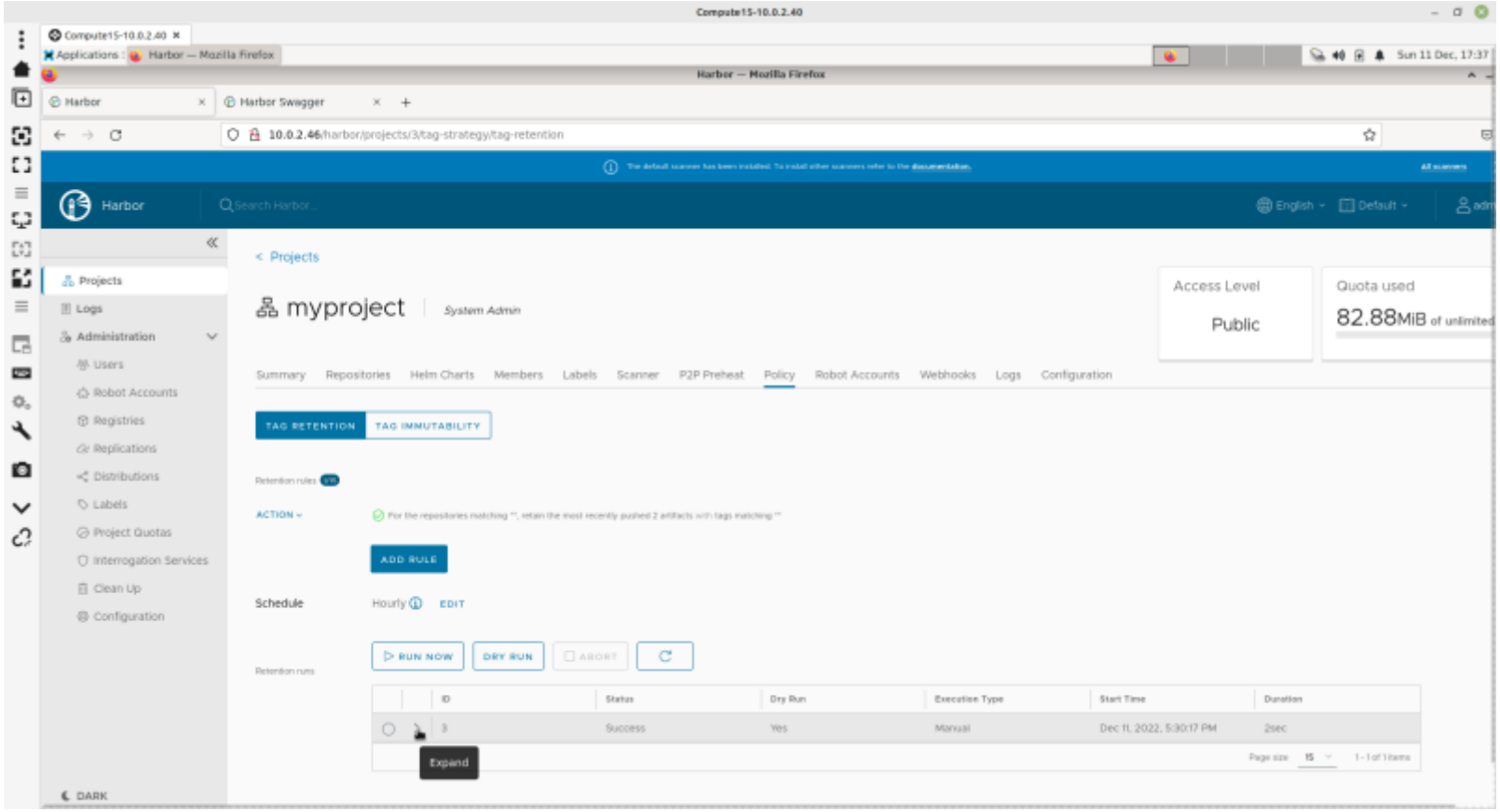
Cliquez sur **EDIT** sur la ligne **Schedule** et choisissez **hourly** puis cliquez **Save** :



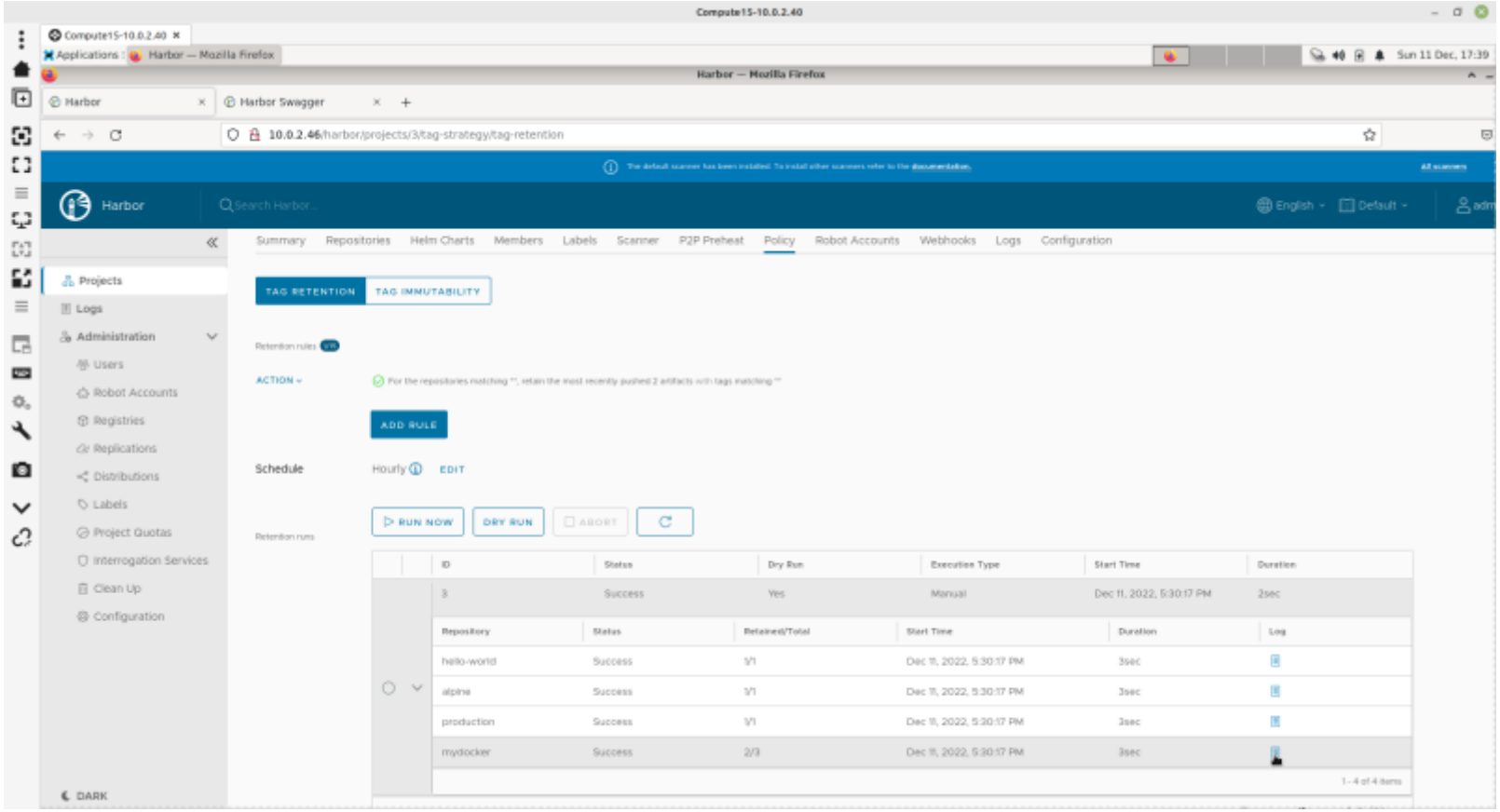
Cliquez sur **OK** dans la boîte d'avertissement :



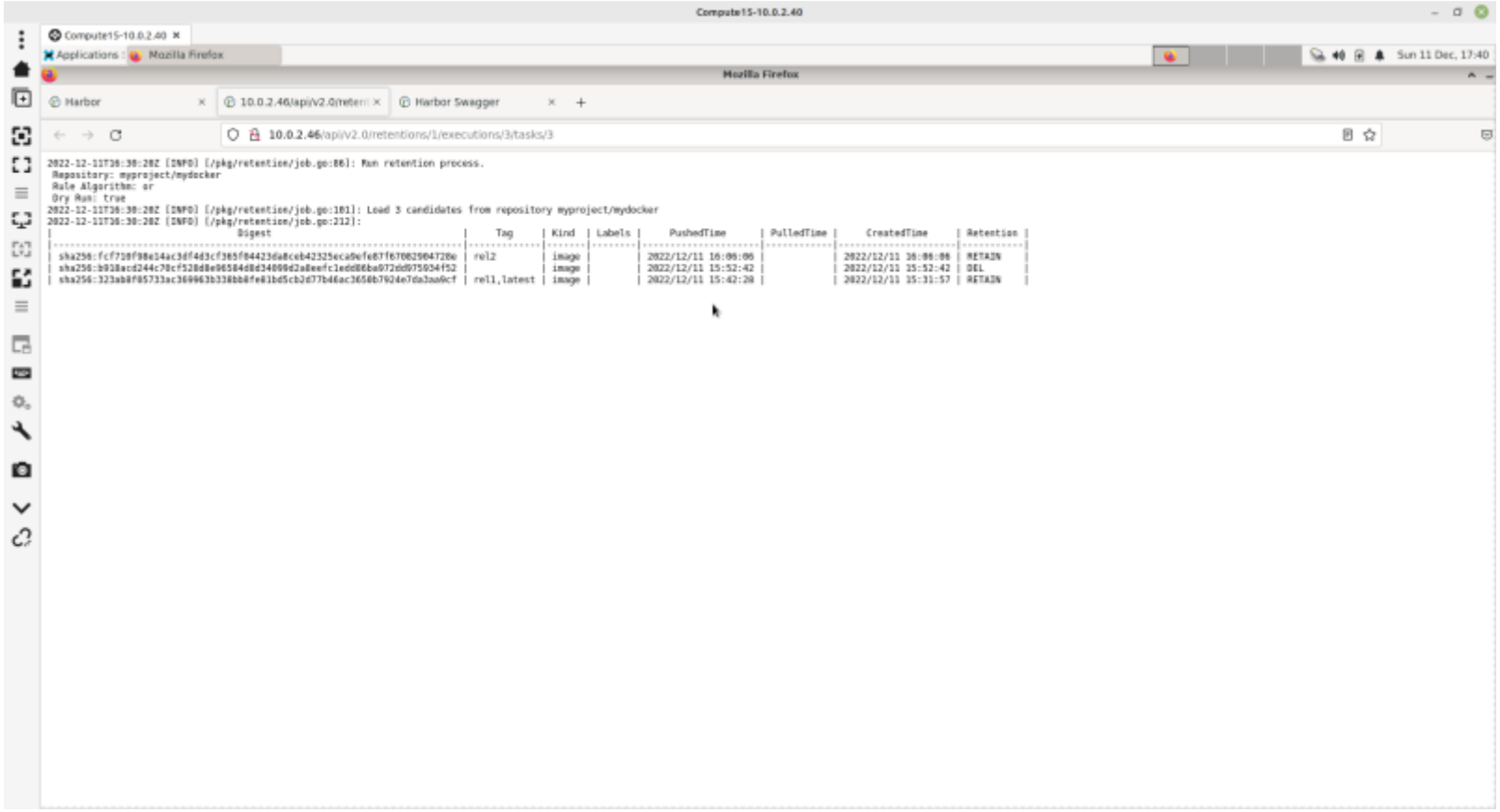
Cliquez ensuite sur le bouton **DRY RUN**. Quand le Dry Run a terminé avec le message **Success**, cliquez sur le bouton **Expand** du Dry Run :



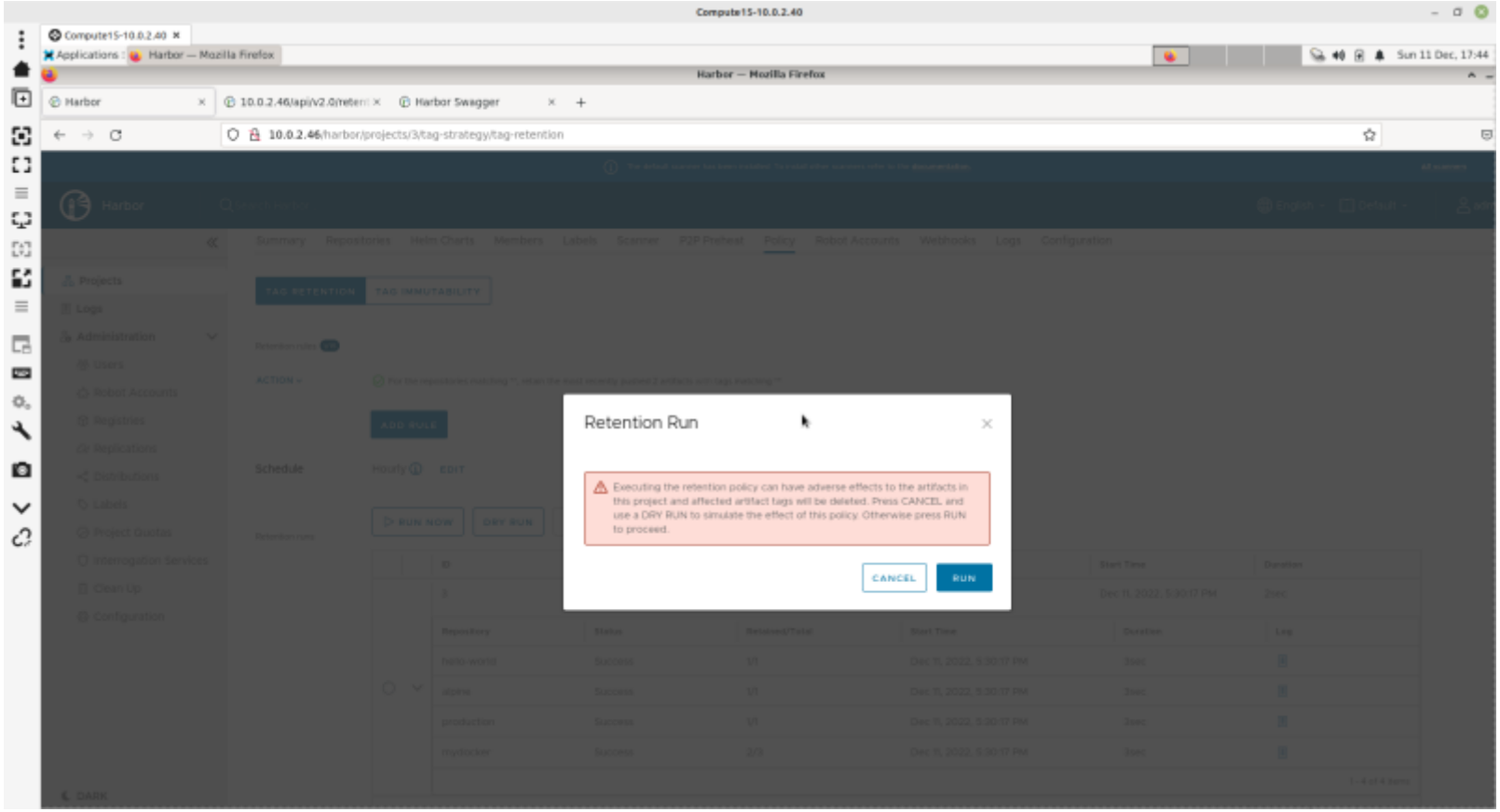
Sur la dernière ligne, cliquez sur le bouton **Log** :



Consultez la colonne **Retention** :



Revenez dans l'interface de Harbor et exécutez la règle en cliquant sur le bouton **RUN NOW**, puis cliquez sur **RUN** dans la boîte d'avertissement :



Editez de nouveau le Dockerfile et changez la valeur de **myVariable** de 5 à **10** :

```
root@debian11:~/myDocker# vi Dockerfile
root@debian11:~/myDocker# cat Dockerfile
FROM centos:latest
MAINTAINER ittraining "infos@ittraining.io"
COPY myEntrypoint.sh /entrypoint.sh
ENV myVariable 10
ENTRYPOINT ["/entrypoint.sh"]
CMD ["mycommand"]
```

Compilez de nouveau l'image :

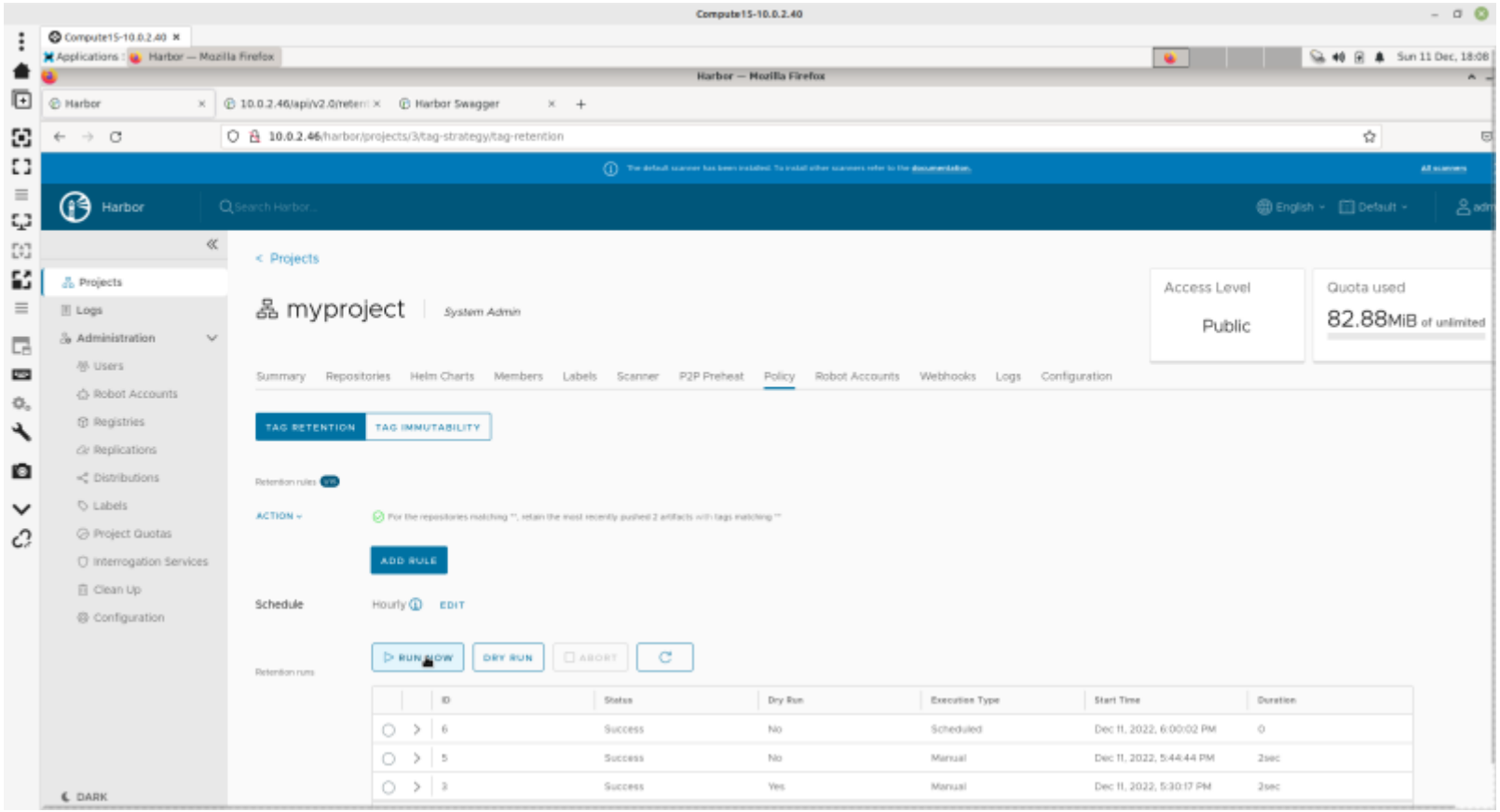
```
root@debian11:~/myDocker# docker build --tag "10.0.2.46:80/myproject/mydocker:rel3" .
Sending build context to Docker daemon  3.072kB
Step 1/6 : FROM centos:latest
 ---> 5d0da3dc9764
Step 2/6 : MAINTAINER ittraining "infos@ittraining.io"
 ---> Using cache
 ---> 7ce8d150d72d
Step 3/6 : COPY myEntrypoint.sh /entrypoint.sh
 ---> Using cache
 ---> c1fda8d1e43c
Step 4/6 : ENV myVariable 10
 ---> Running in 7c938717175f
Removing intermediate container 7c938717175f
 ---> ef07118050c5
Step 5/6 : ENTRYPOINT ["/entrypoint.sh"]
 ---> Running in b2dc04a9363d
Removing intermediate container b2dc04a9363d
 ---> f73a3c2f7930
Step 6/6 : CMD ["mycommand modified"]
 ---> Running in fd7a8952dd7f
Removing intermediate container fd7a8952dd7f
 ---> 3bbe34d8d430
Successfully built 3bbe34d8d430
Successfully tagged 10.0.2.46:80/myproject/mydocker:rel3
```

Procédez maintenant au **push** :

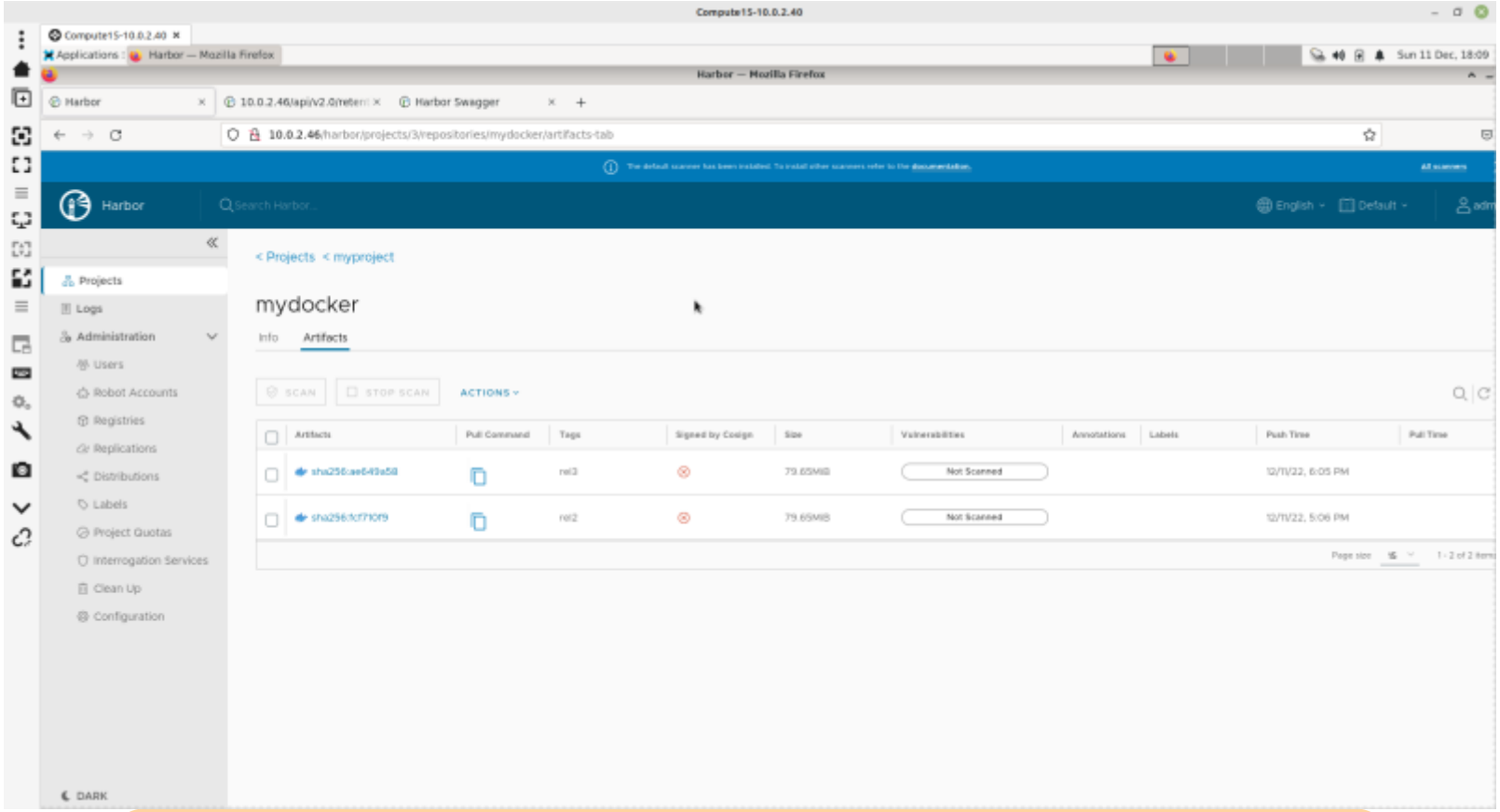
```
root@debian11:~/myDocker# docker push 10.0.2.46:80/myproject/mydocker:rel3
The push refers to repository [10.0.2.46:80/myproject/mydocker]
7c7558b01004: Layer already exists
74ddd0ec08fa: Layer already exists
```

```
rel3: digest: sha256:ae649a586f5d9dcae71440bf88eb98d95c3cc86d40f27eb54fa68e65869c3b48 size: 736
```

Exécutez la règle :



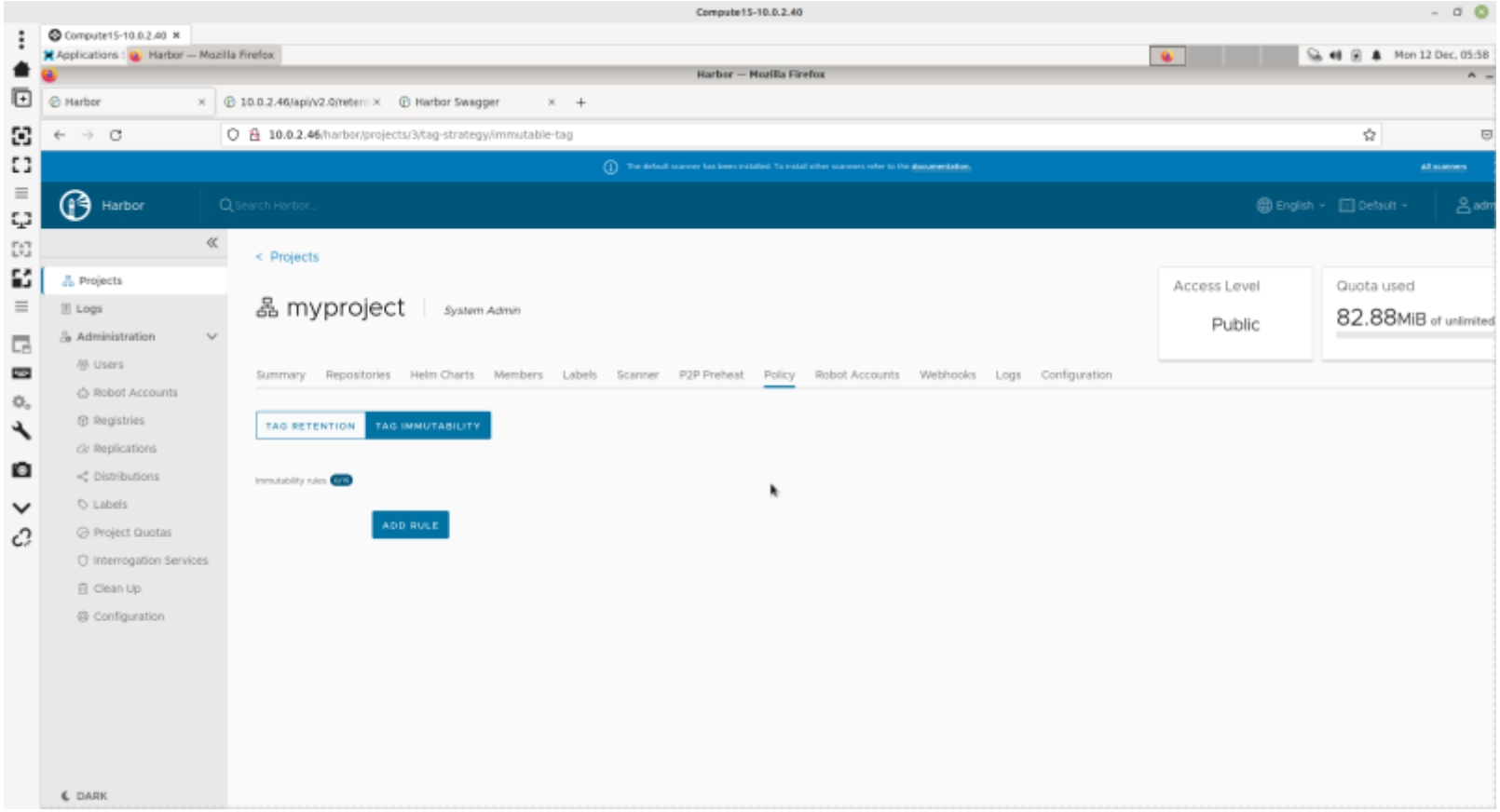
Constatez la liste des Artefacts dans **mydocker** :



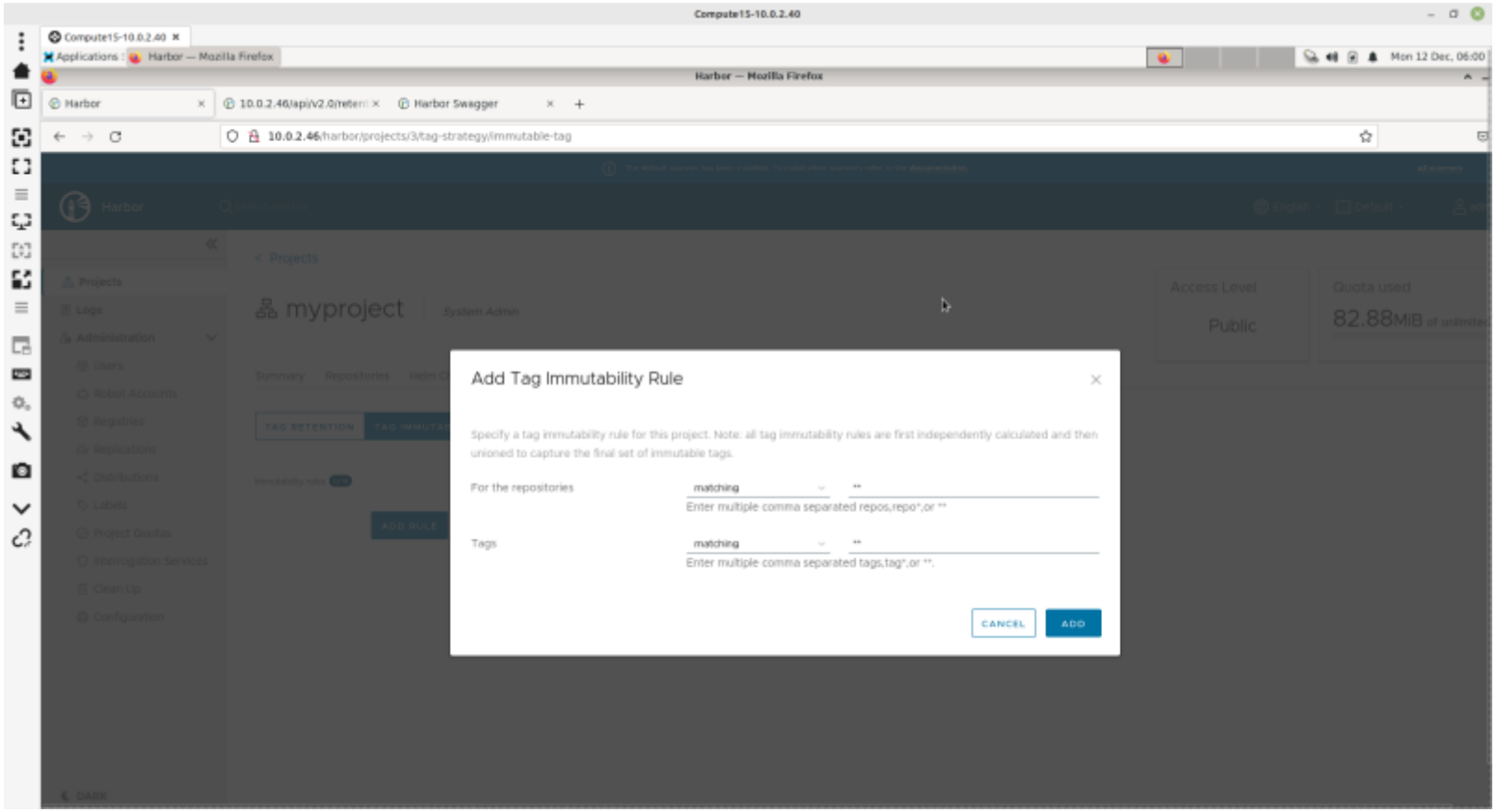
Important : Notez que c'est l'Artefact le plus ancien qui a été supprimé.

5.12 - Immutabilité des Tags

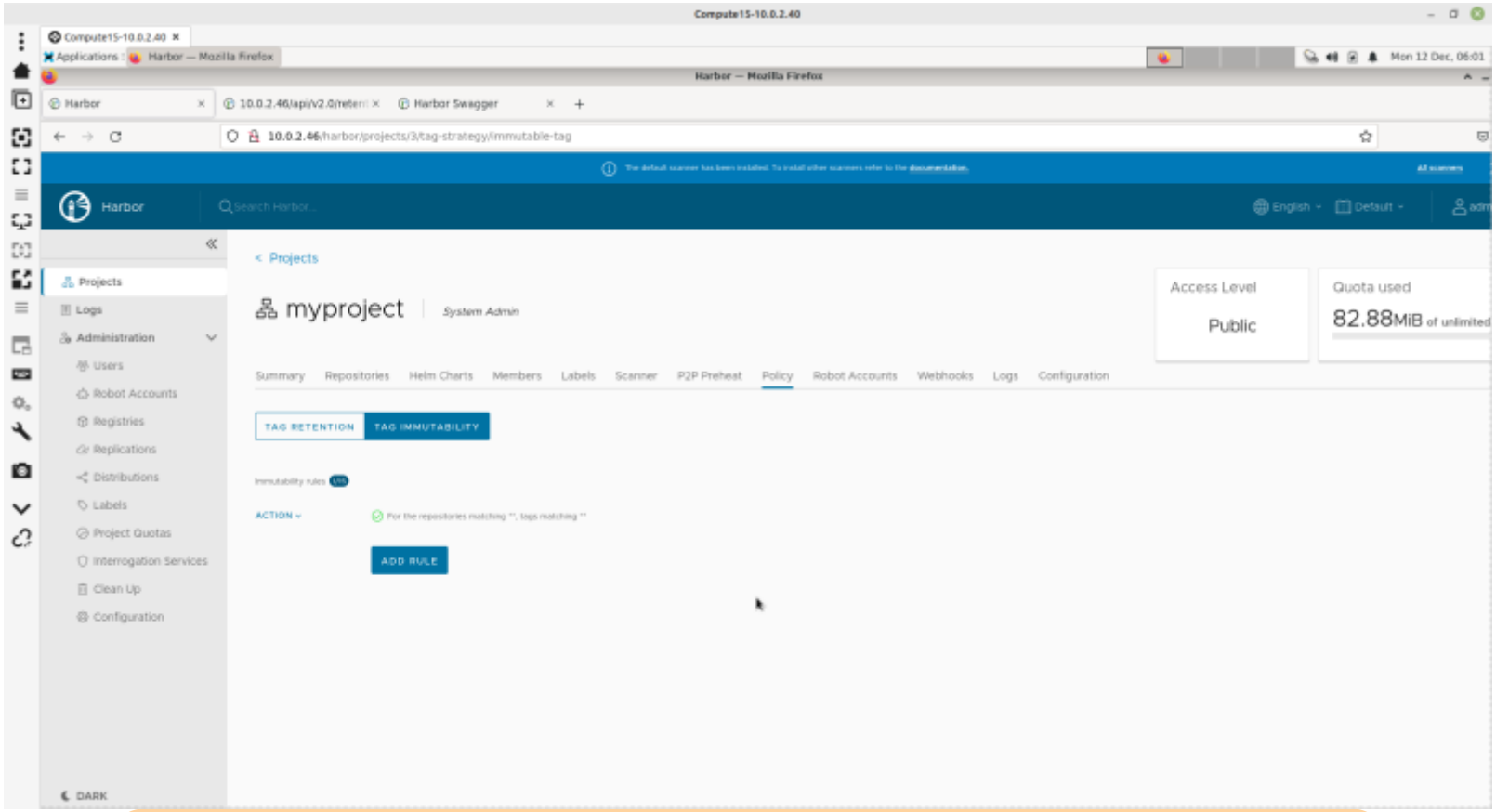
Cliquez sur **myproject > Policy > TAG IMMUTABILITY** :



Cliquez maintenant sur le bouton **ADD RULE** :



Cliquez sur le bouton **ADD** :



Important : Cette règle s'appliquera à tous les dépôts et concernera tous les tags.

Essayez de faire un push de l'image **mydocker:rel2** :

```
root@debian11:~/myDocker# docker push 10.0.2.46:80/myproject/mydocker:rel2
The push refers to repository [10.0.2.46:80/myproject/mydocker]
7c7558b01004: Layer already exists
74ddd0ec08fa: Layer already exists
```

```
unknown: Failed to process request due to 'mydocker:rel2' configured as immutable.
```

Important : Notez qu'il n'est pas possible de faire un push car la règle d'immutabilité l'interdit.

Editez le Dockerfile et changez la valeur de **myVariable** de 10 à **8** :

```
root@debian11:~/myDocker# vi Dockerfile
root@debian11:~/myDocker# cat Dockerfile
FROM centos:latest
MAINTAINER ittraining "infos@ittraining.io"
COPY myEntrypoint.sh /entrypoint.sh
ENV myVariable 8
ENTRYPOINT ["/entrypoint.sh"]
CMD ["mycommand"]
```

Faites un build de **mydocker:rel4** :

```
root@debian11:~/myDocker# docker build --tag "10.0.2.46:80/myproject/mydocker:rel4" .
Sending build context to Docker daemon  3.072kB
Step 1/6 : FROM centos:latest
 ---> 5d0da3dc9764
Step 2/6 : MAINTAINER ittraining "infos@ittraining.io"
 ---> Using cache
 ---> 7ce8d150d72d
Step 3/6 : COPY myEntrypoint.sh /entrypoint.sh
 ---> Using cache
 ---> c1fda8d1e43c
Step 4/6 : ENV myVariable 8
 ---> Running in 541758cf561f
Removing intermediate container 541758cf561f
```

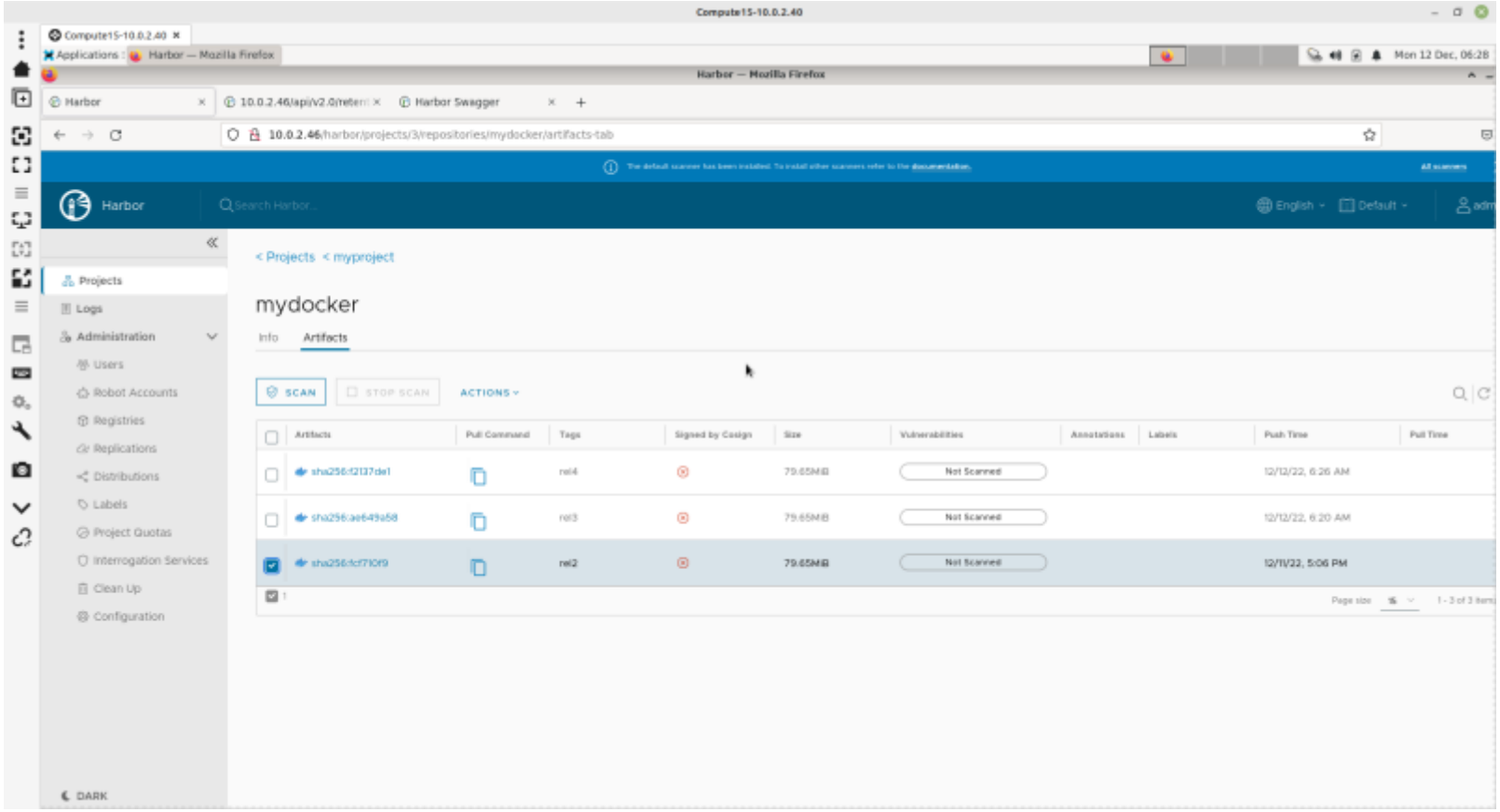
```
 ---> 45f622ff8362
Step 5/6 : ENTRYPOINT ["/entrypoint.sh"]
 ---> Running in 9d309f201613
Removing intermediate container 9d309f201613
 ---> 749b47fe344a
Step 6/6 : CMD ["mycommand modified"]
 ---> Running in e7397e408526
Removing intermediate container e7397e408526
 ---> 85f1cbc19cee
Successfully built 85f1cbc19cee
Successfully tagged 10.0.2.46:80/myproject/mydocker:rel4
```

Procédez maintenant au **push** :

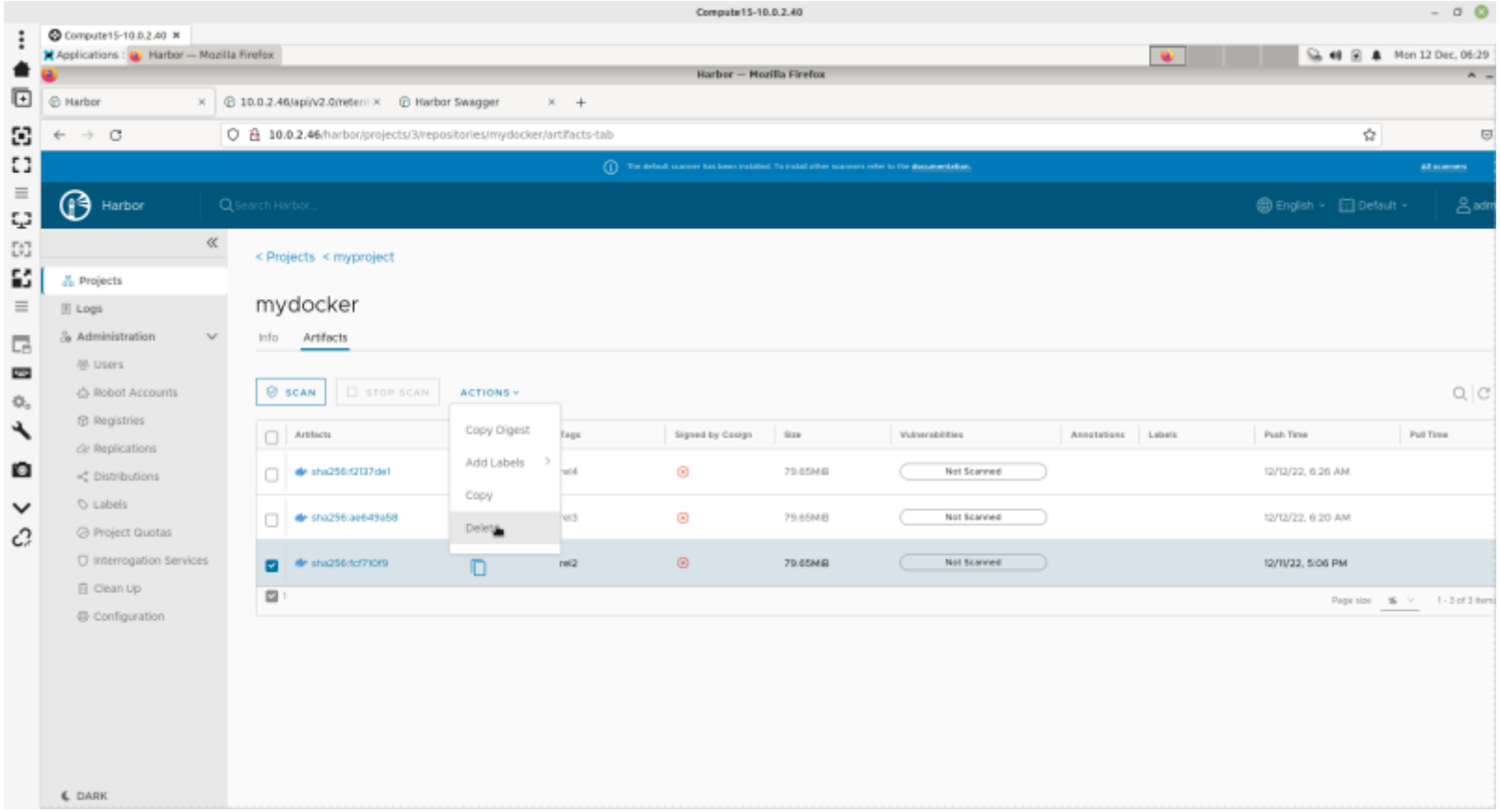
```
root@debian11:~/myDocker# docker push 10.0.2.46:80/myproject/mydocker:rel4
The push refers to repository [10.0.2.46:80/myproject/mydocker]
7c7558b01004: Layer already exists
74ddd0ec08fa: Layer already exists
rel4: digest: sha256:ae649a586f5d9dcae71440bf88eb98d95c3cc86d40f27eb54fa68e65869c3b48 size: 736
```

Important : Notez qu'il est possible de faire un push car le tag rel4 n'existait pas.

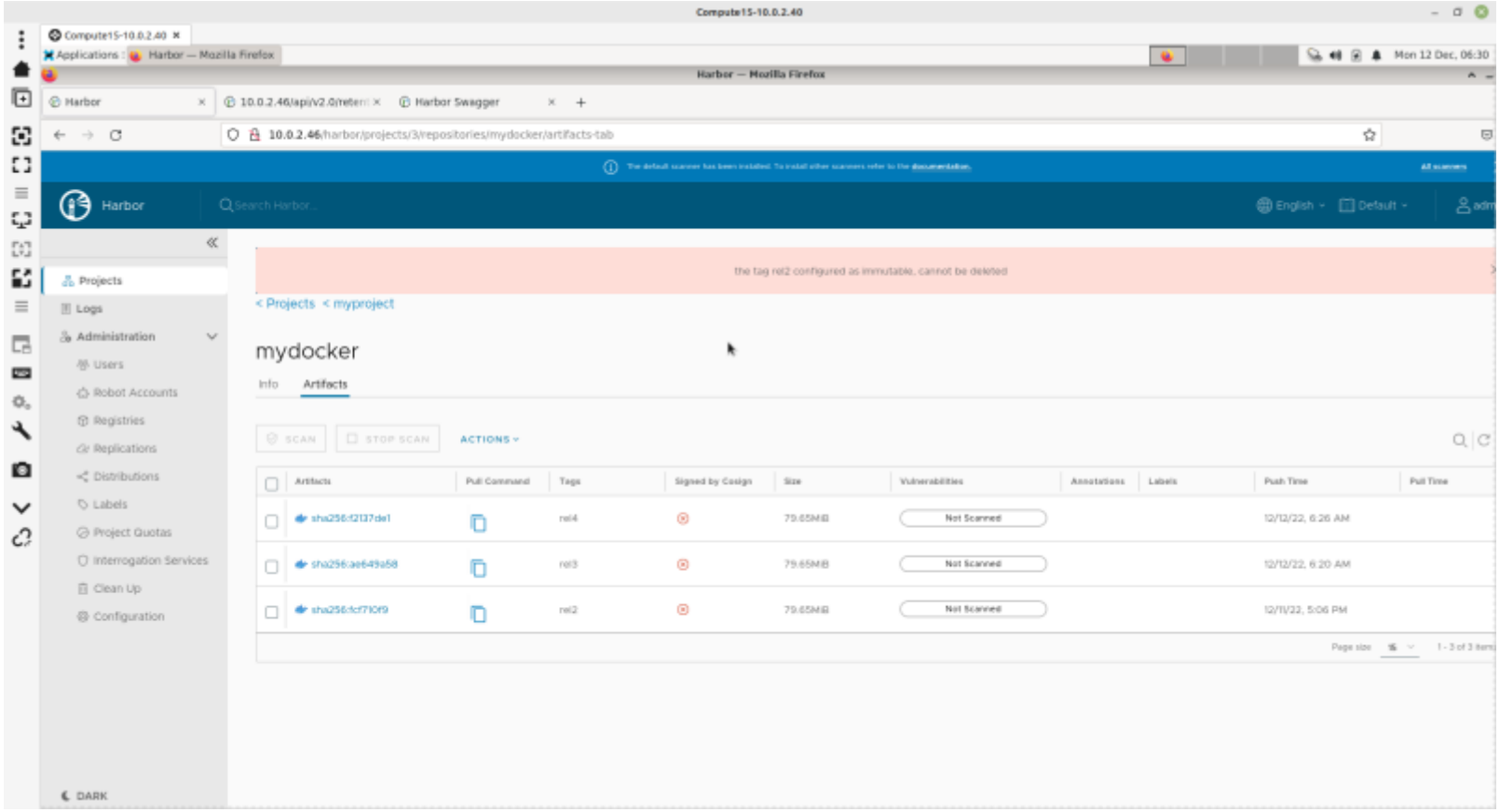
Dans l'interface de Harbor, cliquez sur **Projects > myproject > myproject/mydocker** et cochez **rel2** :



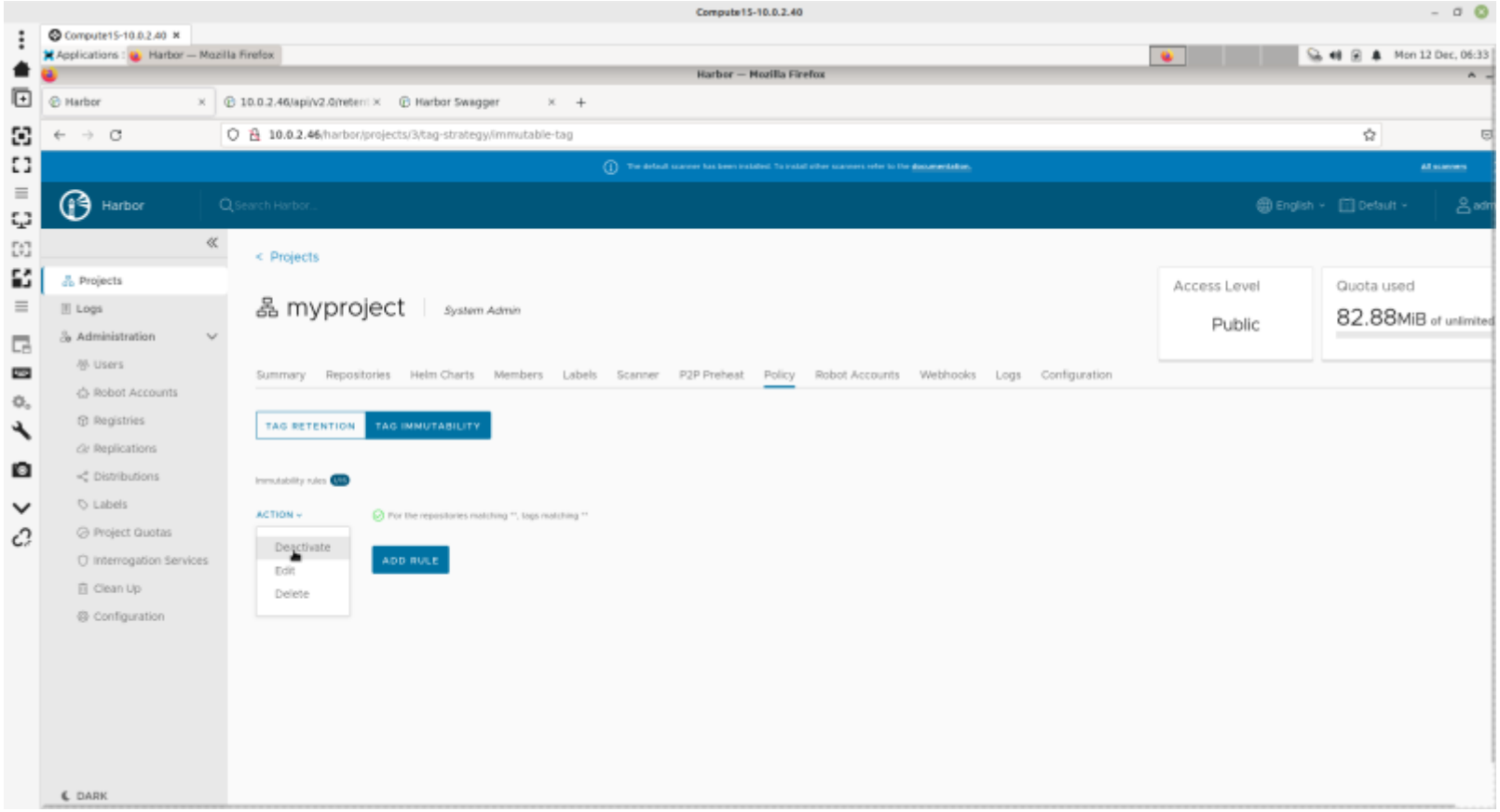
Essayez de supprimer l'Artefact :



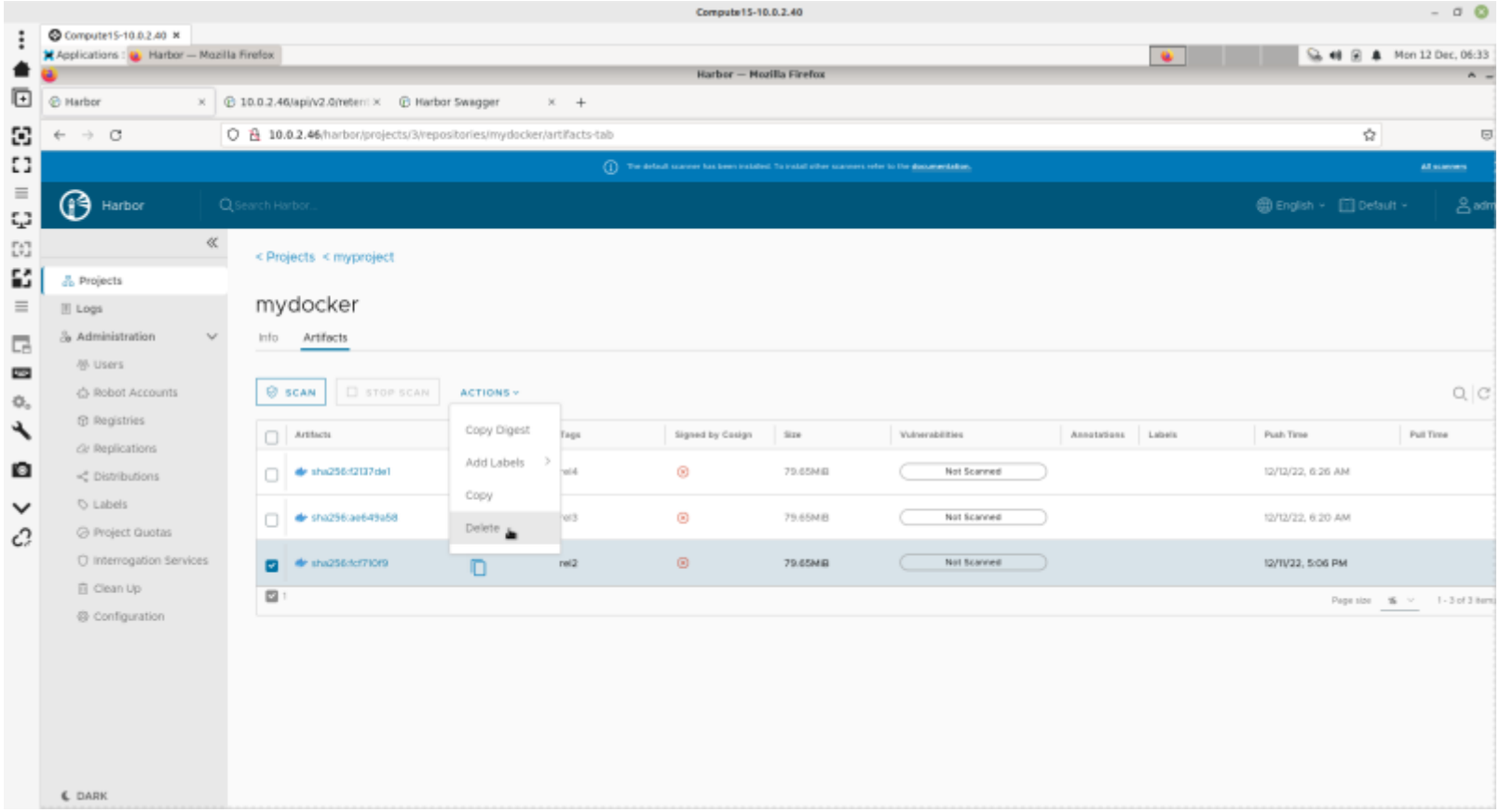
Notez l'erreur :



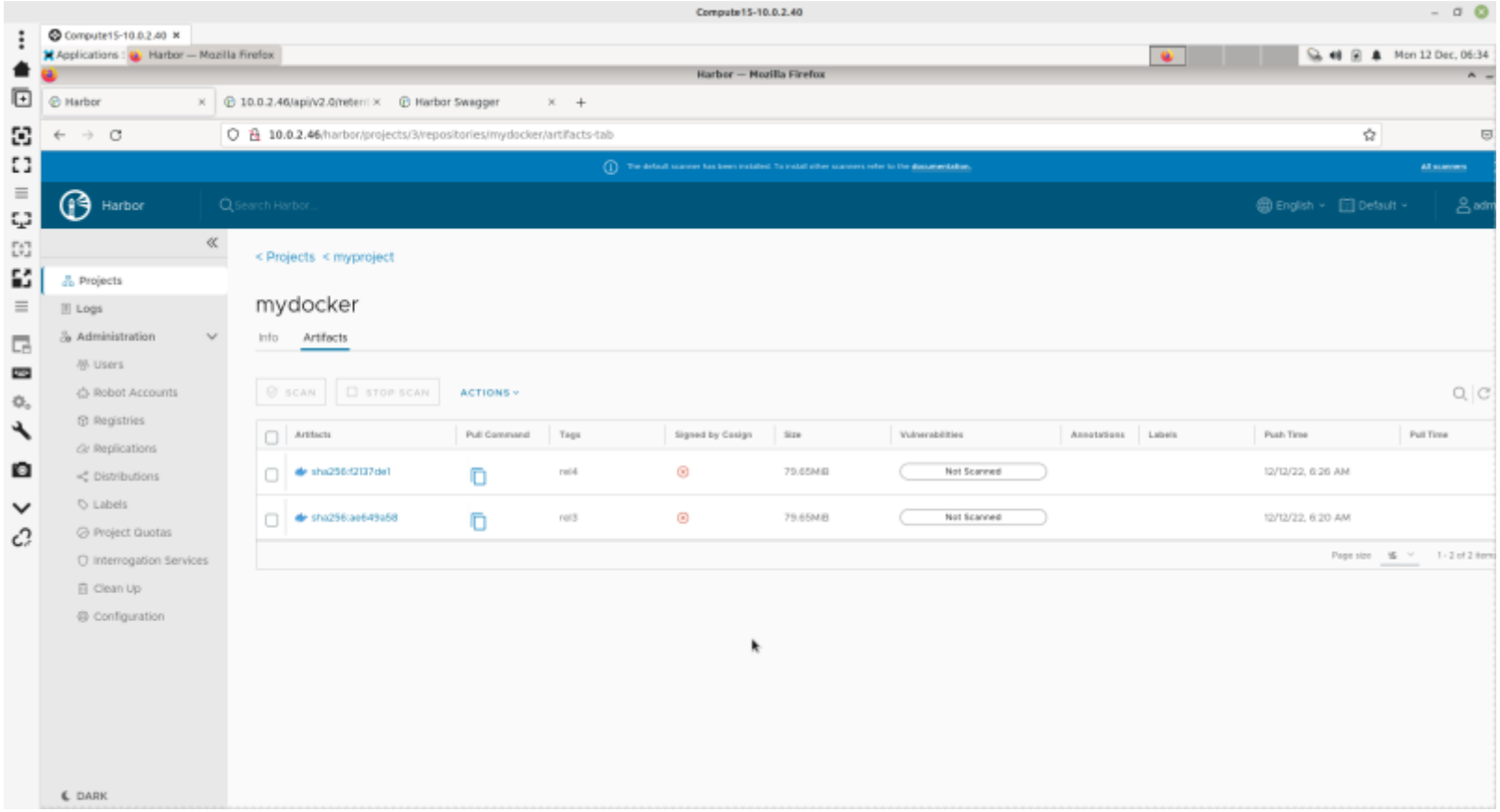
Pour pouvoir supprimer l'Artefact, il faut désactiver la règle d'immutabilité :



Maintenant, essayez de supprimer l'Artefact :



Vous verrez que la suppression a réussi :



Editez le Dockerfile et changez la valeur de **myVariable** de 8 à **7** :

```
root@debian11:~/myDocker# vi Dockerfile
root@debian11:~/myDocker# cat Dockerfile
FROM centos:latest
MAINTAINER ittraining "infos@ittraining.io"
COPY myEntrypoint.sh /entrypoint.sh
ENV myVariable 7
ENTRYPOINT ["/entrypoint.sh"]
CMD ["mycommand"]
```

Faites un build de **mydocker:rel5** :

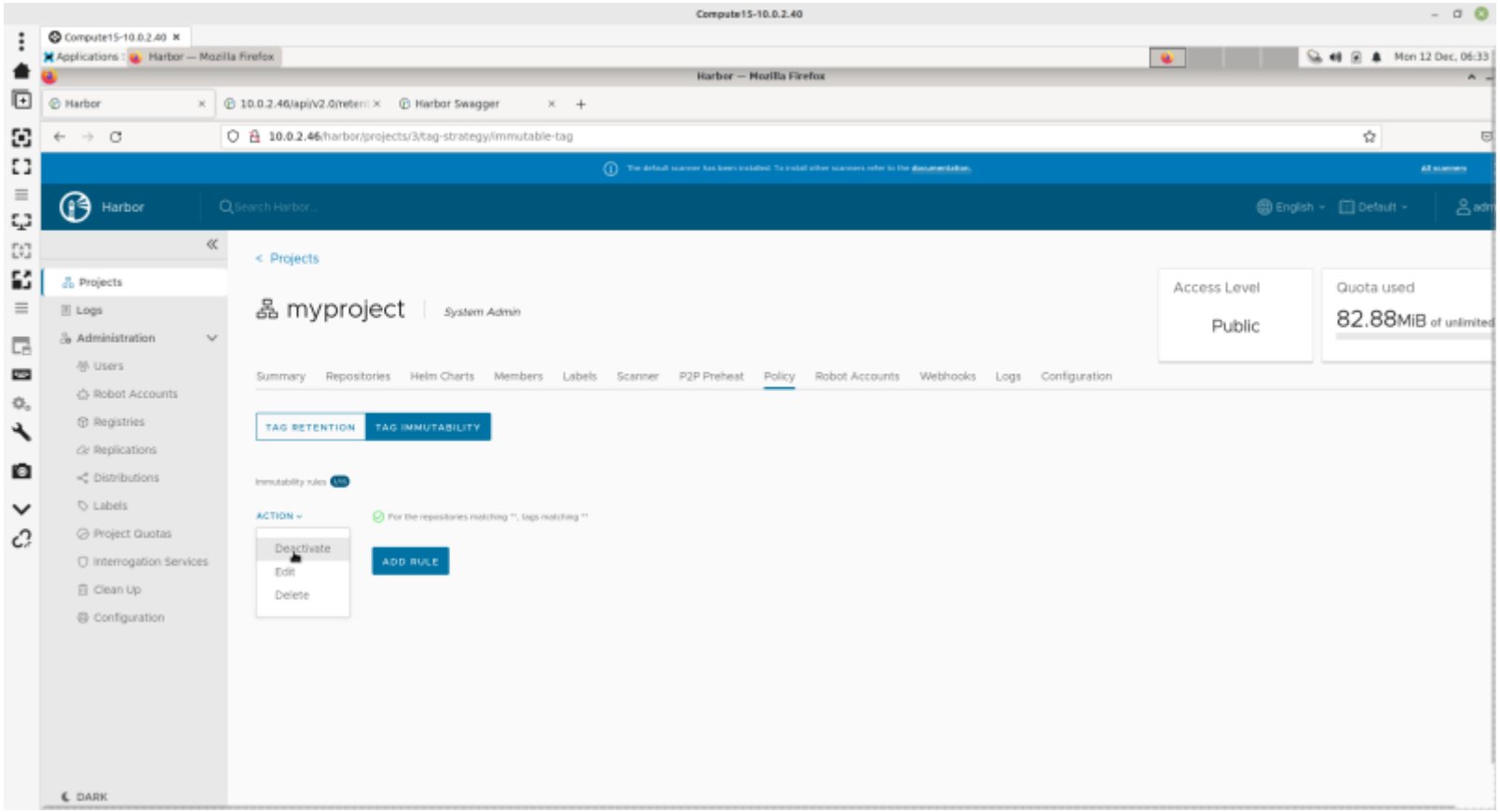
```
root@debian11:~/myDocker# docker build --tag "10.0.2.46:80/myproject/mydocker:rel5" .
Sending build context to Docker daemon  3.072kB
Step 1/6 : FROM centos:latest
 ---> 5d0da3dc9764
Step 2/6 : MAINTAINER ittraining "infos@ittraining.io"
 ---> Using cache
 ---> 7ce8d150d72d
Step 3/6 : COPY myEntrypoint.sh /entrypoint.sh
 ---> Using cache
 ---> c1fda8d1e43c
Step 4/6 : ENV myVariable 7
 ---> Running in b2153c9b0781
Removing intermediate container b2153c9b0781
 ---> 3b18a7da4503
Step 5/6 : ENTRYPOINT ["/entrypoint.sh"]
 ---> Running in 81b97d9082ed
Removing intermediate container 81b97d9082ed
 ---> c283e1e64dca
Step 6/6 : CMD ["mycommand modified"]
 ---> Running in 7252818184fc
Removing intermediate container 7252818184fc
 ---> f9466edbe1f3
Successfully built f9466edbe1f3
Successfully tagged 10.0.2.46:80/myproject/mydocker:rel5
```

Procédez maintenant au **push** :

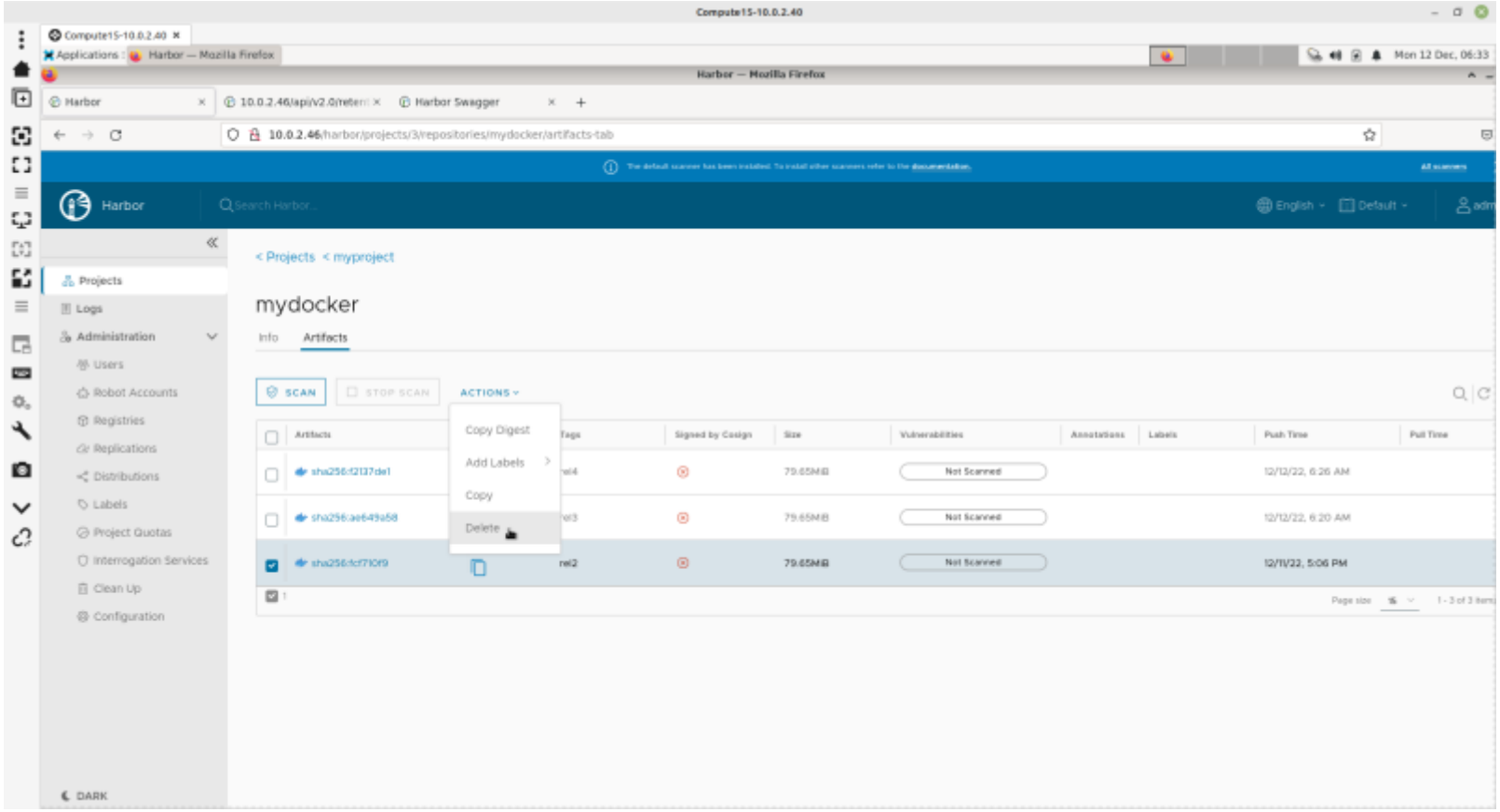
```
root@debian11:~/myDocker# docker push 10.0.2.46:80/myproject/mydocker:rel5
The push refers to repository [10.0.2.46:80/myproject/mydocker]
7c7558b01004: Layer already exists
74ddd0ec08fa: Layer already exists
```

```
rel5: digest: sha256:887e7751d8a7271b238f536327c38a867881a9c354ff54d9ab5db45502a160d7 size: 736
```

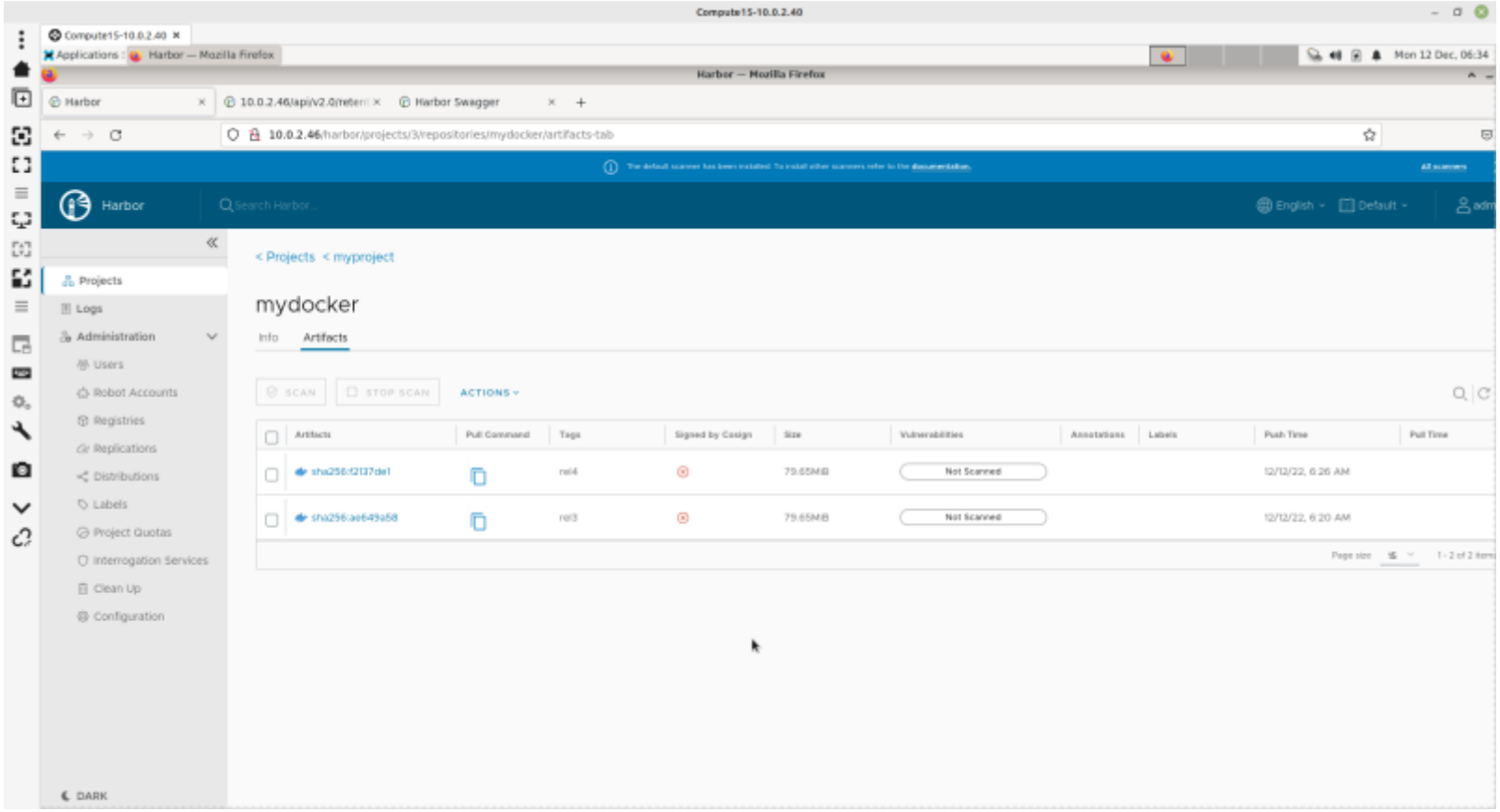
Pour pouvoir supprimer l'Artefact, il faut désactiver la règle d'immutabilité :



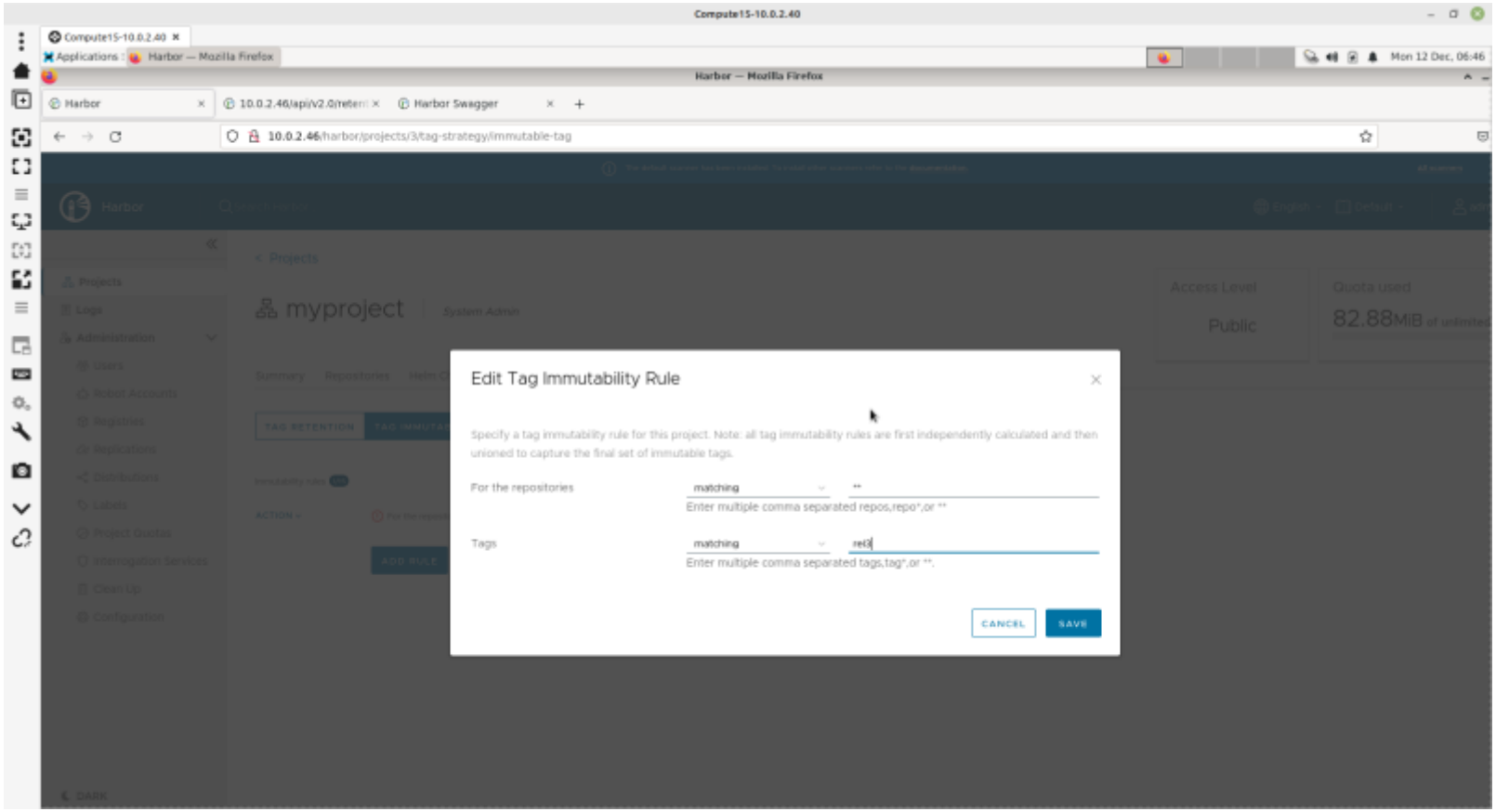
Maintenant, essayez de supprimer l'Artefact :



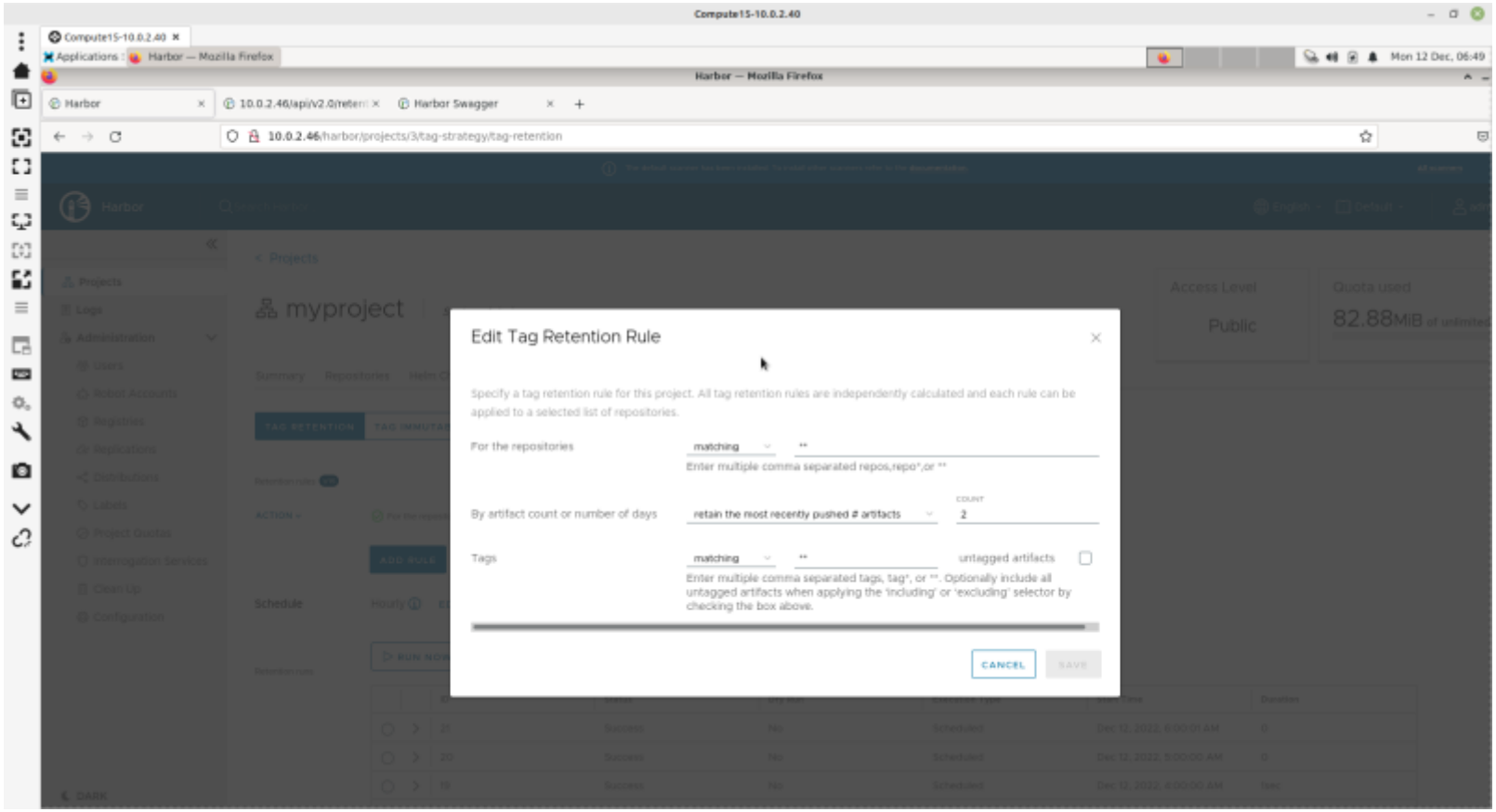
Vous verrez que la suppression a réussi :



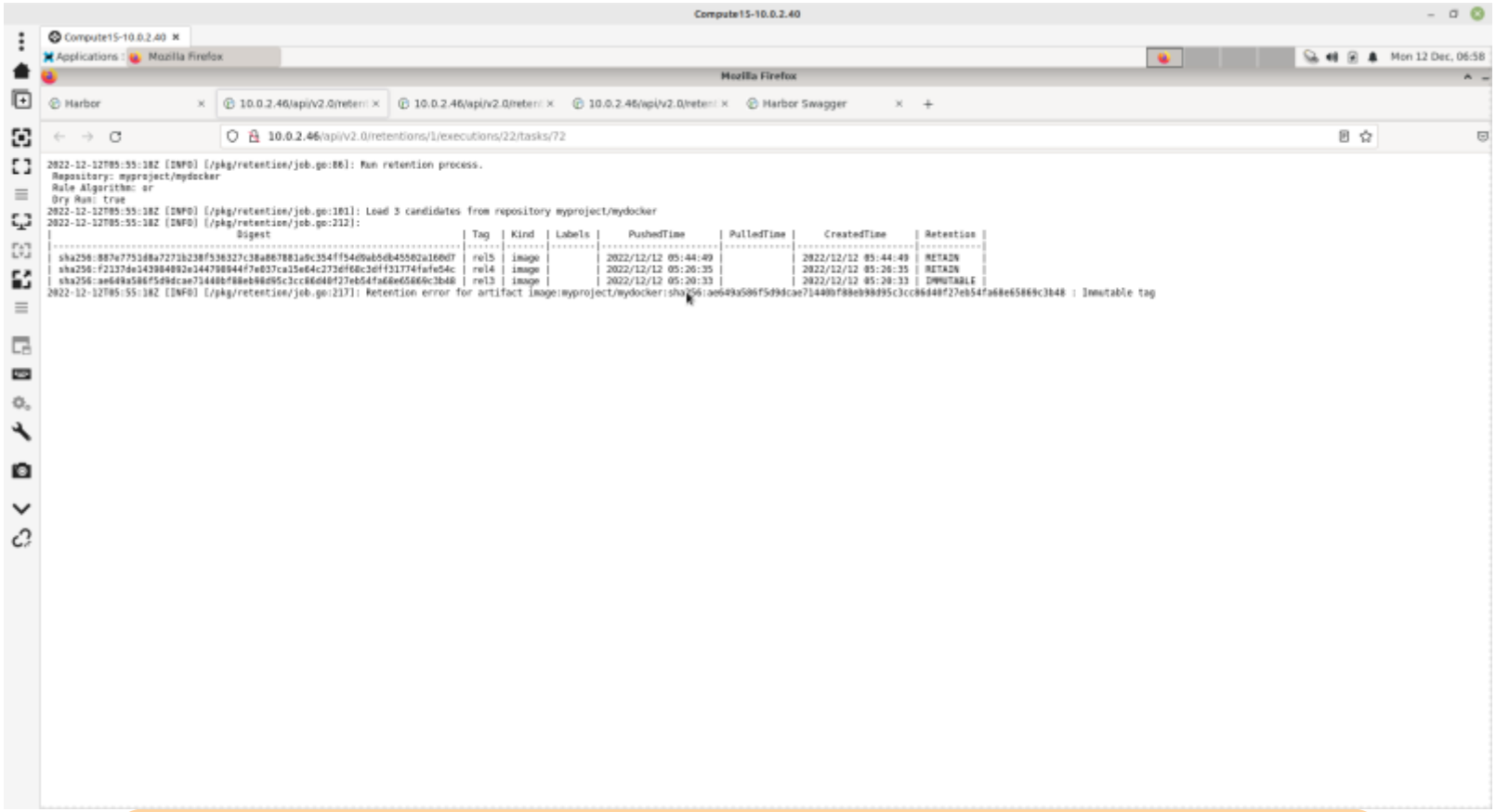
Retournez dans Harbor et modifier la règle d'immutabilité en ne spécifiant que le tag **rel3** puis activez cette règle :



Rappelez-vous que la règle de Rétention de Tags ne garde que les deux Artefacts les plus récents :



Procédez à un Dry Run de la règle de Rétention de Tags et consultez le log de mydocker :



Important : Notez que la règle d'immutabilité empêche la règle de rétention des tags de supprimer l'Artefact **rel3**.