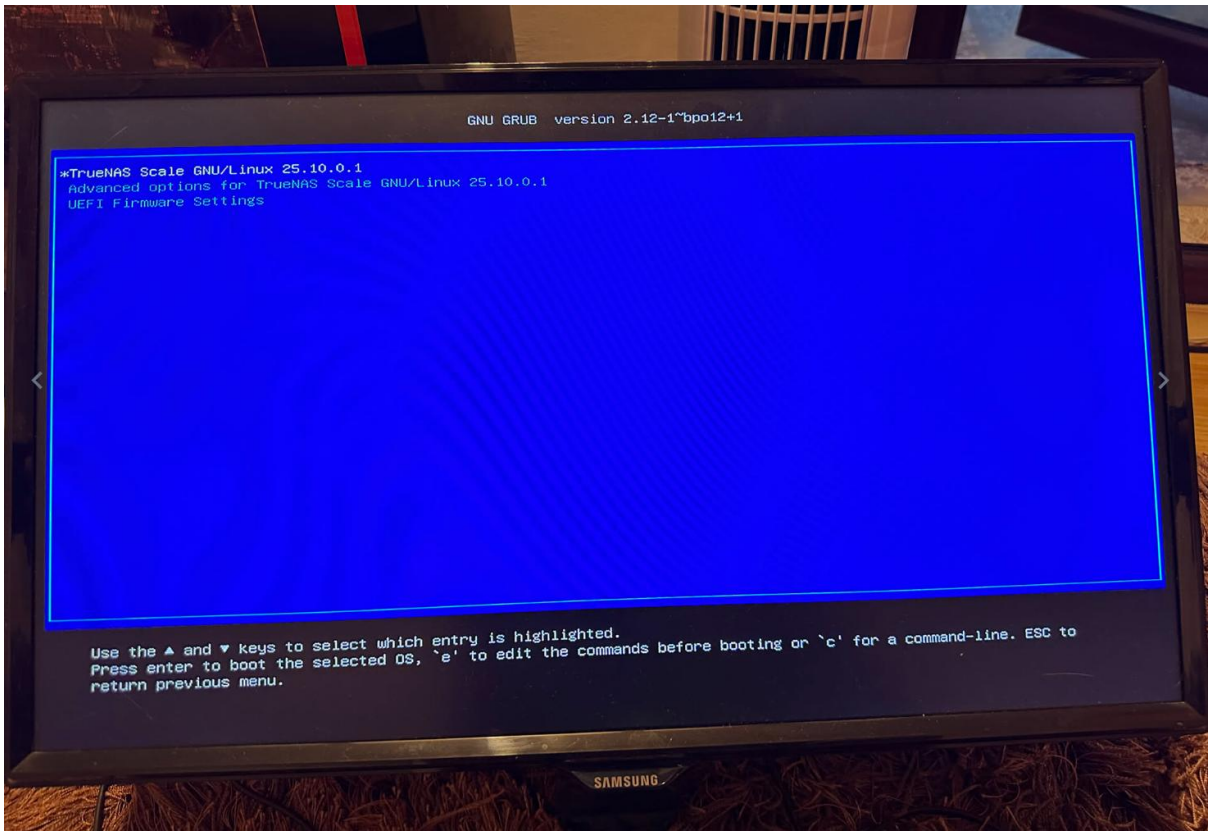
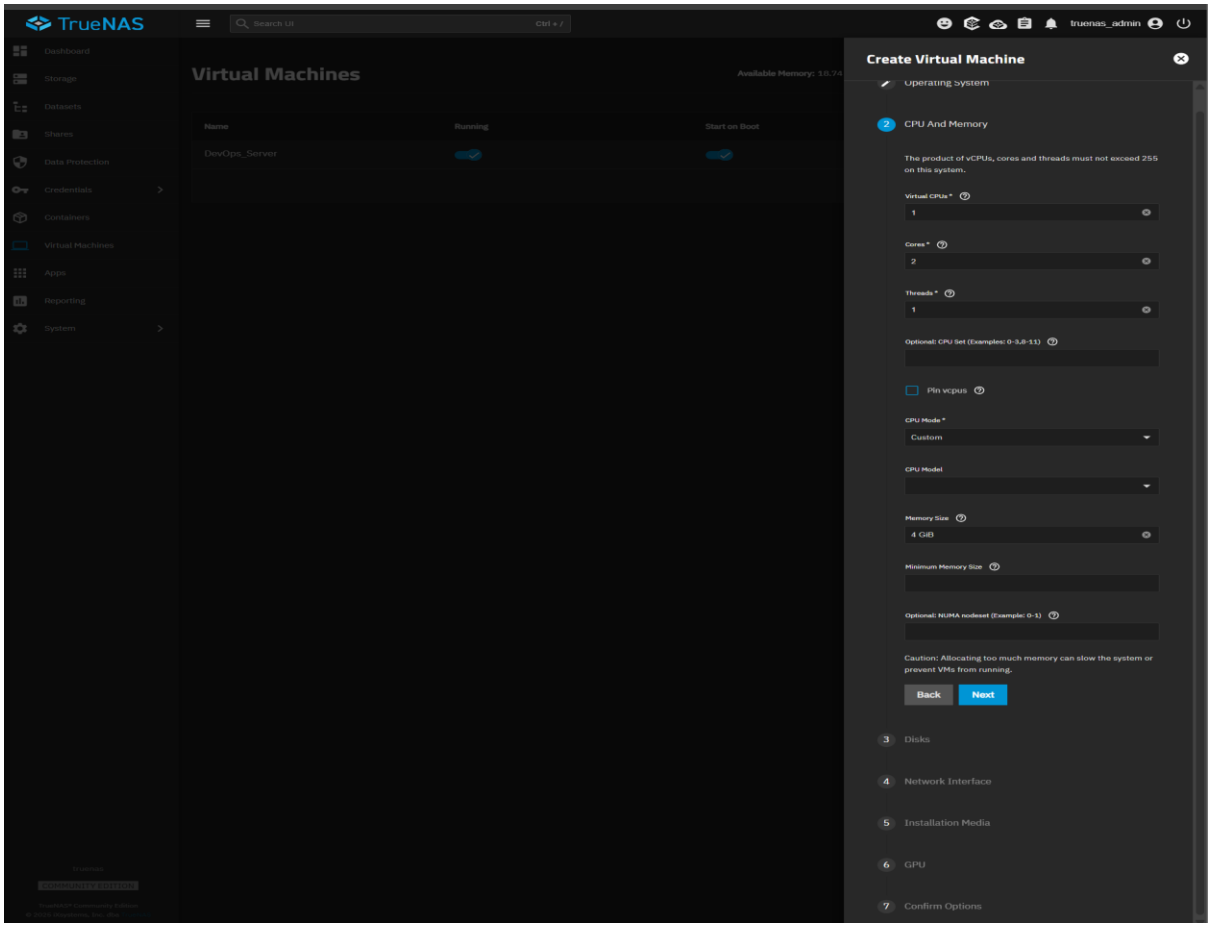
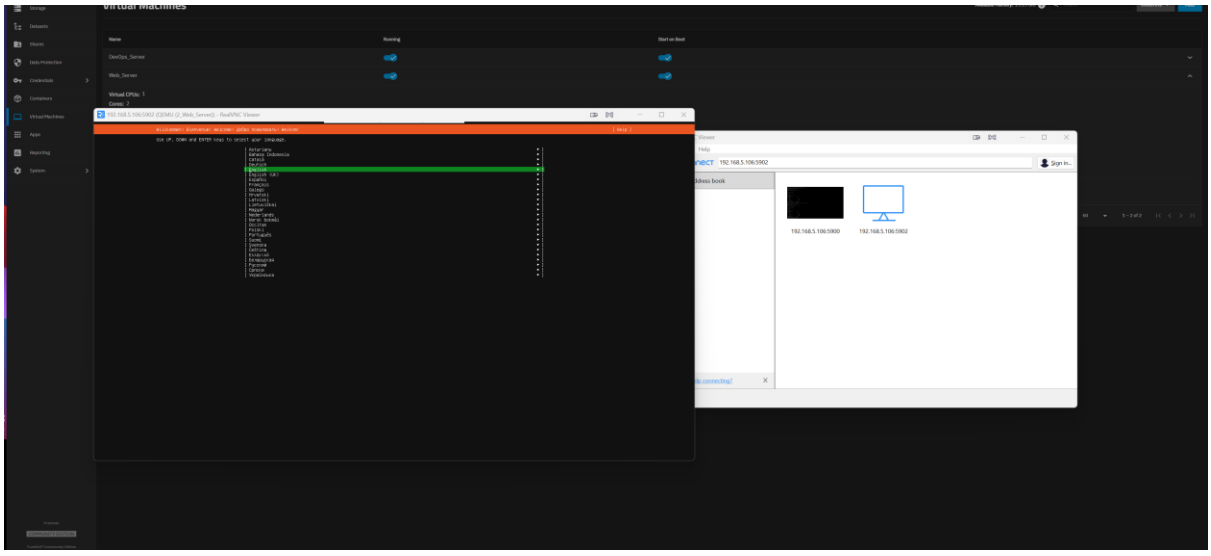
Pentru acest proiect de laborator am optat sa folosesc propriul VPS(un VM facut pe serverul meu propriu de TRUENAS SCALE) chiar daca ne era furnizat unul deja



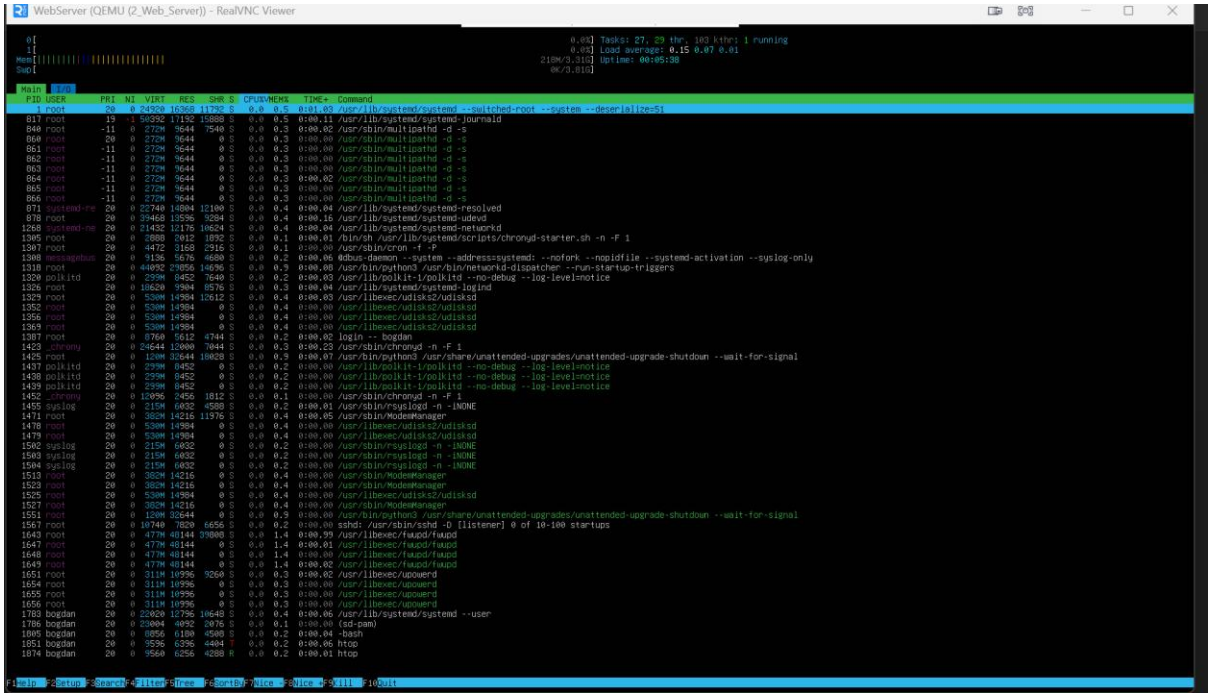




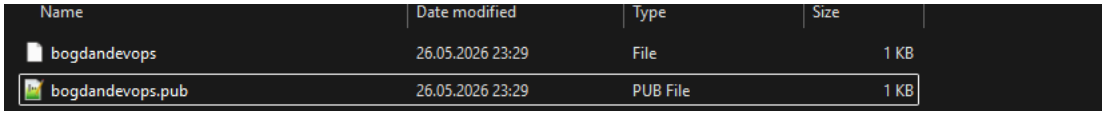
Setupul pentru VM-ul de webserver ^^^

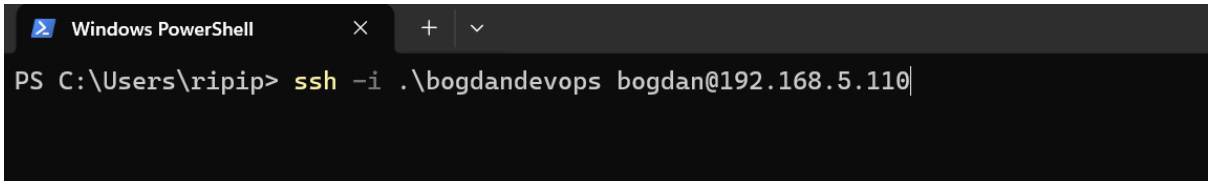
Pentru dev/devops stuff am facut alt VM separat ca sa fie ordine (si best practice din ce am citit)

Dupa logare:



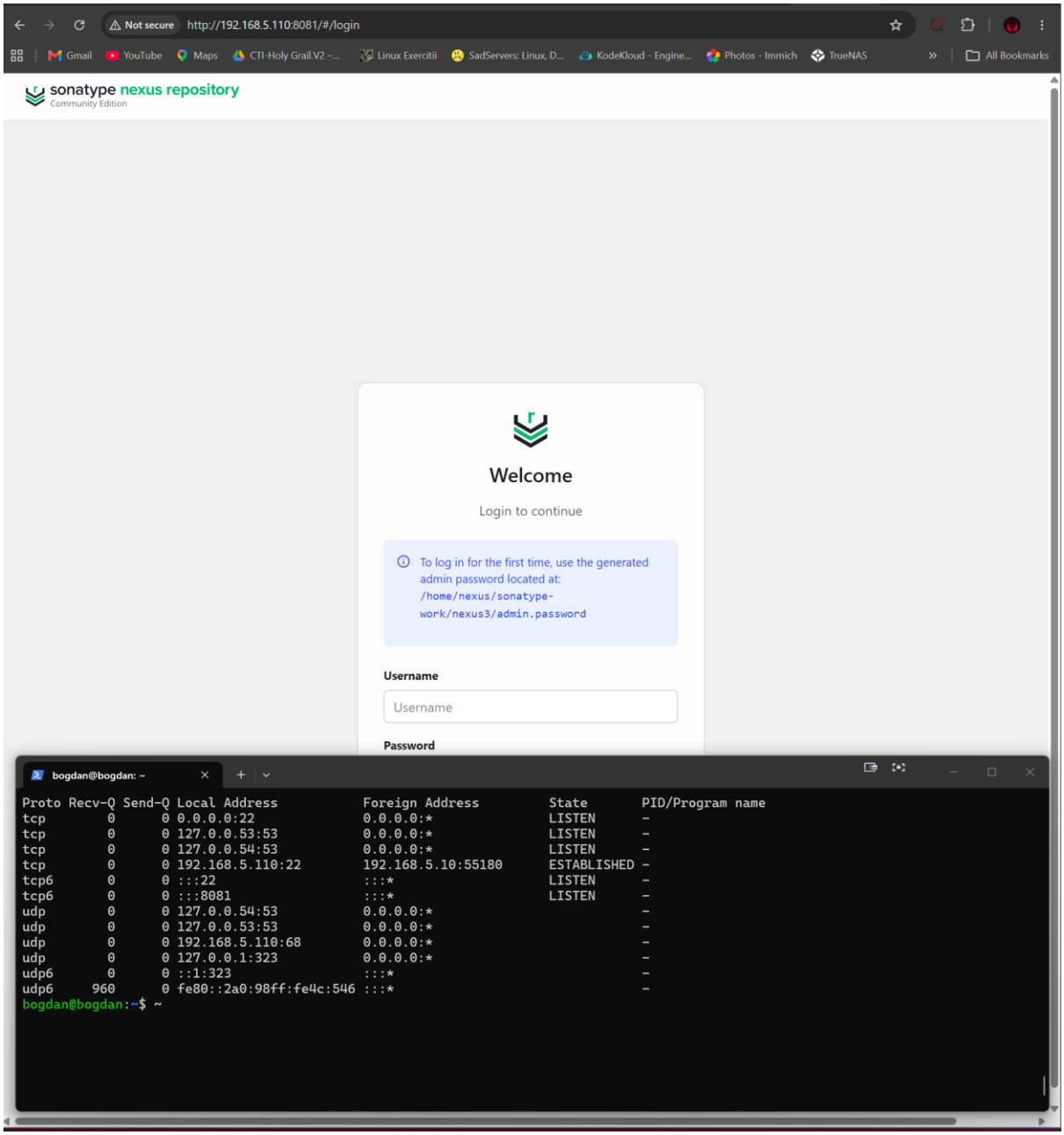
m-am conectat apoi prin ssh ca sa nu mai am inca o aplicatia in plus pe pc

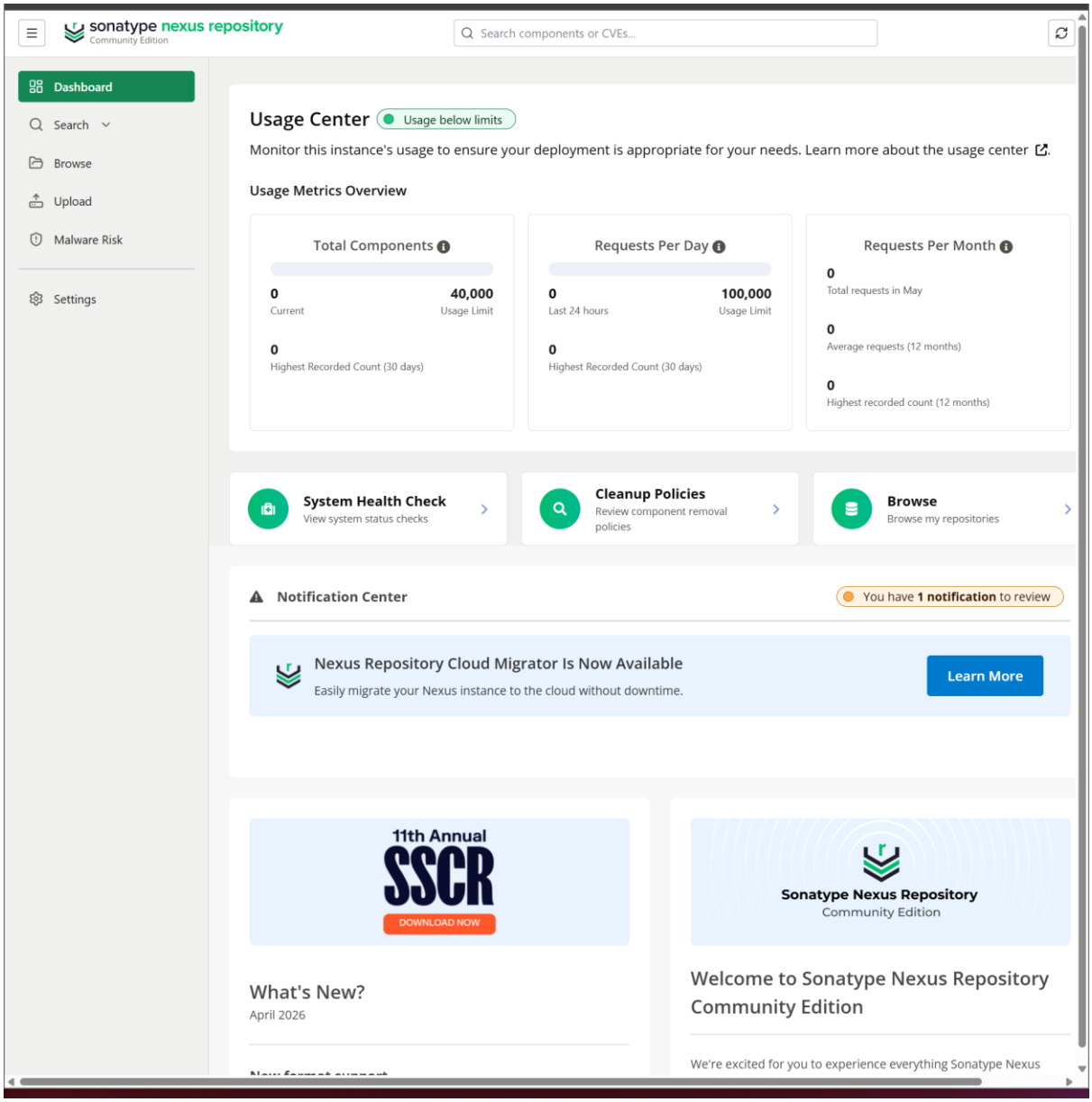




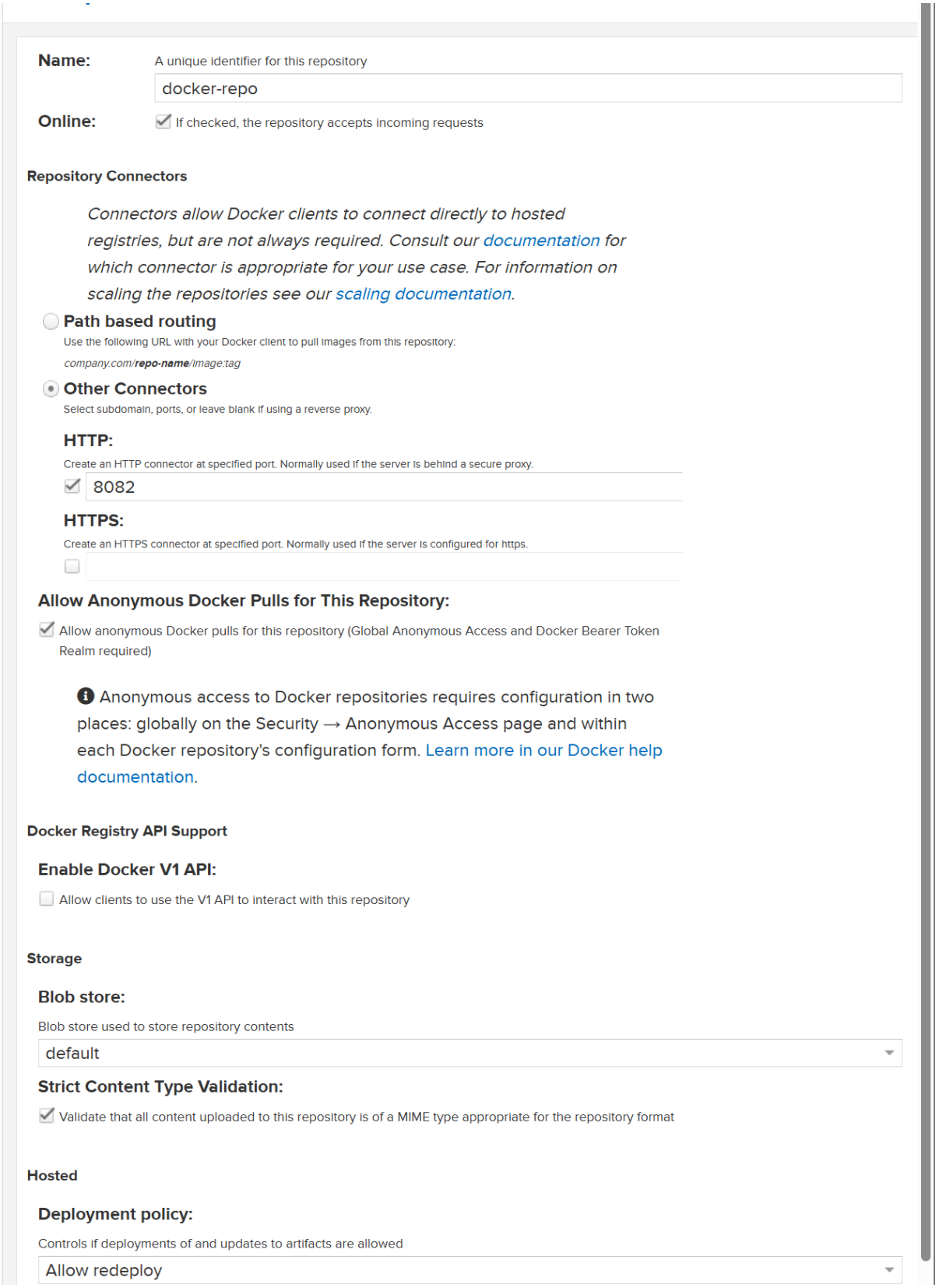

```
bogdan@192.168.5.110's password:
Welcome to Ubuntu 26.04 LTS (GNU/Linux 7.0.0-15-generic x86_64)

 * Documentation:  https://docs.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Tue May 26 08:41:49 PM UTC 2026

  System load:  0.0               Processes:             170
  Usage of /:   14.2% of 55.75GB  Users logged in:       0
  Memory usage: 2%                IPv4 address for ens3: 192.168.5.110
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

4 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Tue May 26 20:38:49 2026 from 192.168.5.10
bogdan@bogdan:~$
```

Am instalat nexus repository manager pe VM

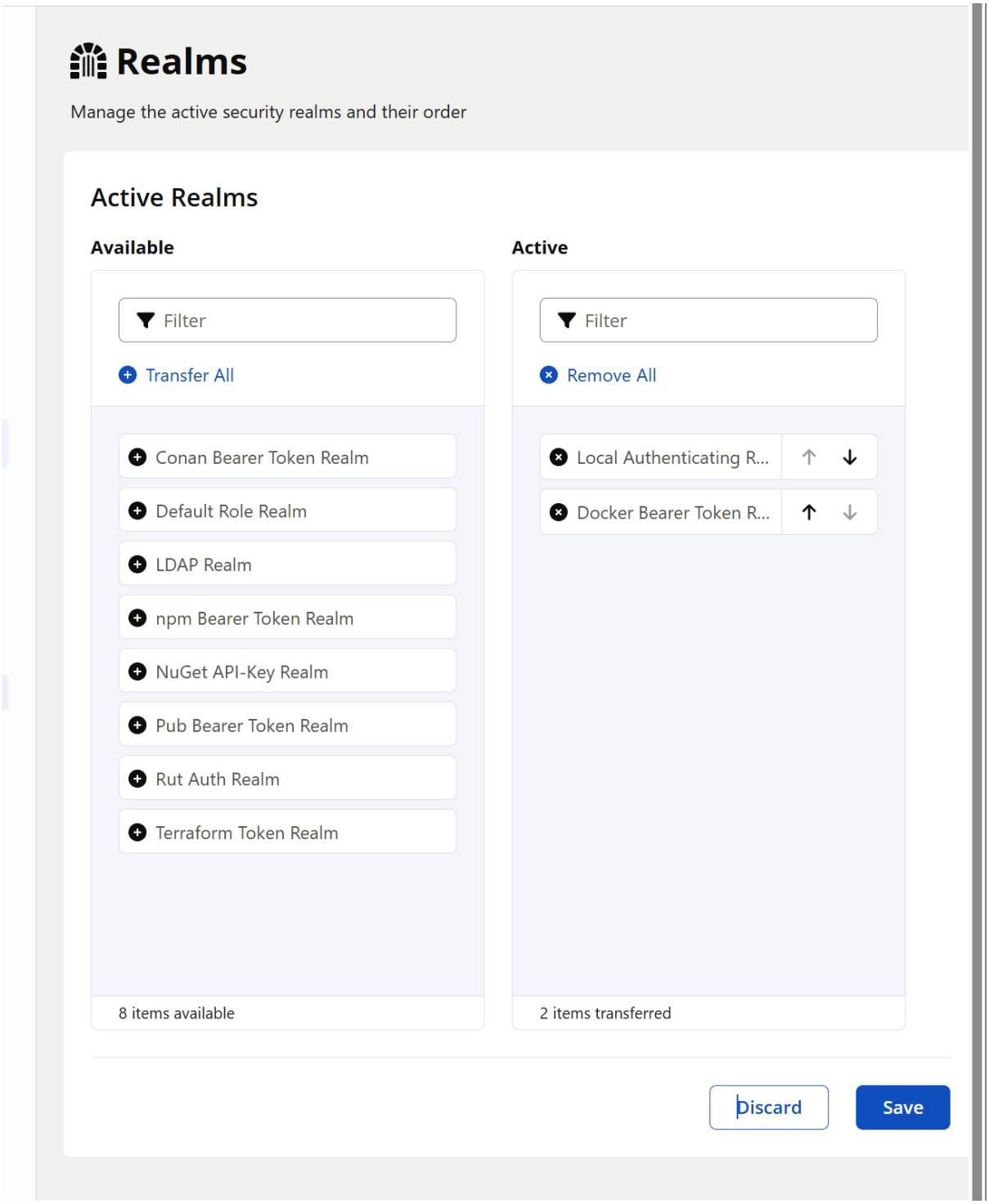




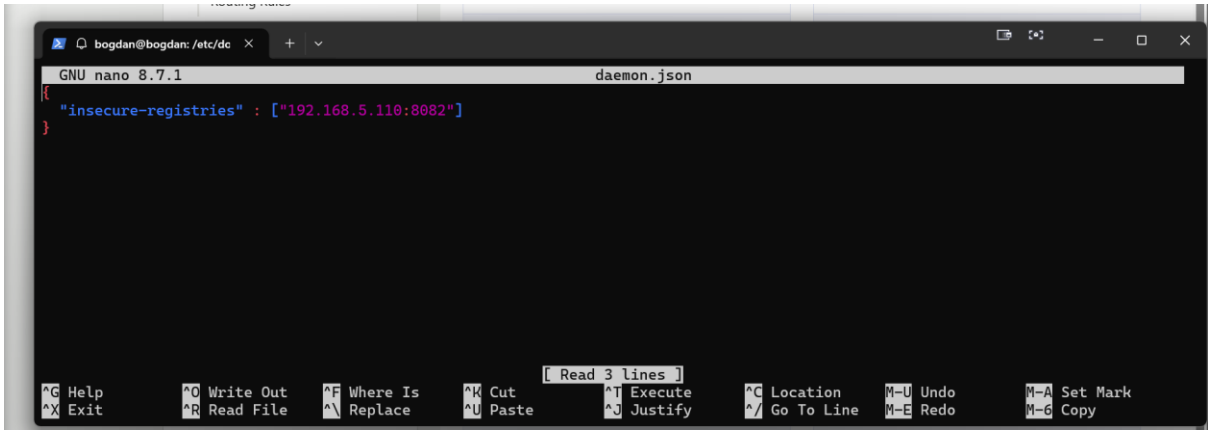
Am facut si cont pe nexus, urmatorul pas aucm este sa transformam nexusul intr un docker registry privat, adica ce se builduieste cu github actions sa se stockeze imaginile de docker (backend si frontend) in el ca mai apoi sa fie “servite” pe celalalt VM de web



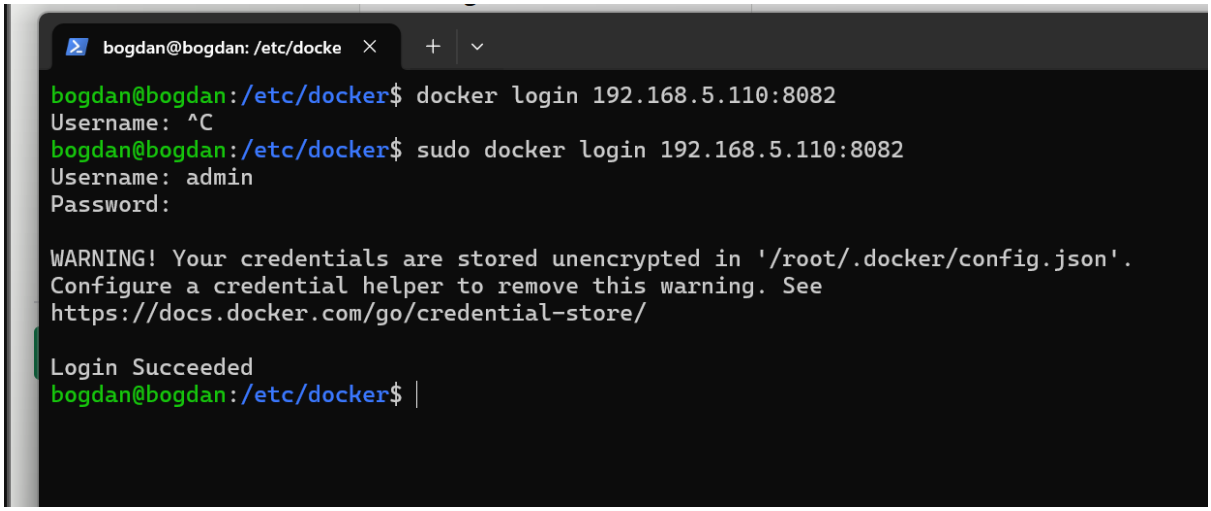
Aici: 8082 portul prin care comunica docker cu nexus, ca nexus e pe 8081, si allow anonymous docker pull ca sa poata sa traga web serveru imaginile fara sa faca login



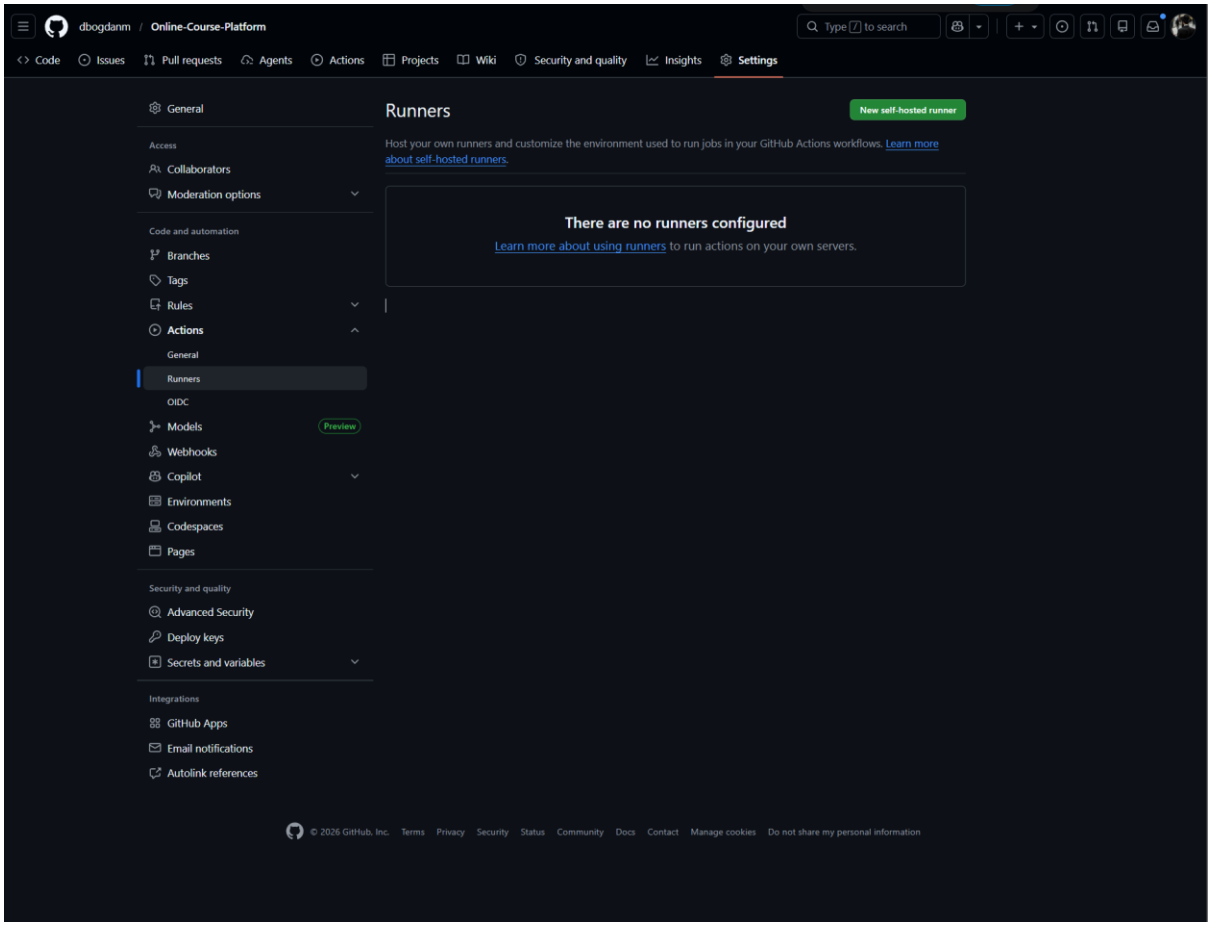
Pt ca docker push si login sa functioneze corect, nexus trb sa stie cum sa valideze crediddentialele docker



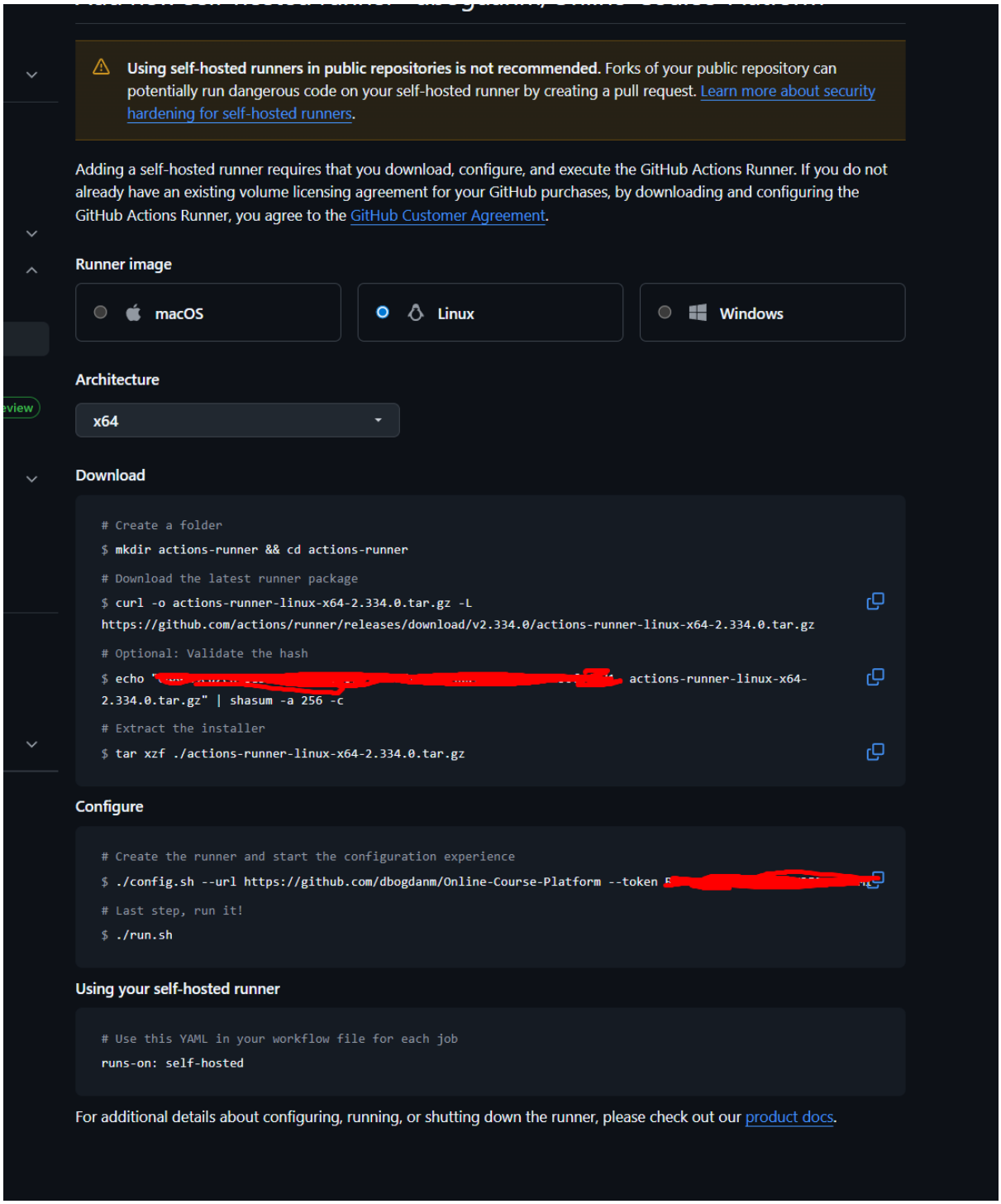
ca sa stie unde sa se conecteze prin http la nexus(ideea e ca docker de baza refuza conexiunile nesecurizate)

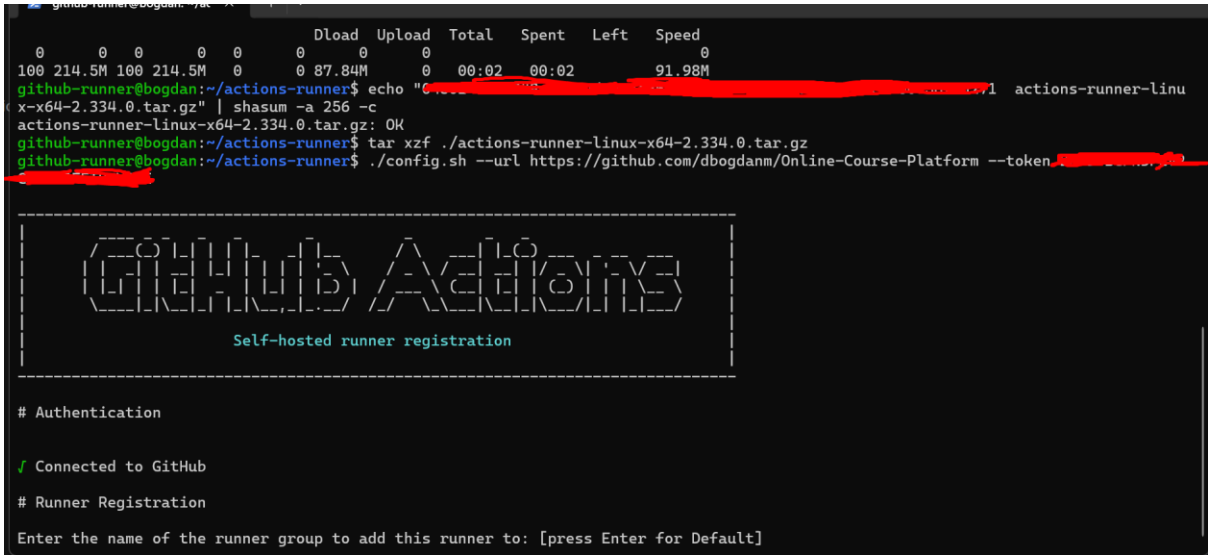


Acum trebuie sa instalam agentul de github(github runner, self hosted normal)

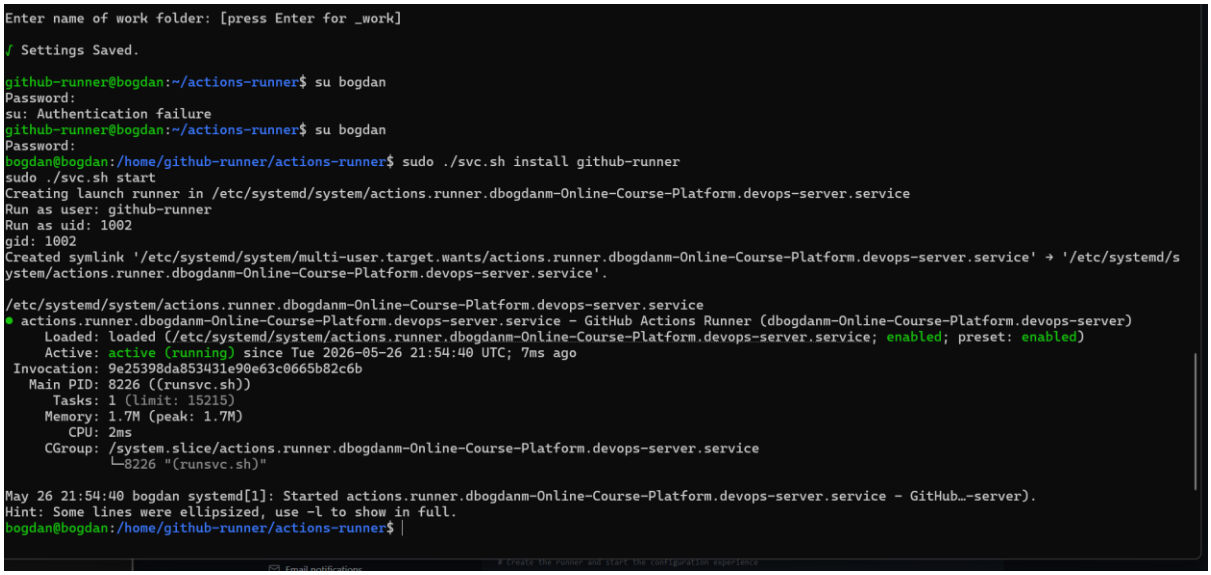


Am facut si alt user (din motive de securitate in mare parte aici), aici ne ghideaza github singur:



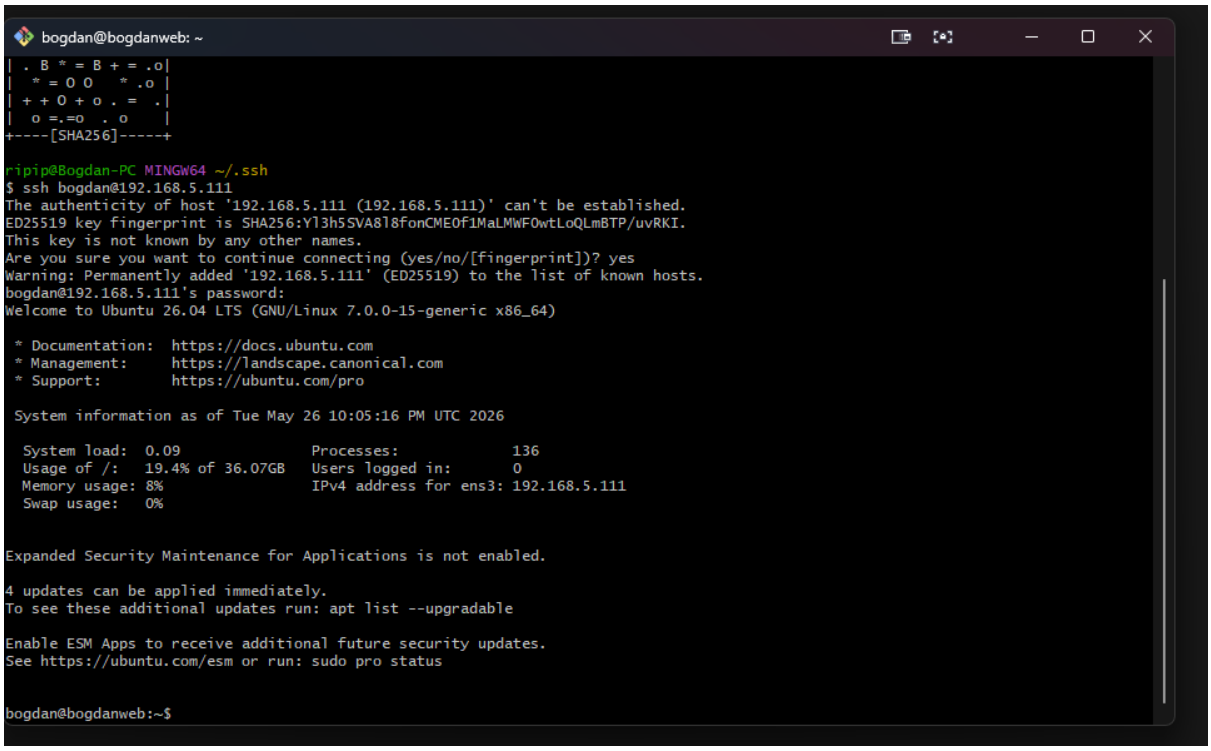


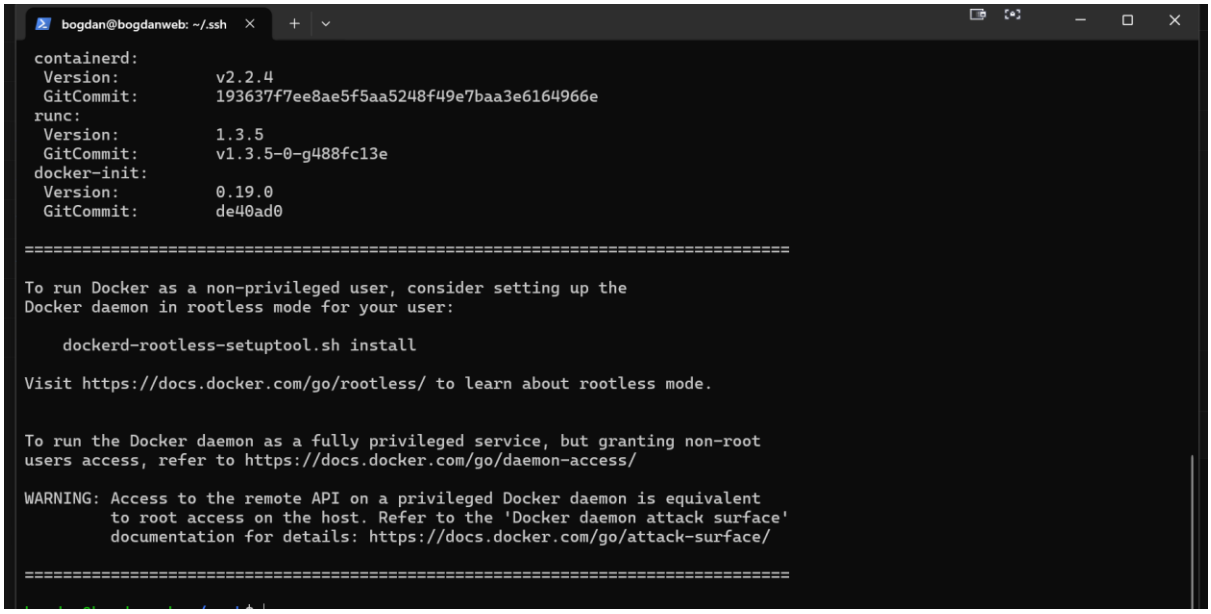
Dupa ce am terminat tutorialul de la github, trebuie ca runnerul sa fie setat sa mearga in permanenta, nu doar o singura data (cand zice la final ca dam ./run.sh simplu doar), deci il facem ca serviciu



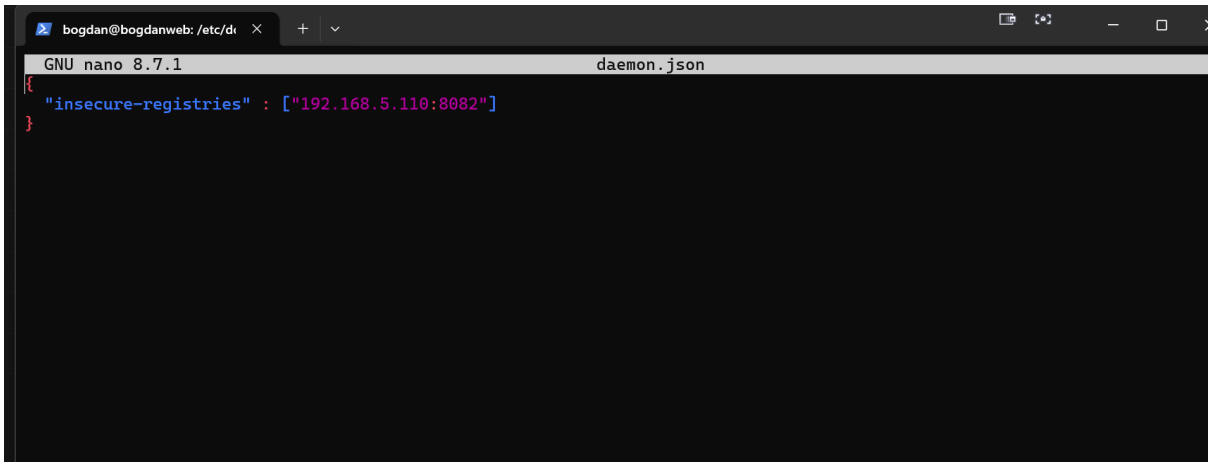
Abia acum e pregatit VM-ul de devops, ca rezumat: instalare de docker, nexus cu repository privat si runner pe githubs

Pregatirea VM-ului de web(initial configurarile prin ssh, instalari de docker si configurarea de insecure registry, la fel ca mai inainte)

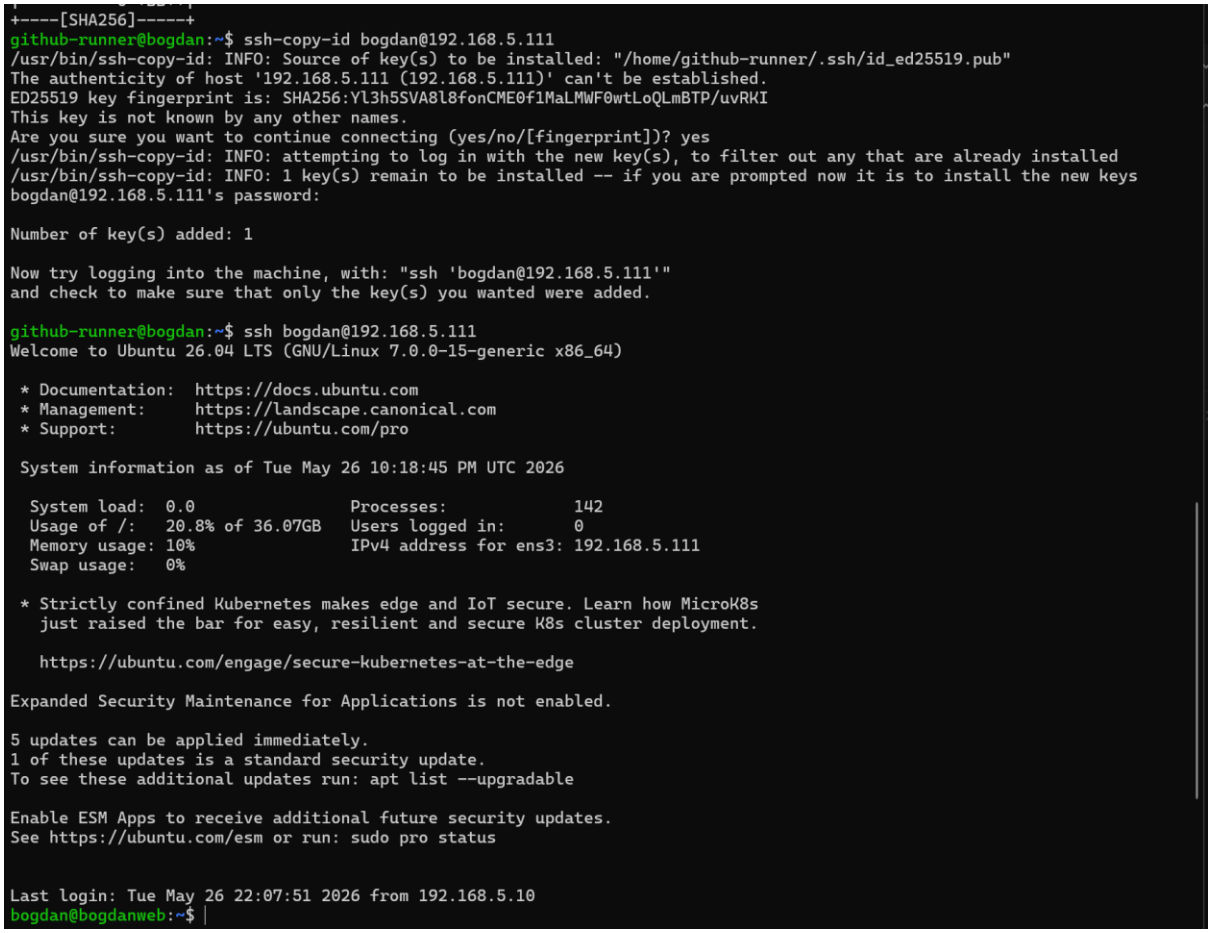




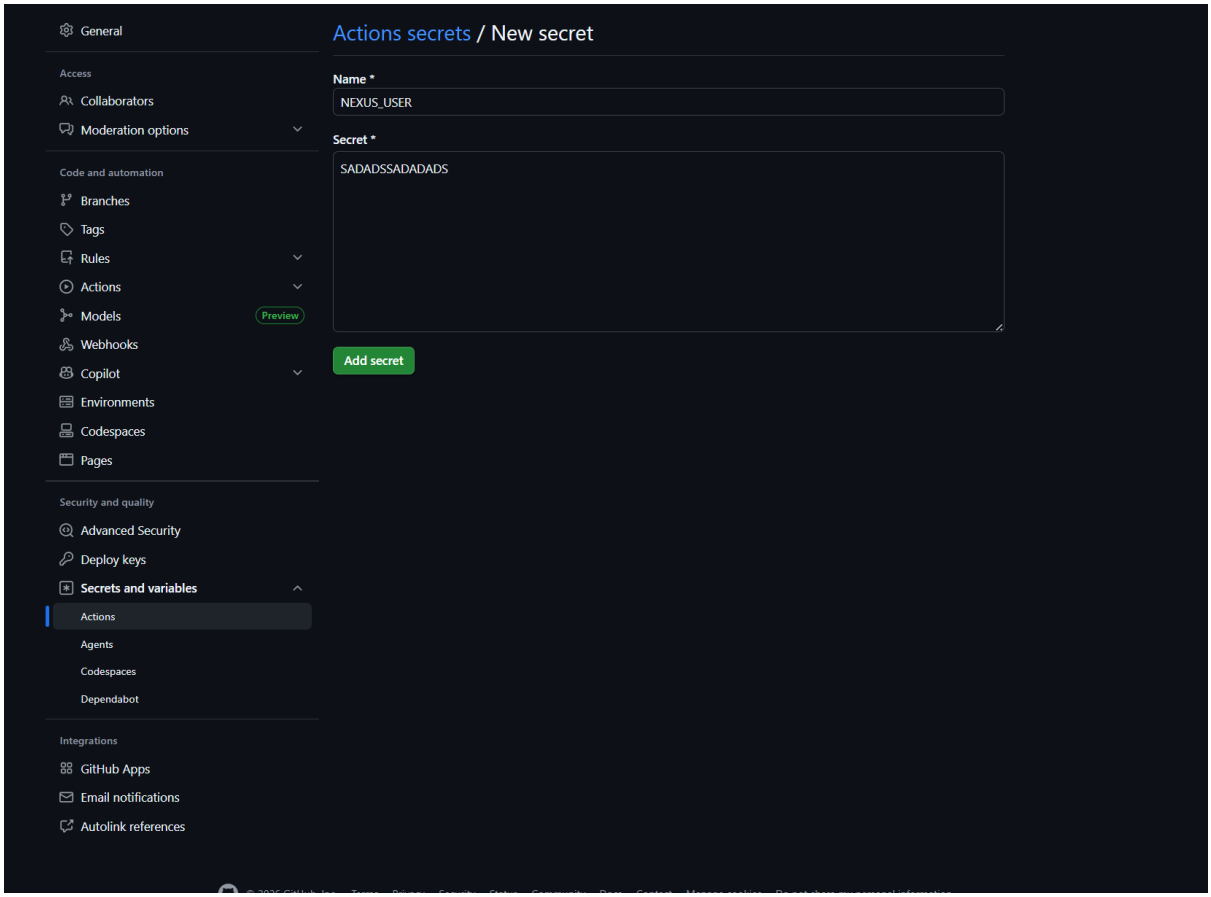
La fel ca mai inainte:



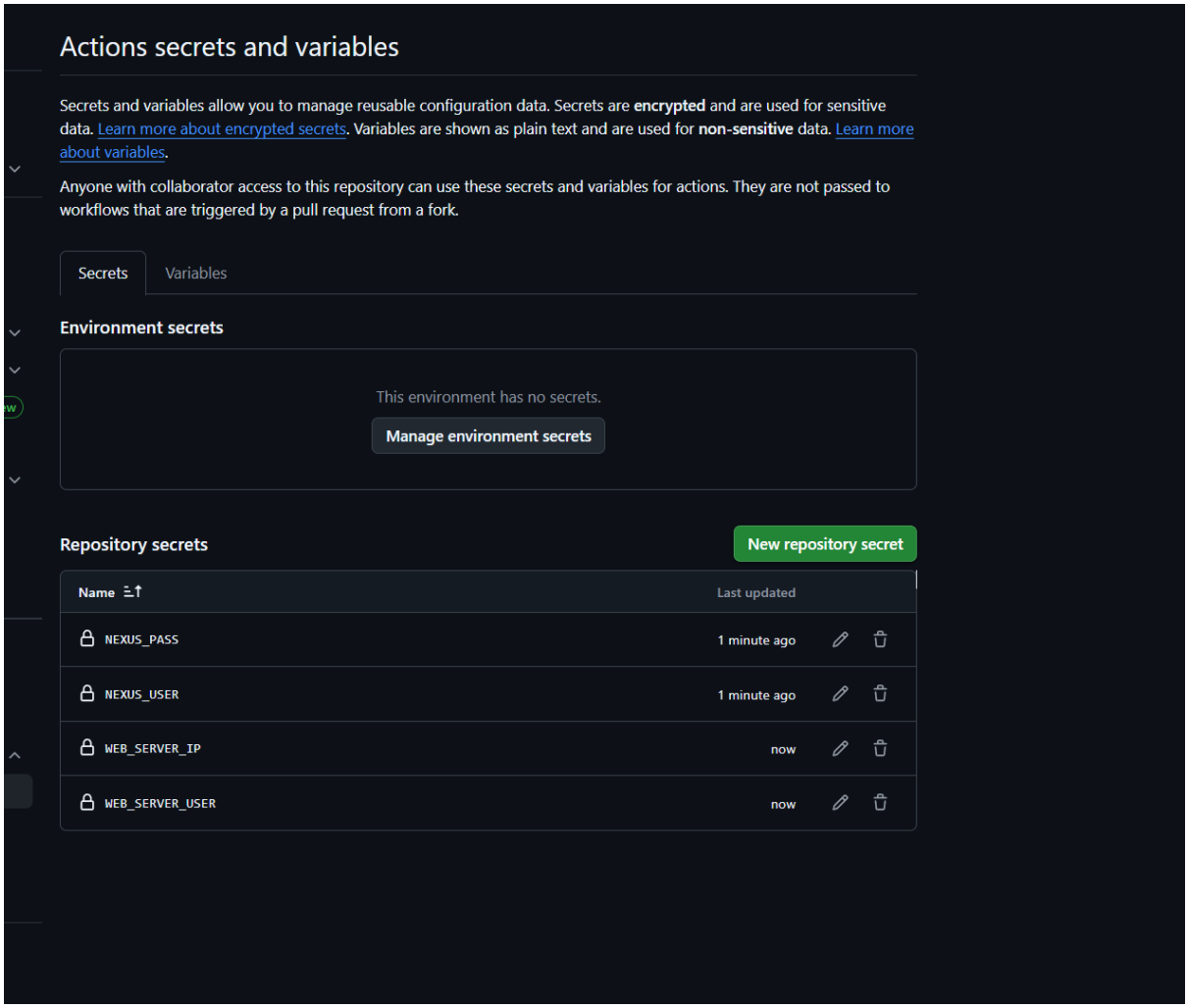
SI iar setarile sa meraga ca serviciu…



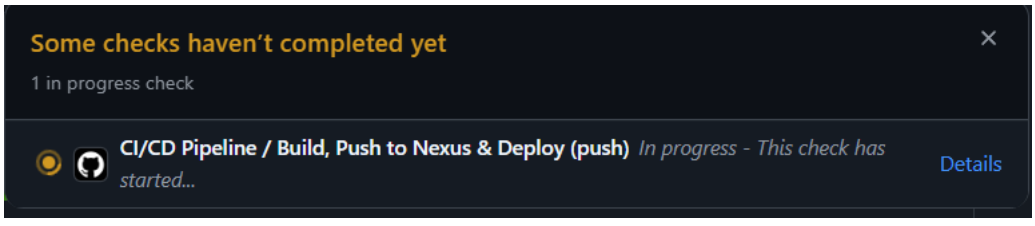
Aici am pus cheia ssh ca sa aiba acces VM-urile intre ele, adica din cel de devops sa pot da ssh in cel de web



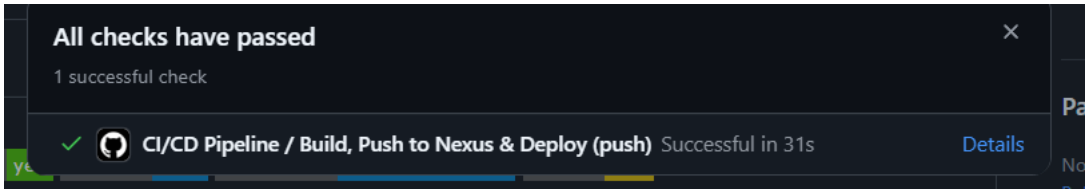
Acum vine si partea interesanta, parerea mea, ca sa facem pipelineul functional, trebuie sa mai facem cv modificari la proiect, adica sa nu lasam parole la vedere

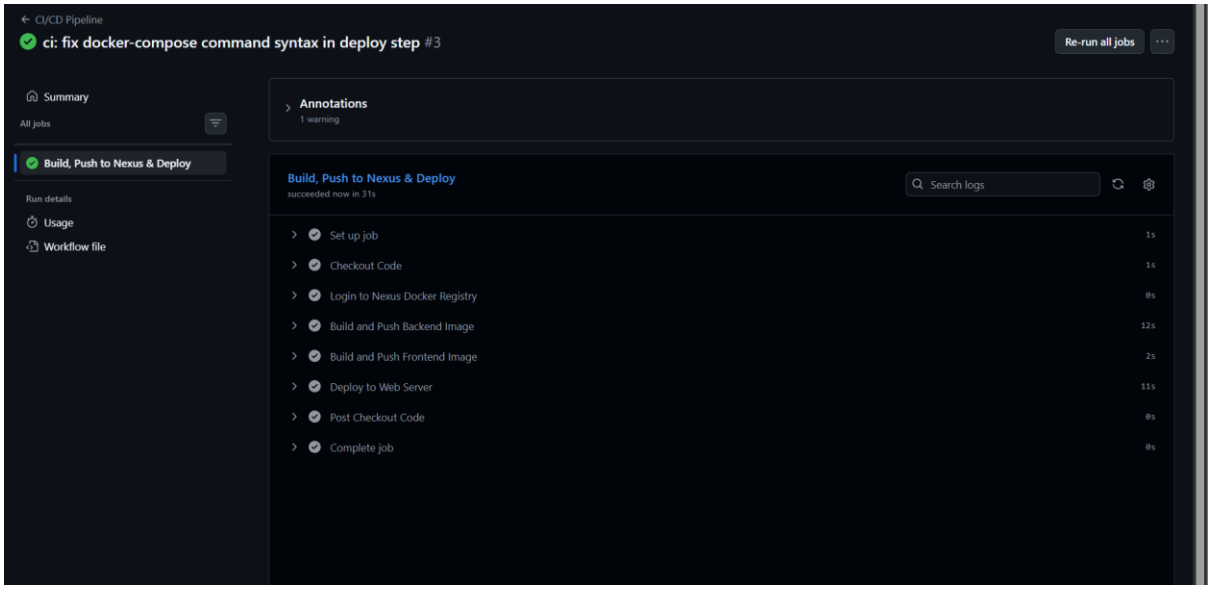


Partea mai grea acum este scrierea fisierelor deploy.yml si docker-compose.prod.yml (nu am facut eu acest lucru singur)

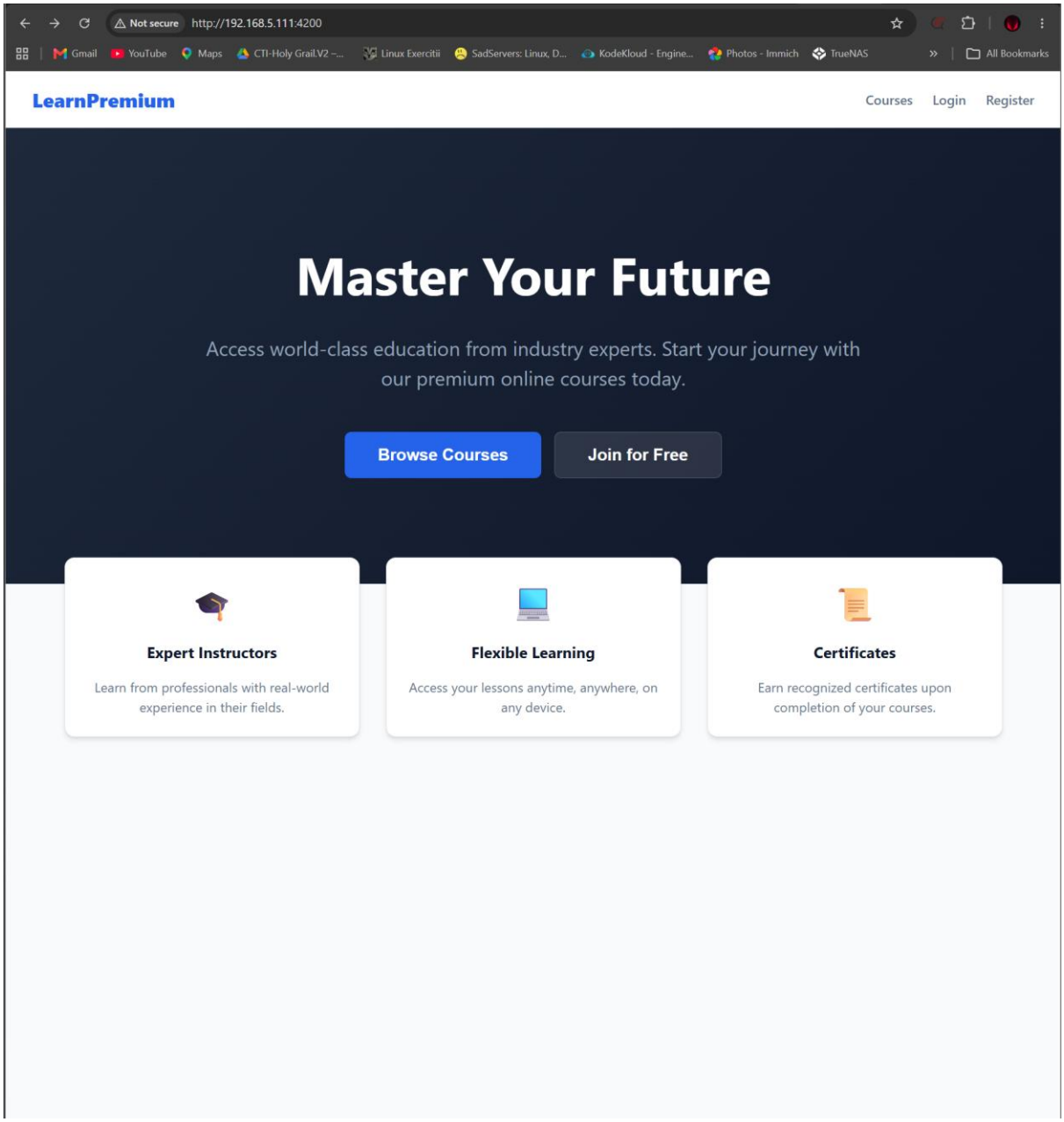


Si pushul final

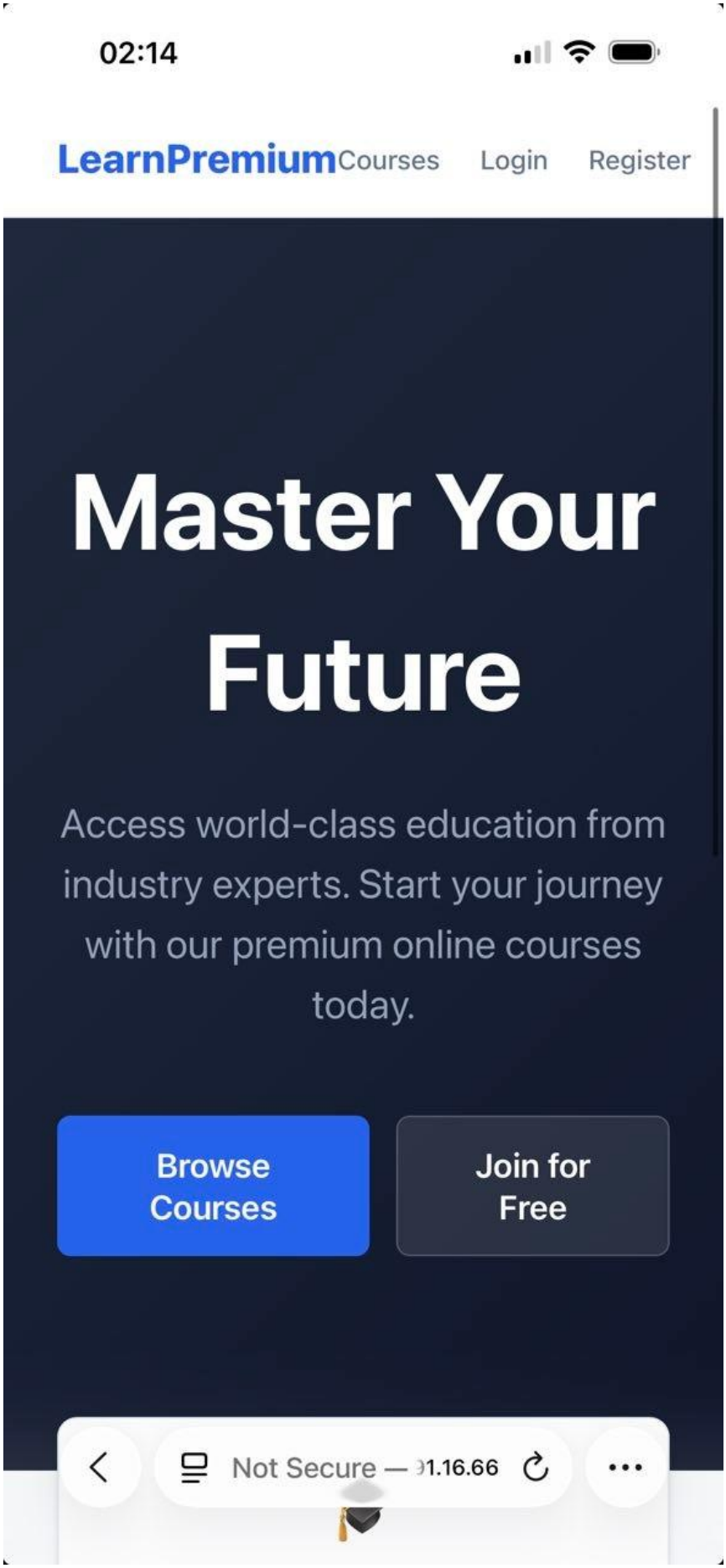




Si aplicatia finala in web browser:



BONUS: am instalat tailscale pe serverul web (ca acolo e nginx)


02:14
LearnPremium
Courses
Login
Register
Master Your Future
Access world-class education from industry experts. Start your journey with our premium online courses today.
Browse Courses
Join for Free
Not Secure — 1.16.66

Q. Cu ce ne ajuta Docker concret la deploy-ul proiectului?

A. Exemplu concret: daca lucram fara docker, trebuia sa intru pe sv de web si sa instalez manual runtimeul de .net 10, baza de date, si alte chestii, daca trebuia sa trec pe viitor la .net 11, trebuia iar sa fac actualizarile astea manual, asa cu docker am instalat doar pe el, deja venind impachetat cu .net 10 inauntru. Inca un exemplu ar mai fi evitarea de conflicte ale versionarii aplicatiilor sau toolurilor, daca copiam direct fisierele de “acasa” pe server, poate lipsea ceva librarie si nu mergea, cu docker, cand am dat docker build pe serverul de devops, codul a fost pus intr-un container care contine propriul mini sistem de operare, acel container ruleaza pe serverulde web ,exact acelasi mediu in care a fost construit

Q. De ce folosim .env în loc să scriem parolele direct în docker-compose.yml?

A. Din motive de securitate, nu se hardcodeaza parole care apoi or sa fie pushate pe git niciodata, parolele se pun in .env , iar .env se adauga in .gitignore ca sa nu ajunge pe git, ramane doar pe local, eu nu am folosit .env aici din 2 motive: primul ar fi ca am un runner pe github si e “semi-bresa” de securitate sa ai un runner private pe repo public, si al doilea este ca nu am vrut sa ma “complic”, si asa nu puteam sa pun repo-ul public si nu aveam de ce sa lucrez cu .env

Q. Ce rol are nginx reverse proxy în fața containerelor voastre (de ce nu expuneți direct portul aplicației la internet)?

A. Se foloseste pt a nu expune direct serverele de aplicatie la internet, asa mai ai un strat in plus de securitate, terminare de SSL si load balancing

Q. Ce ați învățat azi nou și ce vi s-a părut cel mai interesant?

A. Am invatat ca nu stiu (inca!) sa scriu docker files sau fisiere de configurare YAML care sunt vitale pt pipeline, legat de partea non-coding a proiectului sa zic asa, majoritatea chestiilor le invatasem singur deja(mai putin asta cu scrisul de fisiere de configurare), nu stiam precis niste cai de configurare prin nexus,cum se cheama de exemplu calea precisa din care poti porni nexusul, mi se pare foarte interesant felul in care functioneaza docker, modularizeaza totul si face chiar error proof aplicatia