

Kako se povezati Jenkins sa svojim github-nalogom

Generisaćemo token na našem git hab nalogom I od ponudjenih opcija uzećemo samo

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input type="checkbox"/> repo	Full control of private repositories
<input type="checkbox"/> repo:status	Access commit status
<input type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input type="checkbox"/> repo:invite	Access repository invitations
<input type="checkbox"/> security_events	Read and write security events
<hr/>	
<input type="checkbox"/> workflow	Update GitHub Action workflows
<hr/>	
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Regis
<input type="checkbox"/> read:packages	Download packages from GitHub Package

Vratim se sada u moj Jenkins i idem u Manage Jenkins pa u Configure System.

Idem dole sve do odseka sa Githubom.

+ New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Build Queue

No builds in the queue.

Build Executor Status

built-in node (0 of 2 executors busy)

Manage Jenkins

Warnings have been published for the following currently installed components:

Build Pipeline Plugin 2.0.1:
Stored XSS vulnerability (no fix available)
No fixes for these issues are available. It is recommended that you review the security advisories.

System Configuration

- System**
Configure global settings and paths.
- Tools**
Configure tools, their automatic installers.
- Appearance**
Configure the look and feel.

Security

Kada instalirate predložene pluginove vi ste instalirali i plugin za github

Dashboard > Manage Jenkins > System > GitHub

Default notification URL

Notification URL ?

Default

GitHub

GitHub Servers ?

Add GitHub Server

Advanced Edited

GitHub API usage

Github API usage rate limiting strategy ?

Normalize API requests

GitHub Enterprise Servers

Add

Save Apply

Kada dodjemo da izaberemo vezu sa našim nalogom na github-u izabracemo skriveni tekst gde ćemo token iskoristiti za povezivanjem sa našim nalogom.

Default

GitHub

GitHub Servers ?


≡ GitHub Server ?

Name ?

API URL ?

Credentials ?

[+ Add](#)

 Jenkins

Manage hooks

Advanced ▾

Test connection

O vde moram da ubacim nas token, pod skrivenim tekstom

Browser tabs: Sign in [Jenkins], Role-based Authoriza, System [Jenkins]

Browser address bar: localhost:8080/manage/configure

Navigation: Dashboard > Manage Jenkins > System

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain: Global credentials (unrestricted)

Kind: Username with password

- Username with password
- GitHub App
- SSH Username with private key
- Secret file
- Secret text**
- Certificate

Treat username as secret ?

Password ?

ID ?

Description ?

The image shows a web browser window at localhost:8080/manage/configure. The main content is a modal dialog titled "Jenkins Credentials Provider: Jenkins" with the sub-header "Add Credentials". The form contains the following fields:

- Domain:** A dropdown menu with "Global credentials (unrestricted)" selected.
- Kind:** A dropdown menu with "Secret text" selected.
- Scope:** A text input field containing "Global (Jenkins, nodes, items, all child items, etc)".
- Secret:** A text input field filled with dots, representing a hidden token. A red callout bubble points to this field with the text "token kao skriveni tekst".
- ID:** A text input field containing "Github-Djole80". A red callout bubble points to this field with the text "naziv koji mi želimo".
- Description:** An empty text input field.

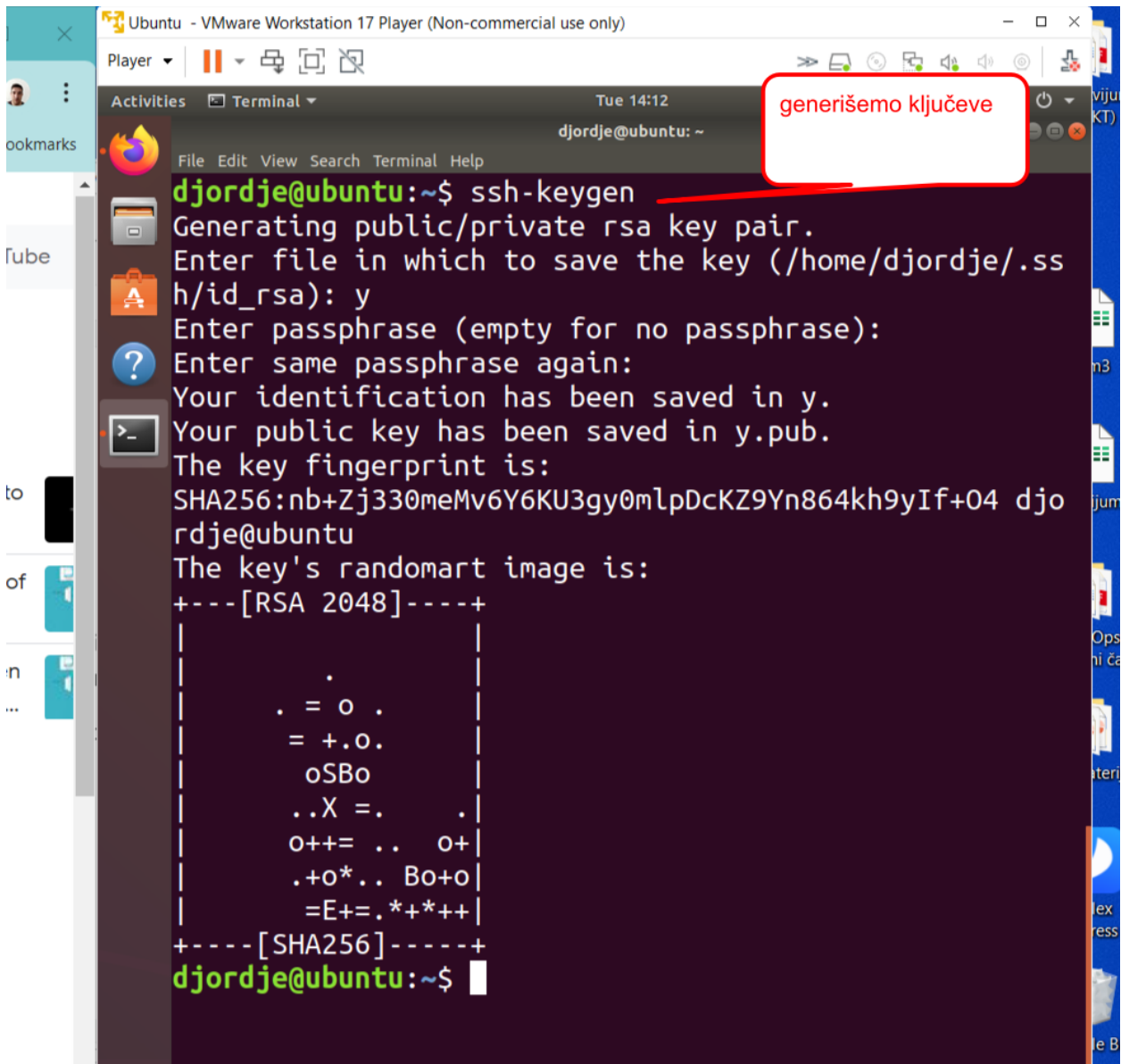
At the bottom of the dialog are two buttons: "Cancel" and "Add".

Proverimo sada podešavanjem na našu konekciju da li ona rati . To meožemo pritiskom na test conection taster.

Sledeće što ćemo uraditi je da se povežemo sa SSH ključem

Na taj način moći ćemo da cloniramo repozitorijum na Jenkinsu lokalnom prostoru. Takođe gde god moj Github ima pristup imaće i Jenkins.

Da vidimo kako to možemo, moramo prvo u terminalu da generišemo privatni i javni ključ ssh.

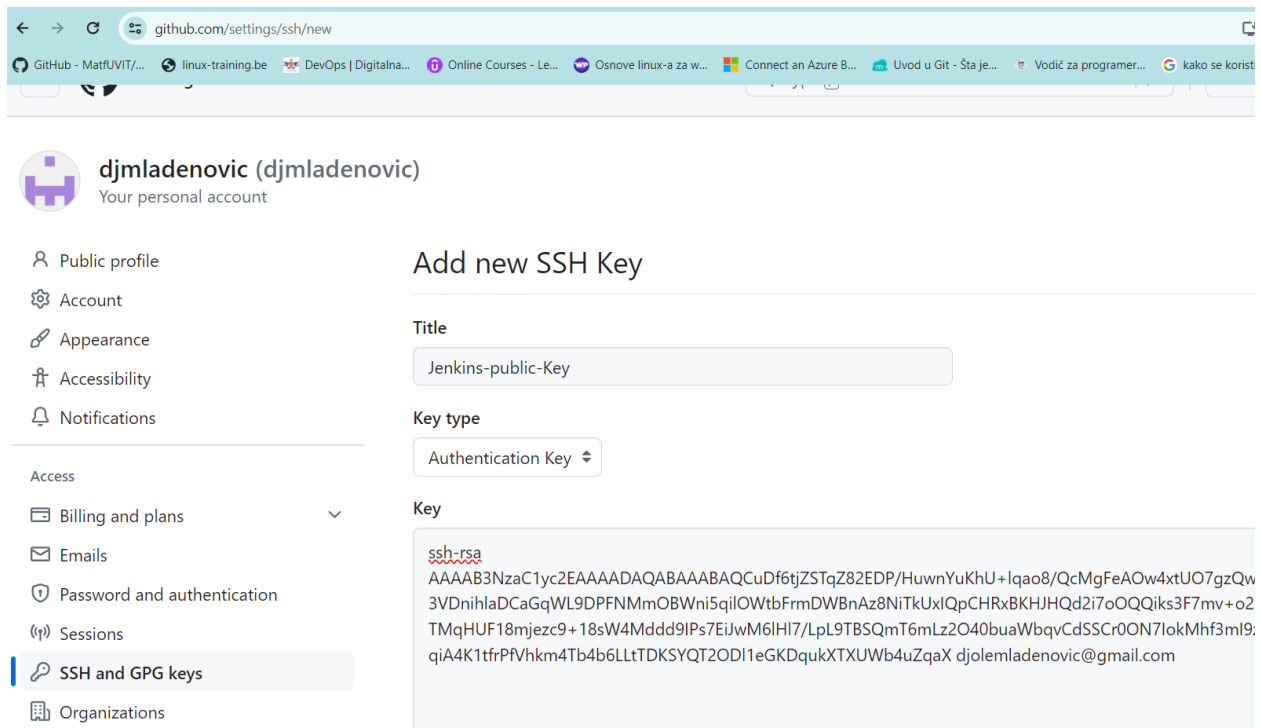


```
Ubuntu - VMware Workstation 17 Player (Non-commercial use only)
Tue 14:12
djordje@ubuntu: ~
generišemo ključeve
ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/djordje/.ssh/id_rsa): y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in y.
Your public key has been saved in y.pub.
The key fingerprint is:
SHA256:nb+Zj330meMv6Y6KU3gy0mlpDcKZ9Yn864kh9yIf+04 djordje@ubuntu
The key's randomart image is:
+----[RSA 2048]-----+
|
|  .
|  = o .
|  = +.o.
|  oSBo
|  ..X =.
|  o++= .. o+
|  .+o*.. Bo+o
|  =E+=.*+*++
+----[SHA256]-----+
djordje@ubuntu:~$
```

Da vidimo sada javni (public key) ključ koji je

```
rw-rw-r-- 1 djordje80 djordje80 4096 Feb 10
-rw-rw-r-- 4 djordje80 djordje80 4096 Mar 20
-rw-rw-r-- 1 djordje80 djordje80 199 Apr 29
jordje@ubuntu:/home/djordje80$ cd .ssh
jordje@ubuntu:/home/djordje80/.ssh$ ls
id_ed25519 id_ed25519.pub id_rsa id_rsa.pub
jordje@ubuntu:/home/djordje80/.ssh$ id_rsa.pub
id_rsa.pub: command not found
jordje@ubuntu:/home/djordje80/.ssh$ cat id_rsa
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQACuDF6tjZS`
qWL9DPFNMmOBWni5qilOWtbFrmDWBnAz8NiTkUxIQpCHRxl
7EiJwM6lHl7/LpL9TBSQmT6mLz2040buaWbqvCdSSCr0ON`
kXTXUWb4uZqaX djolemladenovic@gmail.com
jordje@ubuntu:/home/djordje80/.ssh$
```

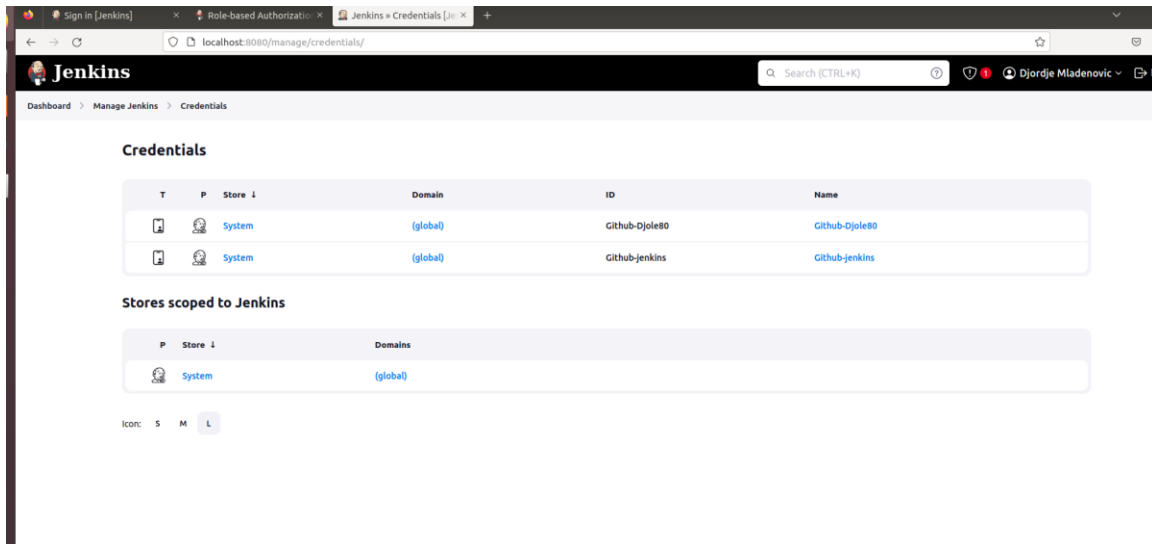
Sada idem na moj Github nalog da postavim taj javni ključ.




Idemo sada da iskopiramo naš private key u Jenkinsu gde treba.


```
...-BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEAA3hARovKwQJCoEGXVoVF5tflVr3xl0N/Euv2vBl0eRZttWDPu
let8oqHBBrwB39C5Ra/wXI7wjvD9No+a/m5ochrFYyBiUttiWBoXEsbNtv06qvv
;5fQc5gybydJouYccW4o2oaST5vS1ReFrftdCpiVh8XAPmgIamFUk1YFDpggBv
:1DnbaSHYMAiik8CGXr00Y0w59AX0UBkbuXQ6LLk4A+NiuQtxqqkli53etJpJZ
3euetmpEMfL1J8aT3kHz0u6aA4Gn5KzdL3d0NArNc0/7SS3EI/9croZEWfYx2b
1EHJIHd3IRLHoSC78N0AefDw6AyIG+itVybQIDAQABAoIBAQCW8BEs0wzeNfUm
:/HyQ9ejuotkn2gEL2XuySxMSIx6bBJztXFLRodG++/shcMDJGCPp6AK2BzqHZ
oPecSlpNa0pPZ/XdesYu1Mnnh3B3cn84nsuqtBwqFi9hwjCZGxQGPbxjHMLidy
/Zm1sn+BbkllKhML2Caw0EwGgZeGG00oL67vkEsyWobv701gd6/AfXLkU3Pu17
7MadHkl /02fh08FFeDnRix4o6/vA7s1ITVz7hM+1 /G7b9/7n8r9hEwRr11e7wt+
```

Sada u Jenkinsu idemo u **Manage Jenkins** pa u deo o sigurnosti pronađemo **Manage Credentials**



System

Domain ↓	Description
 Global credentials (unrestricted) Add credentials	Credentials that should be available irrespective of domain specification to requirements matching.

Icon: S M L

izabraću ovo

localhost:8080/manage/credentials/store/system/domain/_/newCredentials

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

SSH Username with private key

Scope ?
Global (Jenkins, nodes, items, all child items, etc)

ID ?
Private-key-github

Description ?

Username

Treat username as secret ?

Private Key
 Enter directly

Key

```
-----BEGIN PRIVATE KEY-----
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoGBAL...
cBz1DnbaSHYMAiik8CGXr80Y8w59AX0UBkbuX06LLk4A+NiU0tXqkLi53et3pJZ
pXBeuetmpEMfL1J8aT3khZ0u6aA4Gn5KzdL3d0NArNc0/7SS3E1/9cr0ZEWfYx2b
-----END PRIVATE KEY-----
```

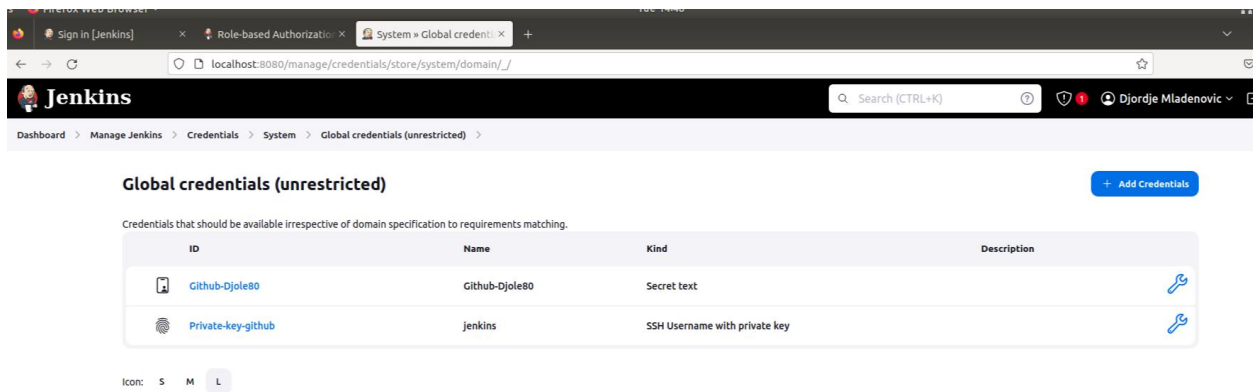
Create

ssh private key želimo

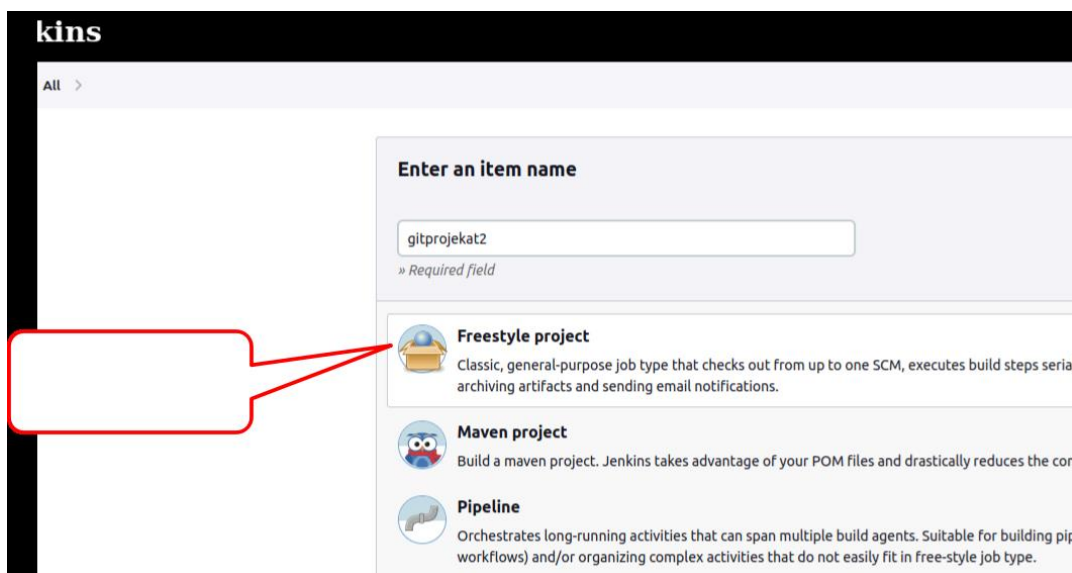
naziv po izboru

postavimo opciju za direktno unošenje privatnog ključa, tu ga iskopiramo!!!!

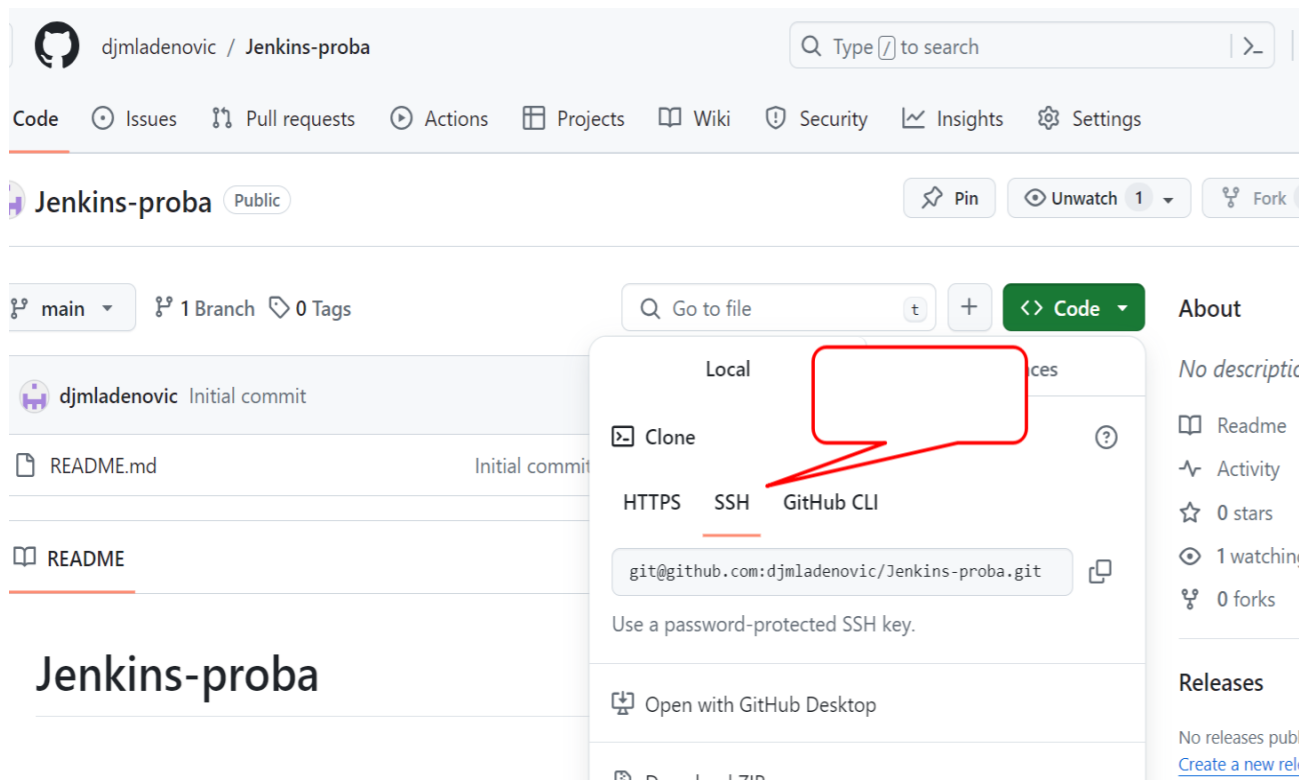
I dobili smo željene konekcije sa github-om



On bi sada trebao da može da klonira repositorijum uz pomoć ssh ključa.



Koristicu na githubu neki prosti primer gde imam samo jedan readme file. Uzeću njegovu ssh konekciju i to ću iskopirati.



U postavkama iz priloženog obratite pažnju na branch koji postavljate. Postavljen je master koji morate promeniti na main. Naravno zavisi gde je željeni program koji želite da preuzmete na Jenkinsu tj. Na kojij grani se nalazi željena verzija .



guration

repositories

Repository URL ?
https://github.com/djmladenovic/Jenkins-proba.git

Credentials ?
jenkins (privatni kljuc)
- none -
jenkins (privatni kljuc)
Ubuntu (Jenkins)

Advanced

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?
*/main

Add Branch

Repository browser ?
(Auto)

predhodno definisam privatni ključ

screenrec
The screenshot is copied to your clipboard. Press CTRL+V to paste it.

Dodali smo i opciju (* * * * *) sa kojom možemo pratimo promene koje se dešavaju na GitHubu i njih bildujemo. Učestalost zavisi od nas i našeg podešavanja.

localhost:8080/job/gitprojekat2/configure

ation

Build Triggers

- Trigger builds remotely (e.g., from scripts) ?
- Build after other projects are built ?
- Build periodically ?
- GitHub Branches
- GitHub Pull Requests ?
- GitHub hook trigger for GITScm polling ?
- Poll SCM ?

Schedule ?
*/5 * * * *

⚠ Spread load evenly by using 'H/5 * * * *' rather than '*/5 * * * *'
Would last have run at Sunday, May 26, 2024 at 2:45:11 AM Pacific Daylight Time; would next run at Sunday, May 26, 2024 at 2:50:11 AM Pacific Daylight Time.

- Ignore post-commit hooks ?

Build Environment

- Delete workspace before build starts
- Use secret text(s) or file(s) ?

učestalost koda pregleda sa githuba

I kao što vidimo bild radi.

Console Output

Console Output

```
Started by user Djordje Mladenovic
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/gitprojekat2
The recommended git tool is: NONE
using credential kljuc
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/gitprojekat2/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/djmladenovic/Jenkins-proba.git # timeout=10
Fetching upstream changes from https://github.com/djmladenovic/Jenkins-proba.git
> git --version # timeout=10
> git --version # 'git version 2.17.1'
using GIT_SSH to set credentials privatni kljuc
Verifying host key using known hosts file
You're using 'Known hosts file' strategy to verify ssh host keys, but your known_hosts file does not exist, please go to 'Manage Jenkins' -> 'Security' -> 'Git Host Key Verification Configuration' and configure host key verification.
> git fetch --tags --progress -- https://github.com/djmladenovic/Jenkins-proba.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 3747524ed59d77092a1d1a893ba91f98788595f3 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 3747524ed59d77092a1d1a893ba91f98788595f3 # timeout=10
Commit message: "Add files via upload"
> git rev-list --no-walk 3747524ed59d77092a1d1a893ba91f98788595f3 # timeout=10
Finished: SUCCESS
```