How to Install and Configure GitLab on Ubuntu 20.04

GitLab is an open-source DevOps lifecycle tool used to host and manage Git repositories. It is written in Ruby and offers a lot of features including, wiki, issue management, code review, monitoring, and continuous integration and deployment. You can host it within your own infrastructure and deploy an internal repository for your development team. GitLab is available in three editions, Community Edition (CE), Enterprise Edition (EE), and a GitLab-hosted version.

On this page

In this tutorial, we will show you how to install GitLab CE on Ubuntu 20.04 server.

Prerequisites

- A server running Ubuntu 20.04.
- A valid domain name pointed with your VPS.
- A root password is setup on your server.

Getting Started

First, you will need to update your system packages to the latest version. You can update them with the following command:

```
apt-get update -y
apt-get upgrade -y
```

Once your system is updated, install other required packages with the following command:

```
apt-get install apt-transport-https gnupg2 curl -y
```

Once all the required packages are installed, you can proceed to the next step.

Install GitLab CE

By default, GitLab is not available in the Ubuntu 20.04 default repository. So you will need to add GitLab official repository in your system.

First, download and add the GitLab GPG key with the following command:

```
curl -sL https://packages.gitlab.com/gitlab/gitlab-ce/gpgkey
| apt-key add -
```

Next, add the GitLab repository in APT with the following command:

```
nano /etc/apt/sources.list.d/gitlab.list
```

Add the following lines:

```
deb https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/ bionic main
deb-src https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu/ bionic
main
```

Save and close the file when you are finished. Then, update the repository and install GitLab CE with the following command:

```
apt-get update -y
apt-get install gitlab-ce -y
```

Once the GitLab has been installed successfully, you should get the following output:

```
It looks like GitLab has not been configured yet; skipping the upgra
de script.

       *.                  *.
      ***                 ***
     *****               *****
    .******             *******
    ********            ********
   ,,,,,,,,,***********,,,,,,,,,
  ,,,,,,,,,,,*********,,,,,,,,,,,
  .,,,,,,,,,,,*******,,,,,,,,,,,
      ,,,,,,,,,*****,,,,,,,,,.
         ,,,,,,,****,,,,,,
            .,,,***,,,,
                ,*,.



     _______ __  __          __
    / ____(_) /_/ /   ____ _/ /_
   / / __/ / __/ /   / __ `/ __ \
  / /_/ / / /_/ /___/ /_/ / /_/ /
  \____/_/\__/_____/\__,_/_.___/


Thank you for installing GitLab!
GitLab was unable to detect a valid hostname for your instance.
Please configure a URL for your GitLab instance by setting `external
_url`
configuration in /etc/gitlab/gitlab.rb file.
Then, you can start your GitLab instance by running the following co
mmand:
  sudo gitlab-ctl reconfigure

For a comprehensive list of configuration options please see the Omn
ibus GitLab readme
https://gitlab.com/gitlab-org/omnibus-gitlab/blob/master/README.md
```

At this point, GitLab is installed in your system. You can now proceed to the next step.

Configure GitLab

Next, you will need to define the URL of GitLab. You can define it inside /etc/gitlab/gitlab.rb file.

```
nano /etc/gitlab/gitlab.rb
```

Change the following line with your valid hostname or domain name:

```
external_url 'http://gitlab.linuxbuz.com'
```

Save and close the file. Then, reconfigure the GitLab by running the following command:

```
gitlab-ctl reconfigure
```

Once the GitLab is configured successfully, you should get the following output:

```
Recipe: gitlab::puma
  * runit_service[puma] action restart (up to date)
  * runit_service[puma] action restart (up to date)
Recipe: gitlab::sidekiq-cluster
  * runit_service[sidekiq] action restart (up to date)
Recipe: gitlab::gitlab-rails
  * execute[clear the gitlab-rails cache] action run
    - execute /opt/gitlab/bin/gitlab-rake cache:clear
Recipe: nginx::enable
  * runit_service[nginx] action restart (up to date)
Recipe: monitoring::grafana
  * runit_service[grafana] action restart (up to date)

Running handlers:
Running handlers complete
Chef Client finished, 12/767 resources updated in 01 minutes 20 seco
nds
gitlab Reconfigured!
```

You can now check the status of GitLab using the following command:

```
gitlab-ctl status
```

You should get the following output:

```
down: alertmanager: 1s, normally up, want up; run: log: (pid 12973)
142s
run: gitaly: (pid 14216) 17s; run: log: (pid 12352) 286s
run: gitlab-exporter: (pid 14165) 19s; run: log: (pid 12885) 161s
run: gitlab-workhorse: (pid 14152) 19s; run: log: (pid 12767) 185s
run: grafana: (pid 14252) 15s; run: log: (pid 13855) 54s
run: logrotate: (pid 12818) 174s; run: log: (pid 12826) 173s
```

```
run: nginx: (pid 12798) 180s; run: log: (pid 12807) 179s
run: node-exporter: (pid 14160) 19s; run: log: (pid 12867) 167s
run: postgres-exporter: (pid 14246) 15s; run: log: (pid 13082) 130s
run: postgresql: (pid 12515) 280s; run: log: (pid 12532) 277s
run: prometheus: (pid 14183) 18s; run: log: (pid 12931) 149s
run: puma: (pid 12718) 199s; run: log: (pid 12727) 196s
run: redis: (pid 12309) 293s; run: log: (pid 12318) 292s
run: redis-exporter: (pid 14176) 18s; run: log: (pid 12908) 154s
run: sidekiq: (pid 12737) 193s; run: log: (pid 12749) 190s
```

Secure GitLab with Let's Encrypt

At this point, GitLab is installed and configured. Next, it is recommended to secure your GitLab instance with Let's Encrypt SSL.

First, install the Let's Encrypt client tool with the following command:

```
apt-get install letsencrypt -y
```

Once installed, edit the /etc/gitlab/gitlab.rb file and enable the Let's Encrypt.

```
nano /etc/gitlab/gitlab.rb
```

Change the following lines:

```
external_url 'https://gitlab.linuxbuz.com'
letsencrypt['enable'] = true
letsencrypt['contact_emails'] = ['admin@example.com']
letsencrypt['auto_renew'] = true
```

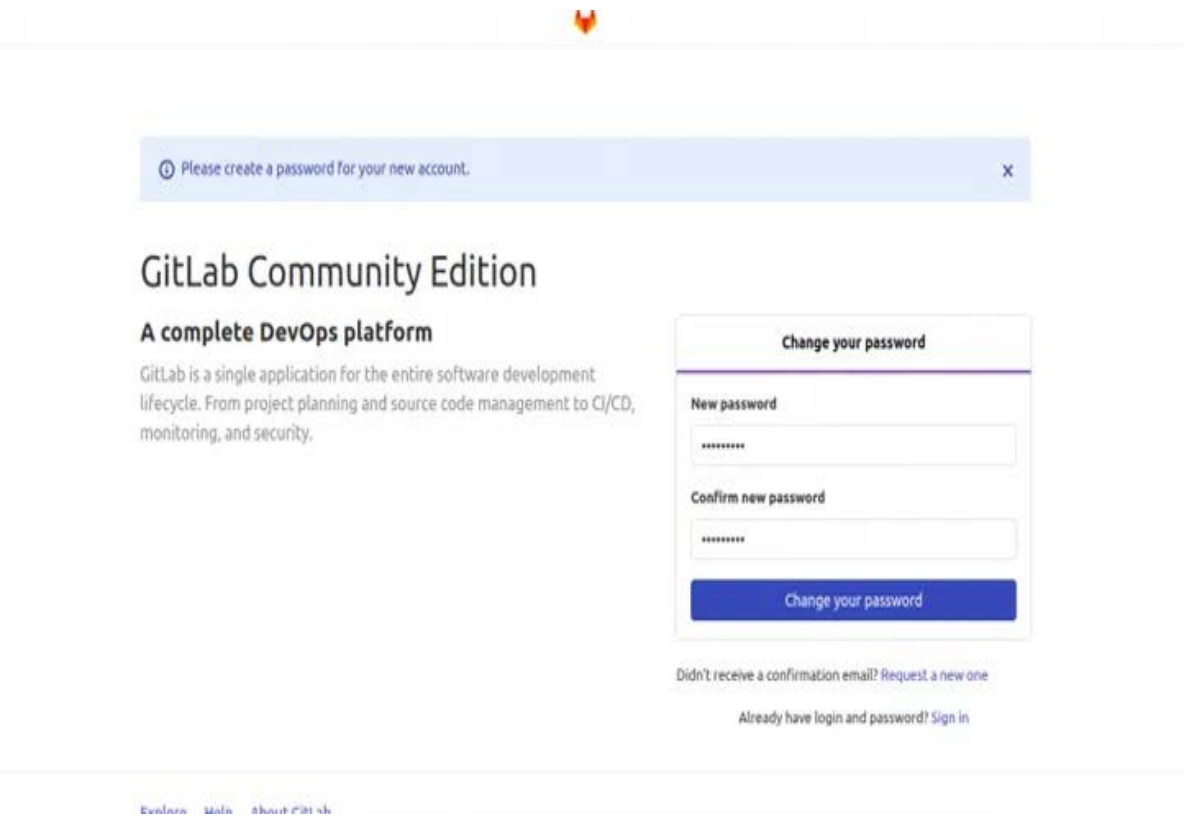
Save and close the file when you are finished. Then, reconfigure GitLab with the following command:

```
gitlab-ctl reconfigure
```

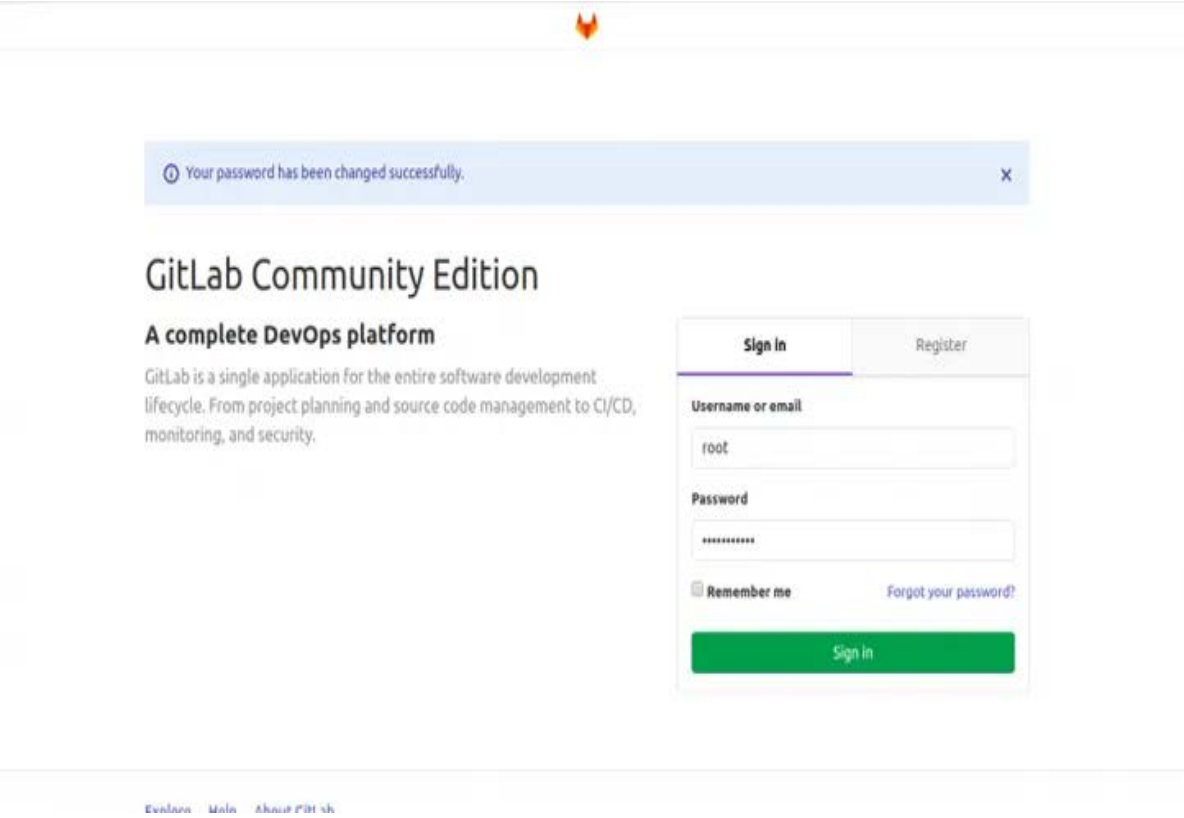
Once you are finished. You can proceed to the next step.

Access GitLab Interface

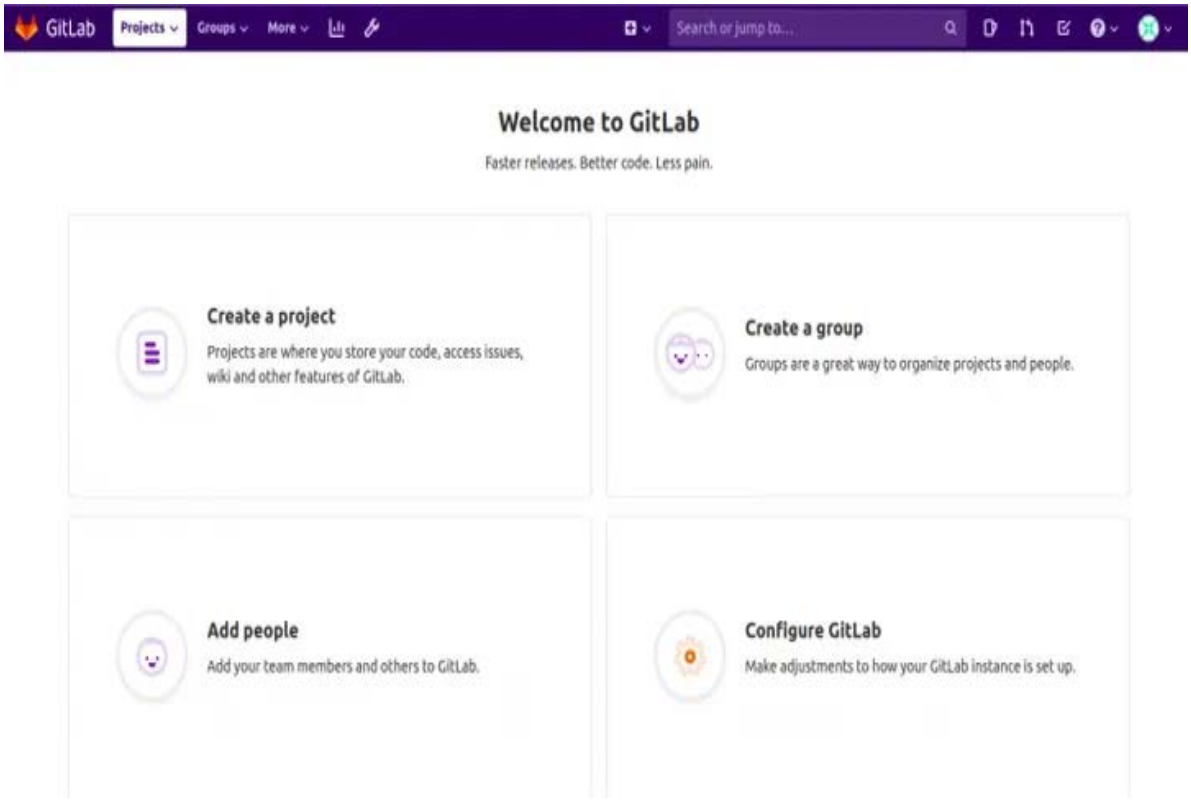
At this point, GitLab is secured with Let's Encrypt free SSL. Now, open your web browser and type the URL **https://gitlab.linuxbuz.com**. You will be redirected to the following page:



Now, set your new password and click on the **Change your password** button. You should see the following screen:



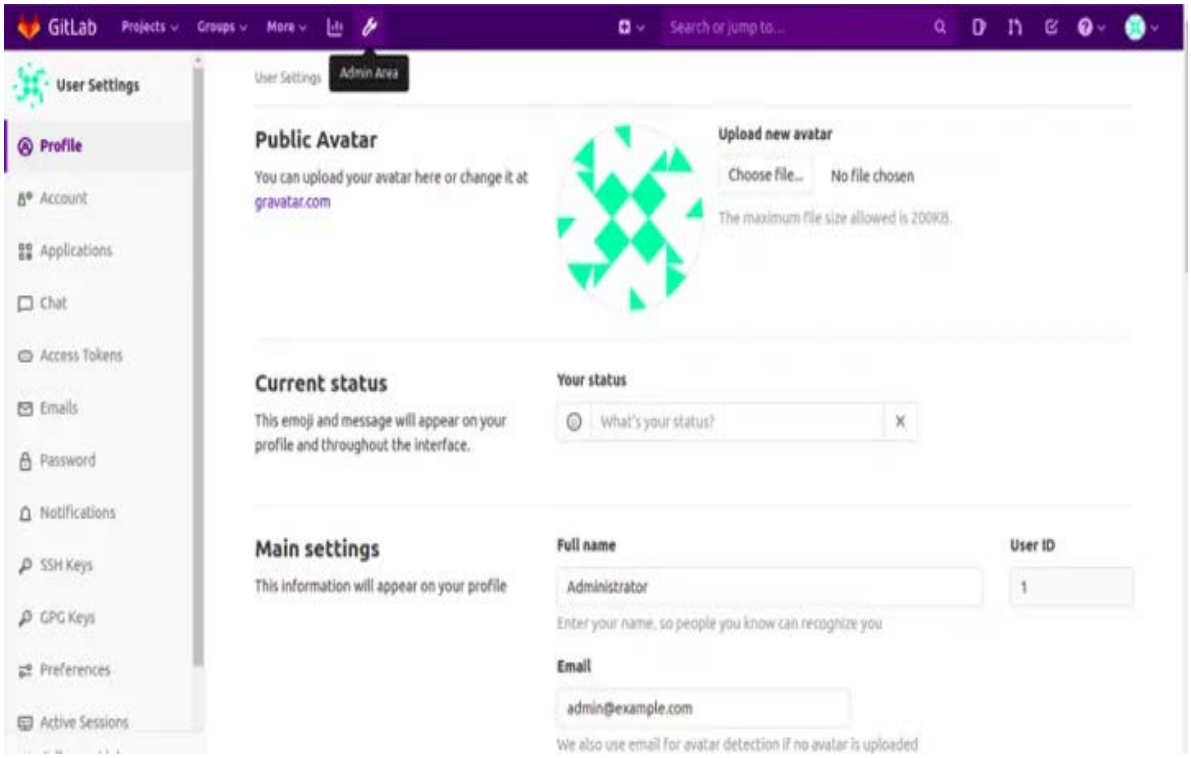
Now, provide your root username and password, then click on the **Sign in** button. You should see the following screen:

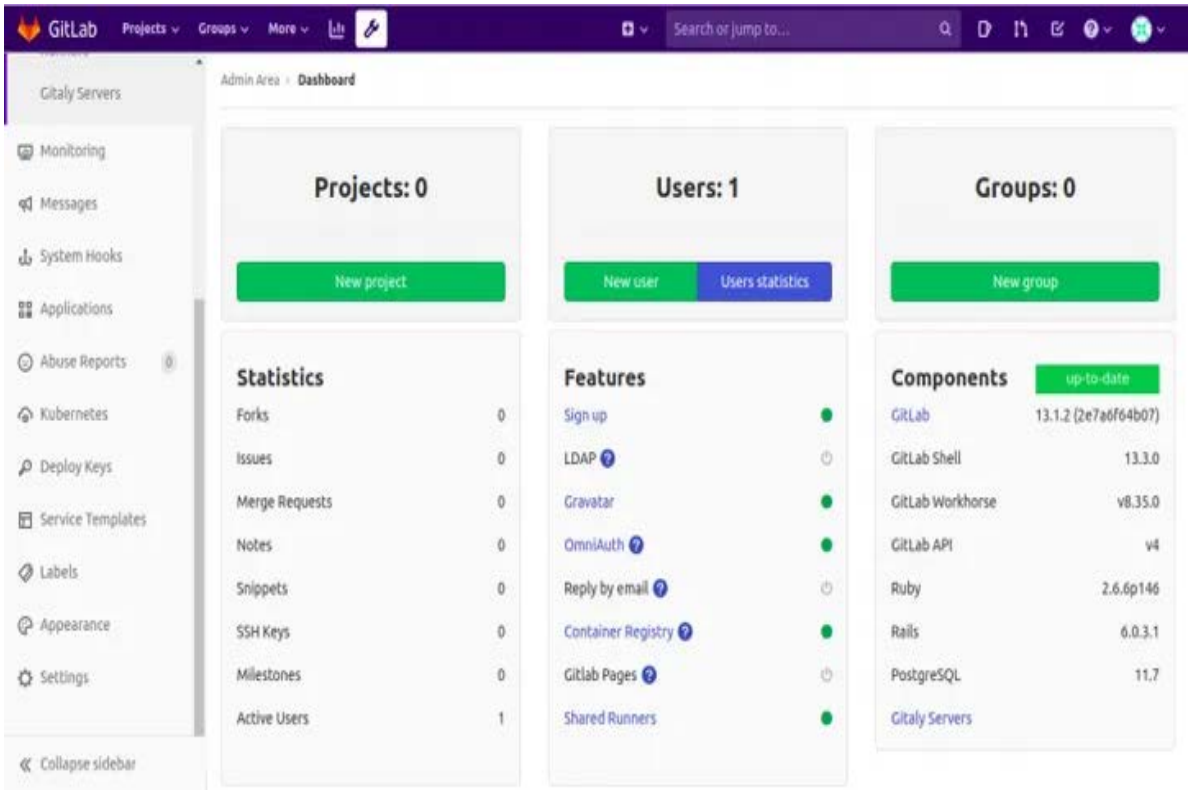


Disable Public Sign-up

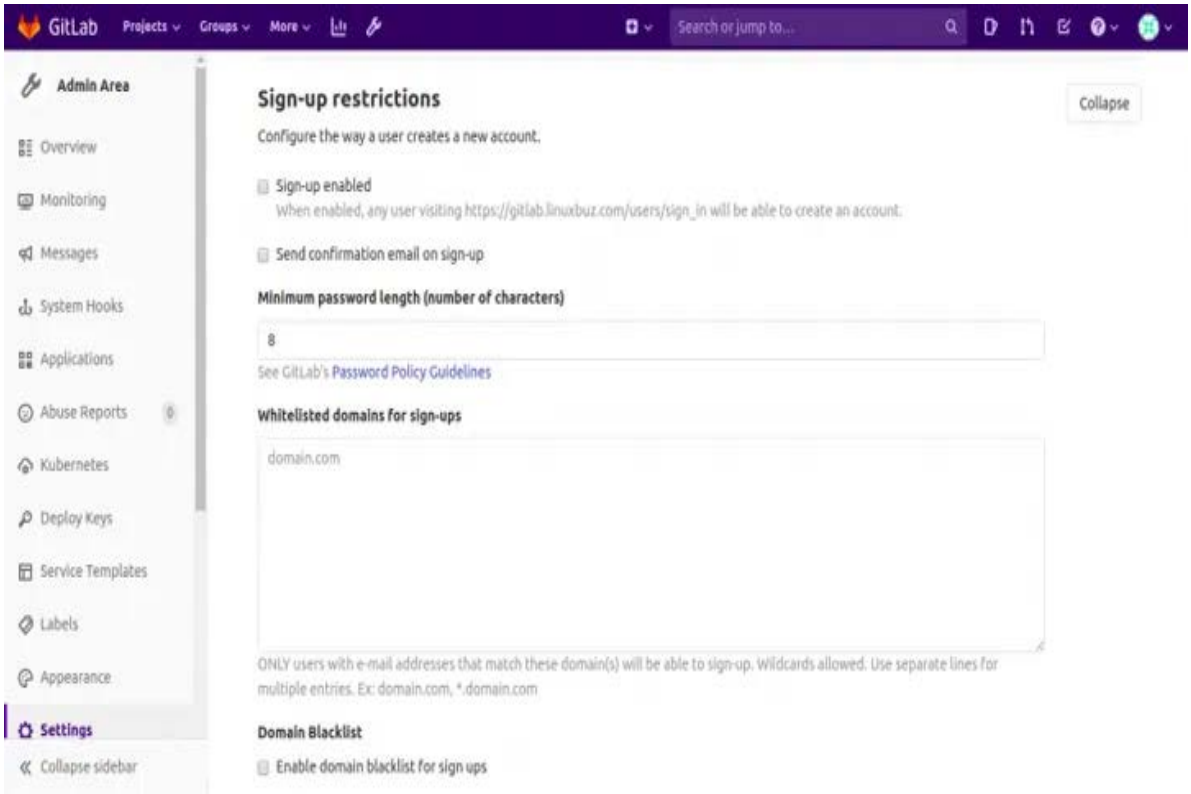
By default, GitLab allows anyone to sign up for an account using the GitLab URL. This settins is useful if you want to host a public project. However, if you want to use GitLab for your internal project, it is recommended to disable Public Sign-up.

On the GitLab dashboard, click on the **admin** area icon. You should see the following screen:



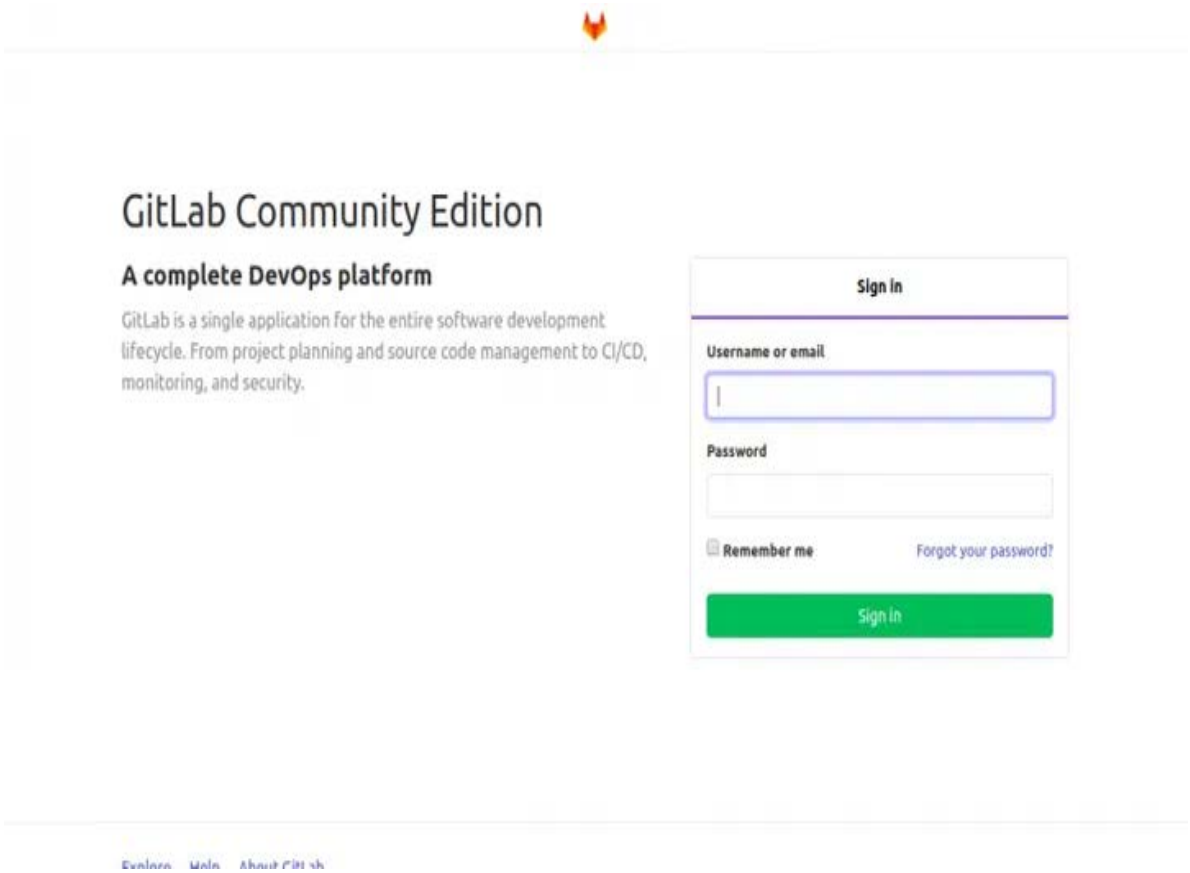


Now, click on the **Settings** in the left pane. You should see the following screen:



Next, scroll down to the Sign-up restrictions and click on the Expand button. Uncheck the Sign-up enabled box and click on Save changes when finished.

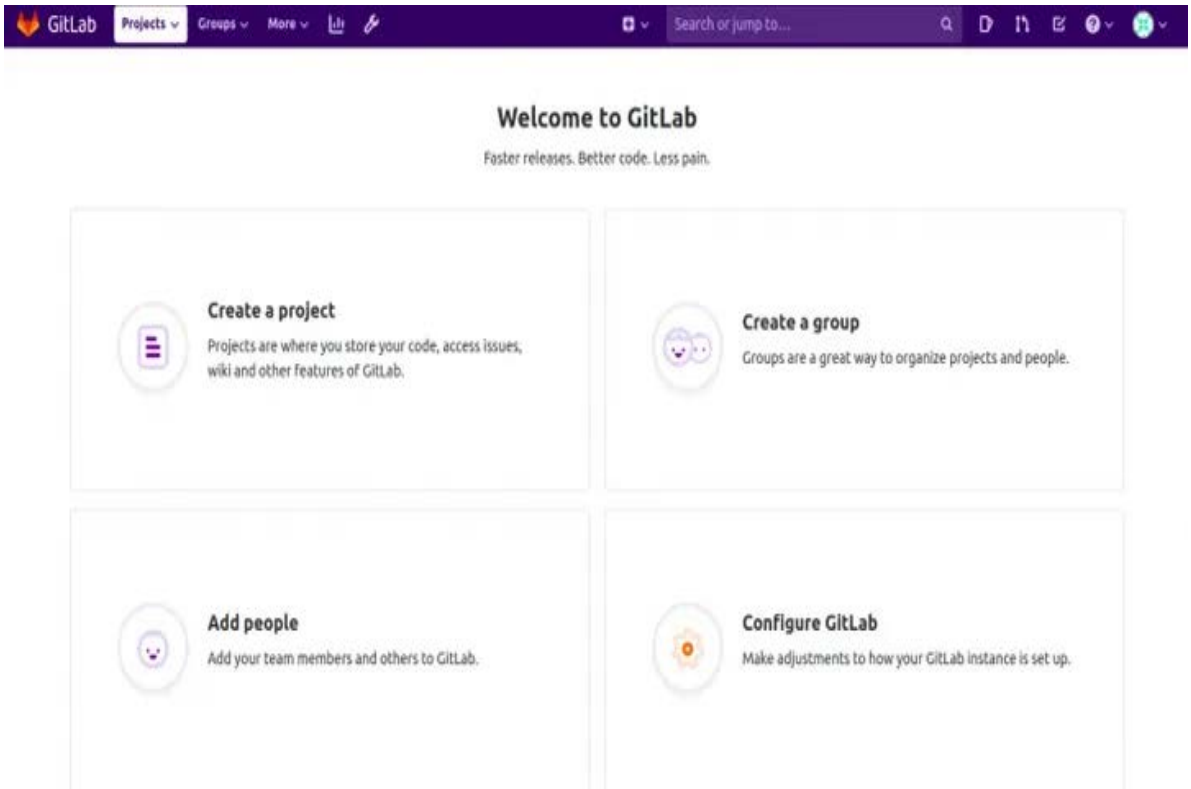
Now, log out from GitLab and access GitLab again. You should see the following screen with public sign up disabled:



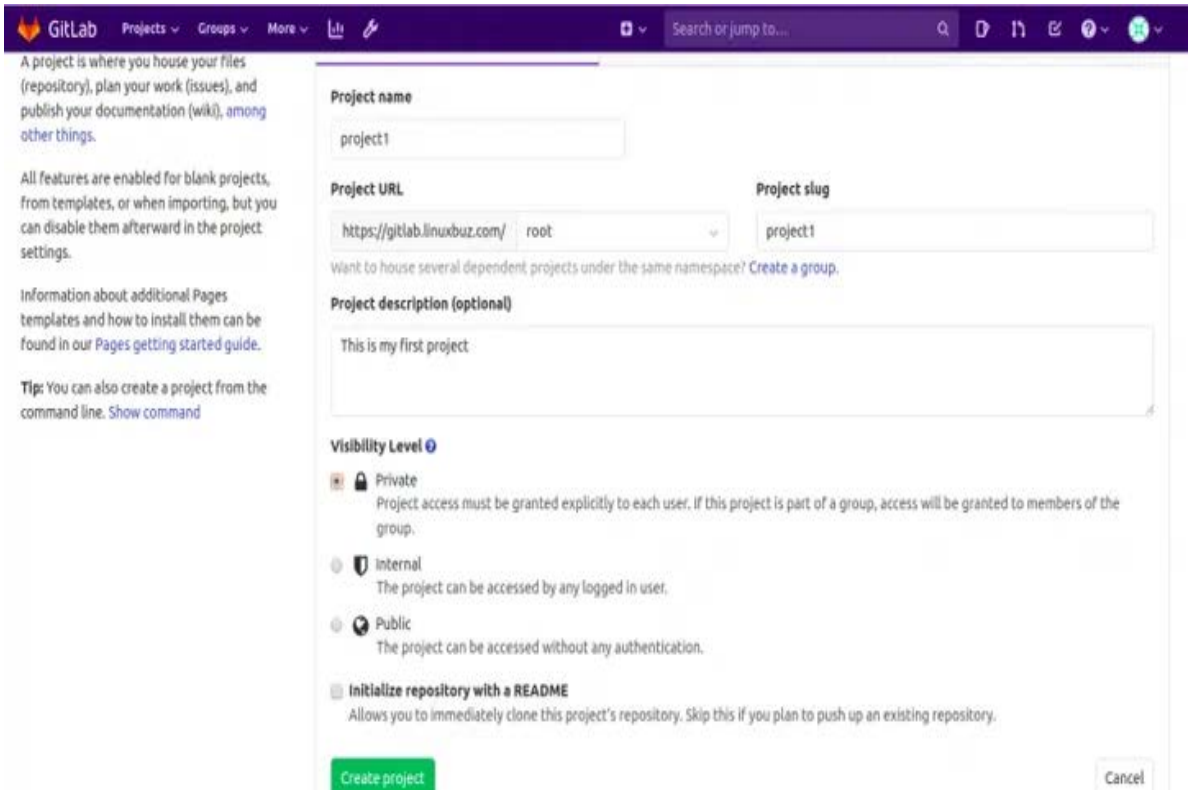
Verify GitLab Functionality

At this point, GitLab is installed and configured. Next, create a new project from the GitLab dashboard.

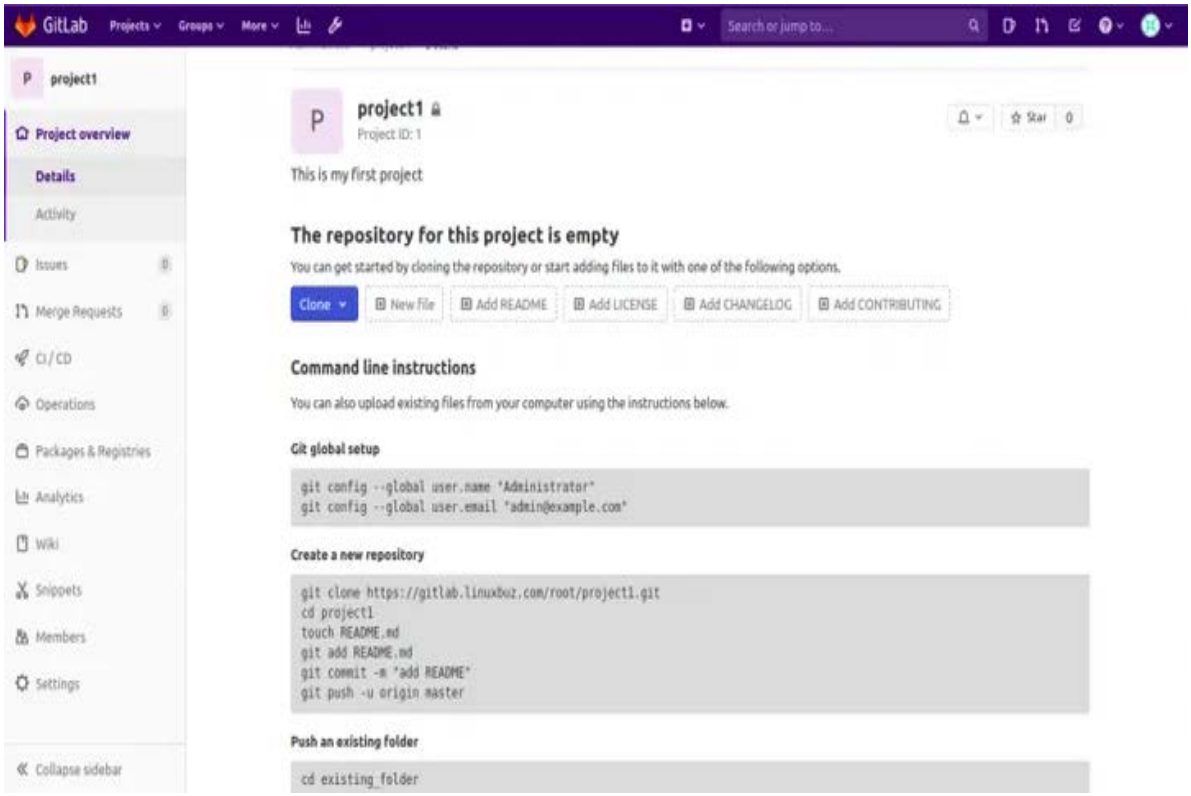
On the GitLab dashboard, click on the project button. You should see the following screen:



Now, click on the **Create a project** button. You should see the following screen:



Provide your project name, URL, description, visibility level and click on the **Create project** button. You should see the following screen:



Next, go to the remote system and configure the Git with the following command:

```
git config --global user.name "Hitesh Jethva"
git config --global user.email "hitjethva@gmail.com"
```

Next, clone the repository of your project with the following command:

```
git clone https://gitlab.linuxbuz.com/root/project1.git
```

You will be asked to provide your GitLab username and password as shown below:

```
Cloning into 'project1'...
Username for 'https://gitlab.linuxbuz.com': root
Password for 'https://root@gitlab.linuxbuz.com':
warning: You appear to have cloned an empty repository.
Checking connectivity... done.
```

Next, change the directory to your project and create a README.md file:

```
cd project1
echo "This is my first file" > README.md
```

Next, add the README.md file to your repository and commit changes with the following command:

```
git add README.md
git commit -m "add README"
```

You should see the following output:

```
[master (root-commit) ae4d108] add README
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
```

Next, push your file to the remote repository with the following command:

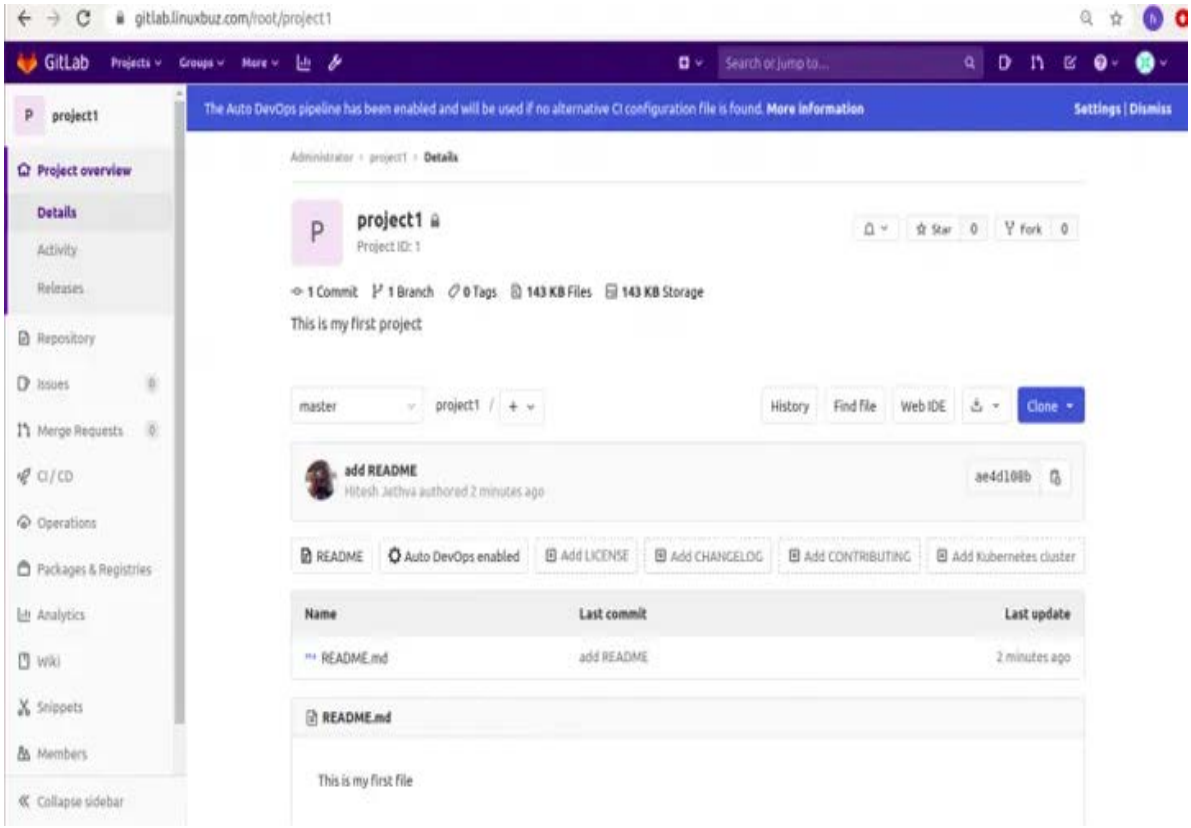
```
git push -u origin master
```

Provide your GitLab username and password to push the file to the remote repository:

```
Username for 'https://gitlab.linuxbuz.com': root
Password for 'https://root@gitlab.linuxbuz.com':
Counting objects: 3, done.
Writing objects: 100% (3/3), 237 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
```

```
To https://gitlab.linuxbuz.com/root/project1.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.
```

Now, open your GitLab dashboard and click on the **project** button. You should see your README.md file in the following screen:



Conclusion

In the above guide, you learned how to install GitLab and secure it with Let's Encrypt SSL on Ubuntu 20.04. You also learned how to perform basic Git operations with GitLab. I hope you can now deploy GitLab in your development environment. Feel free to ask me if you have any questions.

About Hitesh Jethva

Over 8 years of experience as a Linux system administrator. My skills include a depth knowledge of Redhat/Centos, Ubuntu Nginx and Apache, Mysql, Subversion, Linux, Ubuntu, web hosting, web server, Squid proxy, NFS, FTP, DNS, Samba, LDAP, OpenVPN, Haproxy, Amazon web services, WHMCS, OpenStack Cloud, Postfix Mail Server, Security etc.