GitLab 私有远程仓库搭建

参考：百度
制作日期：2019-01-29
制作人：小桅[yw_forgit@163.com]

1、GitLab 安装

```
// 安装GitLab依赖包
[root@localhost home]# yum install -y curl policycoreutils-python openssh-server
openssh-clients

// 查看centos的版本，本教程的是7.5，是centos7
[root@localhost home]# cat /etc/centos-release
CentOS Linux release 7.5.1804 (Core)

下载对应操作系统的gitlab的rpm包，用浏览器打开（本教程用gitlab-ce-11.5.7）
centos6系统的下载地址：https://mirrors.tuna.tsinghua.edu.cn/gitlab-ce/yum/el6
centos7系统的下载地址：https://mirrors.tuna.tsinghua.edu.cn/gitlab-ce/yum/el7

// centos7的执行这个（本教程用的centos7）
[root@localhost home]# wget https://mirrors.tuna.tsinghua.edu.cn/gitlab-
ce/yum/el7/gitlab-ce-11.5.7-ce.0.el7.x86_64.rpm

// centos6的执行这个
[root@localhost home]# wget https://mirrors.tuna.tsinghua.edu.cn/gitlab-
ce/yum/el6/gitlab-ce-11.5.7-ce.0.el6.x86_64.rpm

// 这里我先进来home目录，然后执行了wget
[root@localhost home]# ll
total 454180
-rw-r--r--. 1 root   root   465080251 Jan 16 10:36 gitlab-ce-11.5.7-
ce.0.el7.x86_64.rpm
drwx------. 3 houtai houtai       78 Jan 28 22:37 houtai
drwxr-xr-x. 6 root   root         86 Jan 25 01:37 svnProject

// 如果没有wget命令，则安装一下，然后再下载rpm
[root@localhost home]# yum install wget


// 通过yum安装本地GitLab的rpm包
[root@localhost home]# yum -y localinstall gitlab-ce-11.5.7-ce.0.el7.x86_64.rpm
```

安装完成有个大大的 logo

```
Transaction test succeeded
Running transaction
Warning: RPMDB altered outside of yum.
  Installing : gitlab-ce-11.5.7-ce.0.el7.x86_64
It looks like GitLab has not been configured yet; skipping the upgrade script.

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



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


Thank you for installing GitLab!
GitLab should be available at http://192.168.0.21:9999

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

  Verifying  : gitlab-ce-11.5.7-ce.0.el7.x86_64

Installed:
  gitlab-ce.x86_64 0:11.5.7-ce.0.el7

Complete!
```

2、修改 GitLab 配置文件

```
// 修改GitLab的主配置文件
[root@localhost home]# vi /etc/gitlab/gitlab.rb
```

这里配置 external_url 为 http://192.168.0.21:9999 (指定 gitlab 运行端口，后续访问 gitlab、代码提交都是这个 url)，如果用域名则直接用域名，但保证域名是可用的。

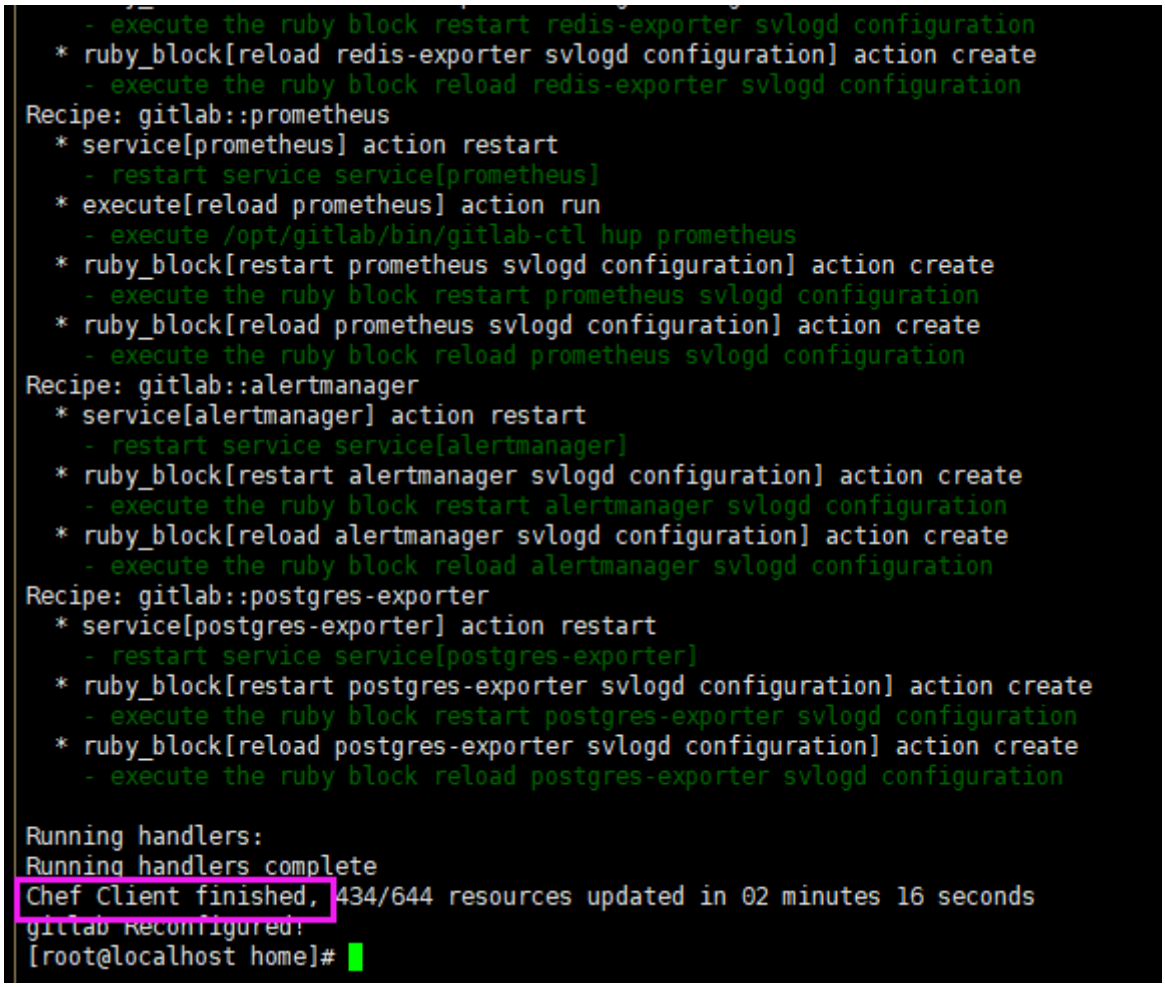
```
## GitLab configuration settings
##! This file is generated during initial installation and **is not** modified
##! during upgrades.
##! Check out the latest version of this file to know about the different
##! settings that can be configured by this file, which may be found at:
##! https://gitlab.com/gitlab-org/omnibus-gitlab/raw/master/files/gitlab-config-template/gitlab.rb.template


## GitLab URL
##! URL on which GitLab will be reachable.
##! For more details on configuring external_url see:
##! https://docs.gitlab.com/omnibus/settings/configuration.html#configuring-the-external-url-for-gitlab
#external_url 'http://gitlab.example.com'
external_url 'http://192.168.0.21:9999'

## Roles for multi-instance GitLab
##! The default is to have no roles enabled, which results in GitLab running as an all-in-one instance.
##! Options:
##!   redis_sentinel_role redis_master_role redis_slave_role geo_primary_role geo_secondary_role
##! For more details on each role, see:
##! https://docs.gitlab.com/omnibus/roles/README.html#roles
##!
# roles ['redis_sentinel_role', 'redis_master_role']

## Legend
##! The following notations at the beginning of each line may be used to
##! differentiate between components of this file and to easily select them using
##! a regex.
##! ## Titles, subtitles etc
##! ##! More information - Description, Docs, Links, Issues etc.
##! Configuration settings have a single # followed by a single space at the
##! beginning; Remove them to enable the setting.

##! **Configuration settings below are optional.**
##! **The values currently assigned are only examples and ARE NOT the default
##!   values.**
```

3、初始化 GitLab

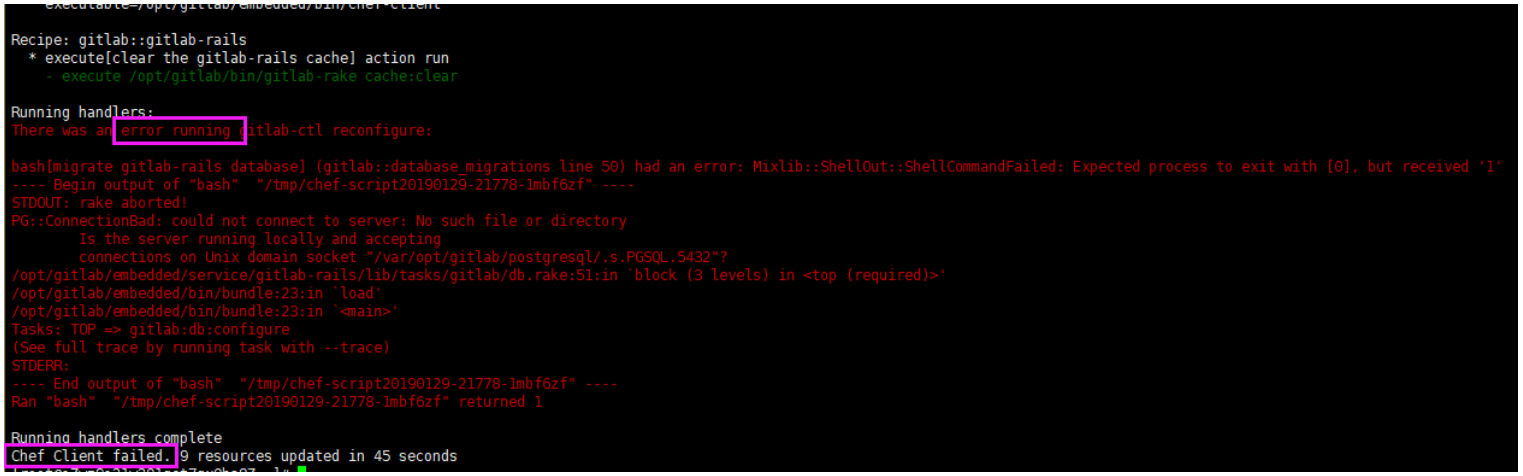
初始化

```
// 初始化GitLab，要等待挺久的，不断的输出
[root@localhost home]# gitlab-ctl reconfigure
```

最后一定要看到这句 Chef Client finished，如果是 Chef Client failed 就要检查哪里报错了，一般错误信息会有红色字体提示。

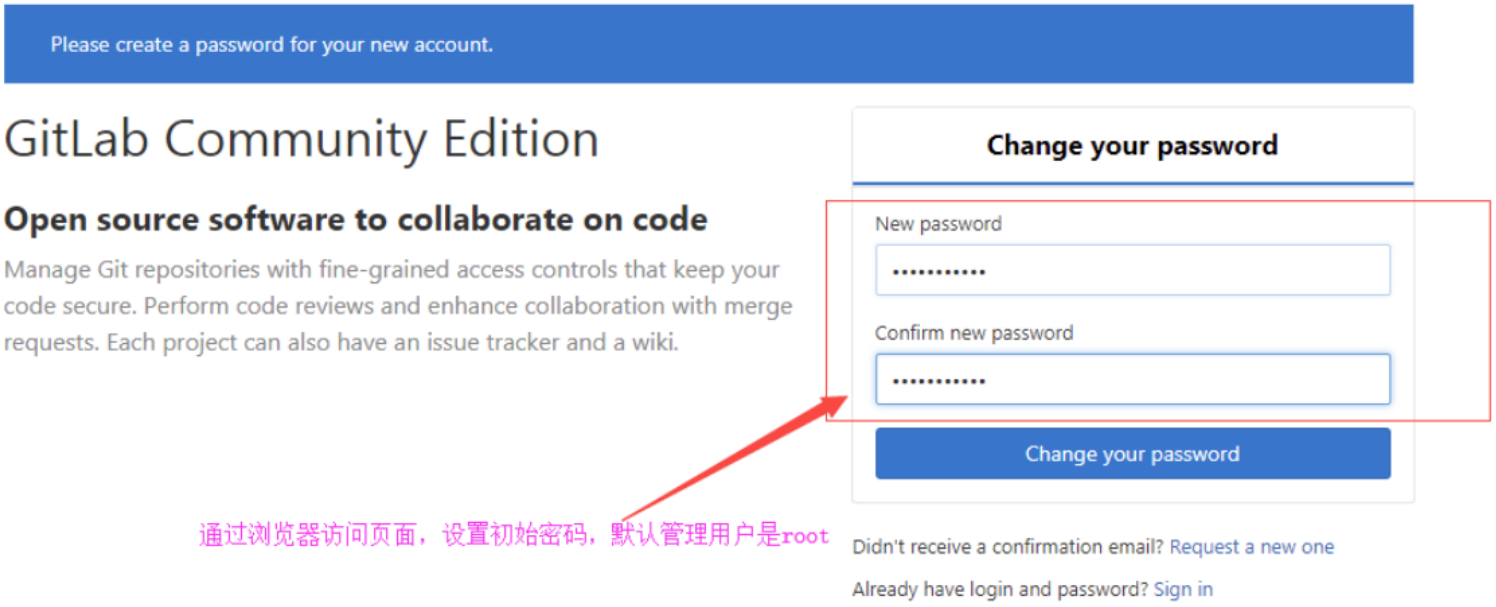


比如我在另外的机子就有报错，这里看不到报错信息，去看日志文件会有提示的



页面访问初始化 root 用户密码

访问 http://192.168.0.21:9999 ，就是那个 external_url，能访问则表示成功，然后修改默认 root 用户(管理员)的密码，不需要旧的密码（也没有），直接修改即可。



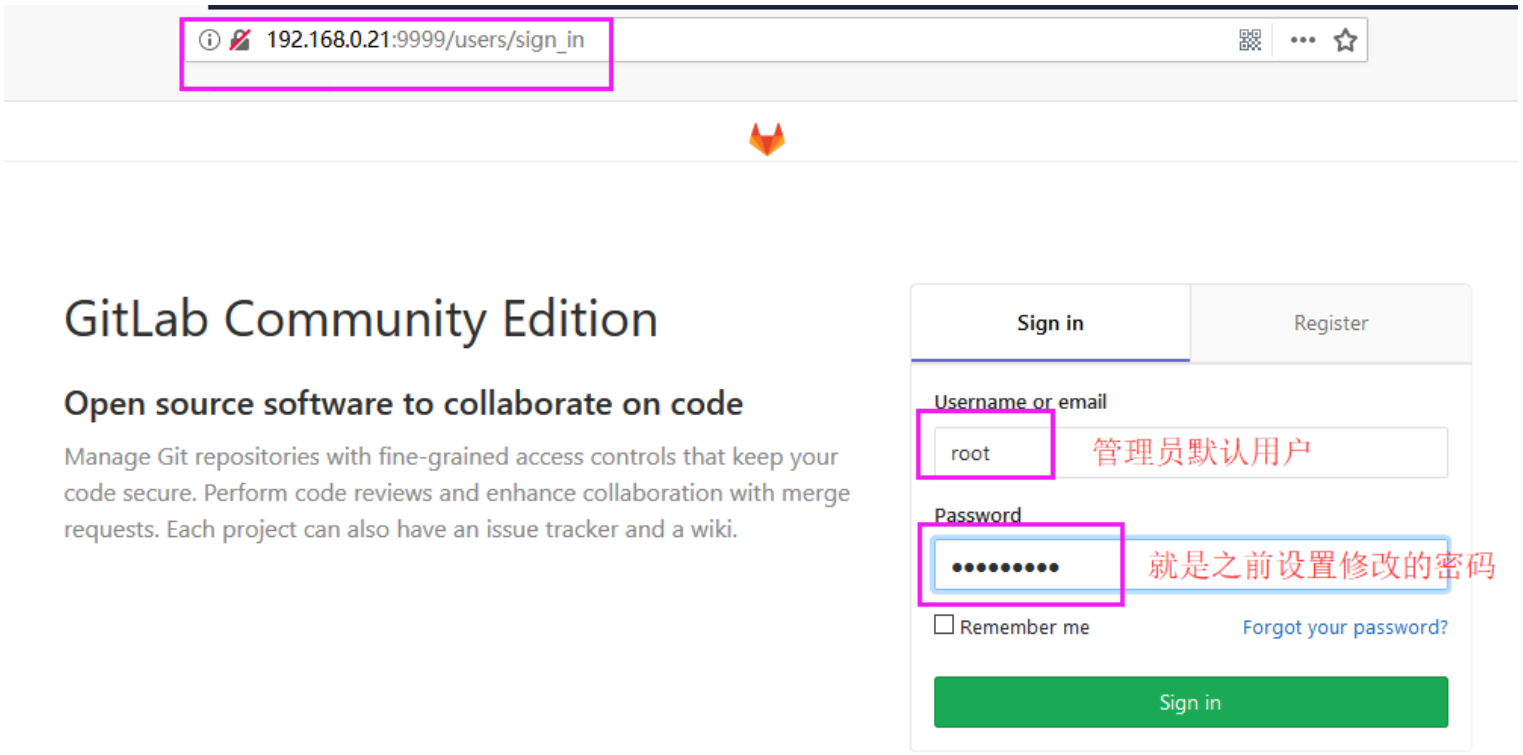
如果不能访问，可能是端口没有开放（gitlab 在我的 CentOS 上，浏览器访问在 win10 上，故要开放端口）
确定是否是端口问题，可以在 gitlab 所在的主机，使用命令：

```
// 如果能返回设置初始化密码的 HTML 页面信息，则表示是端口问题。
[root@localhost ~]# curl http://192.168.0.21:9999
```

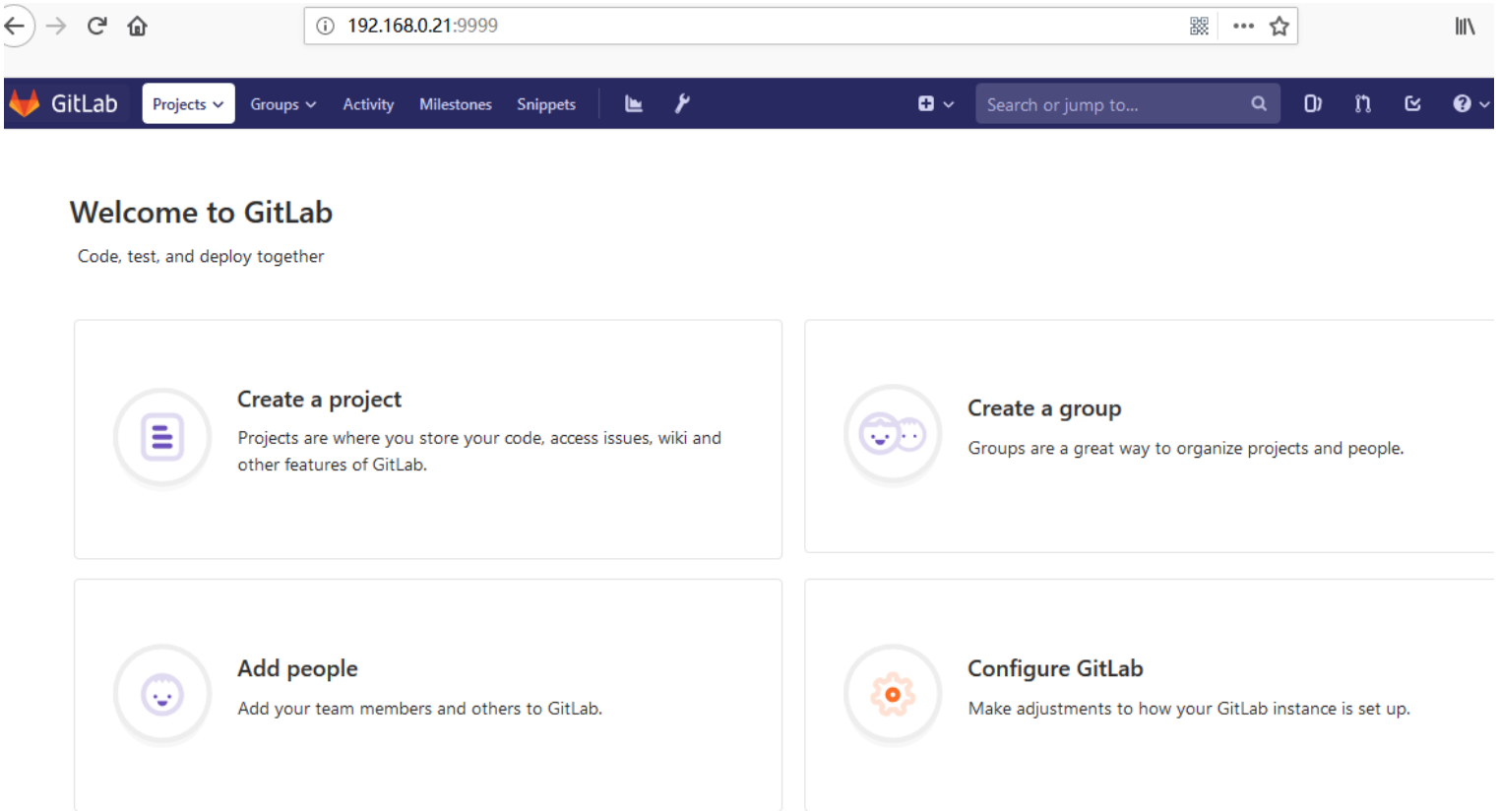
CentOS7（注意 CentOS7 默认使用 firewall 作为防火墙，6 的可能不适用）对外开放端口：

```
// --permanent 永久生效，没有此参数重启后失效
[root@localhost ~]# firewall-cmd --zone=public --add-port=9999/tcp --permanent
// 重新载入配置
[root@localhost ~]# firewall-cmd --reload
```

修改成功就会自动跳转到



登录成功页面



温馨提示：

如果安装错误了，需要重新安装，则

rpm -e gitlab-ce 卸载

cd / 去到根目录

find -name gitlab* 把 gitlab 所有的都找出来，然后

rm -rf .... 全部删除，就可以重新安装了，不然删除干净就重新安装可能会有问题。

管理 GitLab 服务的常用命令：

```
// 如果修改了external_url，重启一下就好了
[root@localhost home]# gitlab-ctl restart
// 如果不行，可能要用
[root@localhost home]# gitlab-ctl reconfigure

启动GitLab服务：
[root@localhost home]# gitlab-ctl start
查看GitLab的状态：
[root@localhost home]# gitlab-ctl status
停止GitLab服务：
[root@localhost home]# gitlab-ctl stop
```

穿插清理缓存的知识点：

```
// 查看运行内存
[root@localhost home]# free -m
// free就是剩余多少；cached就是缓存，centos会缓存大文件什么的
             total       used       free     shared    buffers     cached
Mem:          3961       3749        212          0        225        560
-/+ buffers/cache:       2963        998
Swap:            0          0          0
// 想清除缓存的操作
[root@localhost home]# sync
[root@localhost home]# echo 3 > /proc/sys/vm/drop_caches
// 然后再看一下，就cached就很少了，可用运行内存就多了
[root@localhost home]# free -m
             total       used       free     shared    buffers     cached
Mem:          3961       2919       1042          0          2         18
-/+ buffers/cache:       2898       1062
Swap:            0          0          0
[root@localhost home]#
```

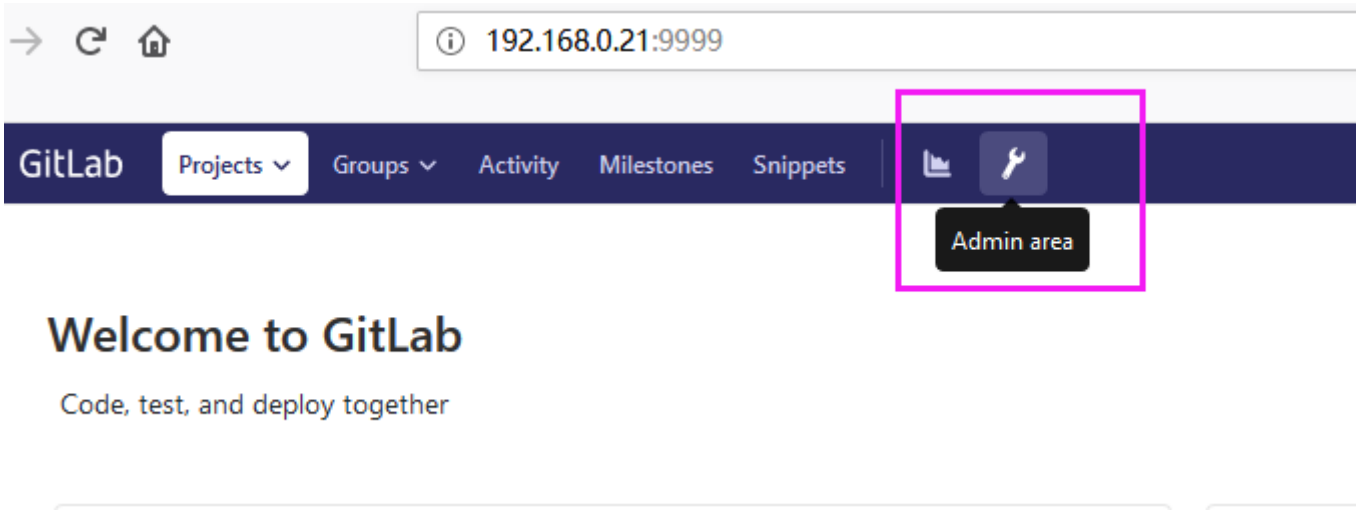
GitLab 感觉挺耗内存的

```
// 安装、运行GitLab之前的内存
[root@localhost home]# free -m
             total       used       free     shared  buff/cache   available
Mem:          15828        255      13120          8       2452       15088
```

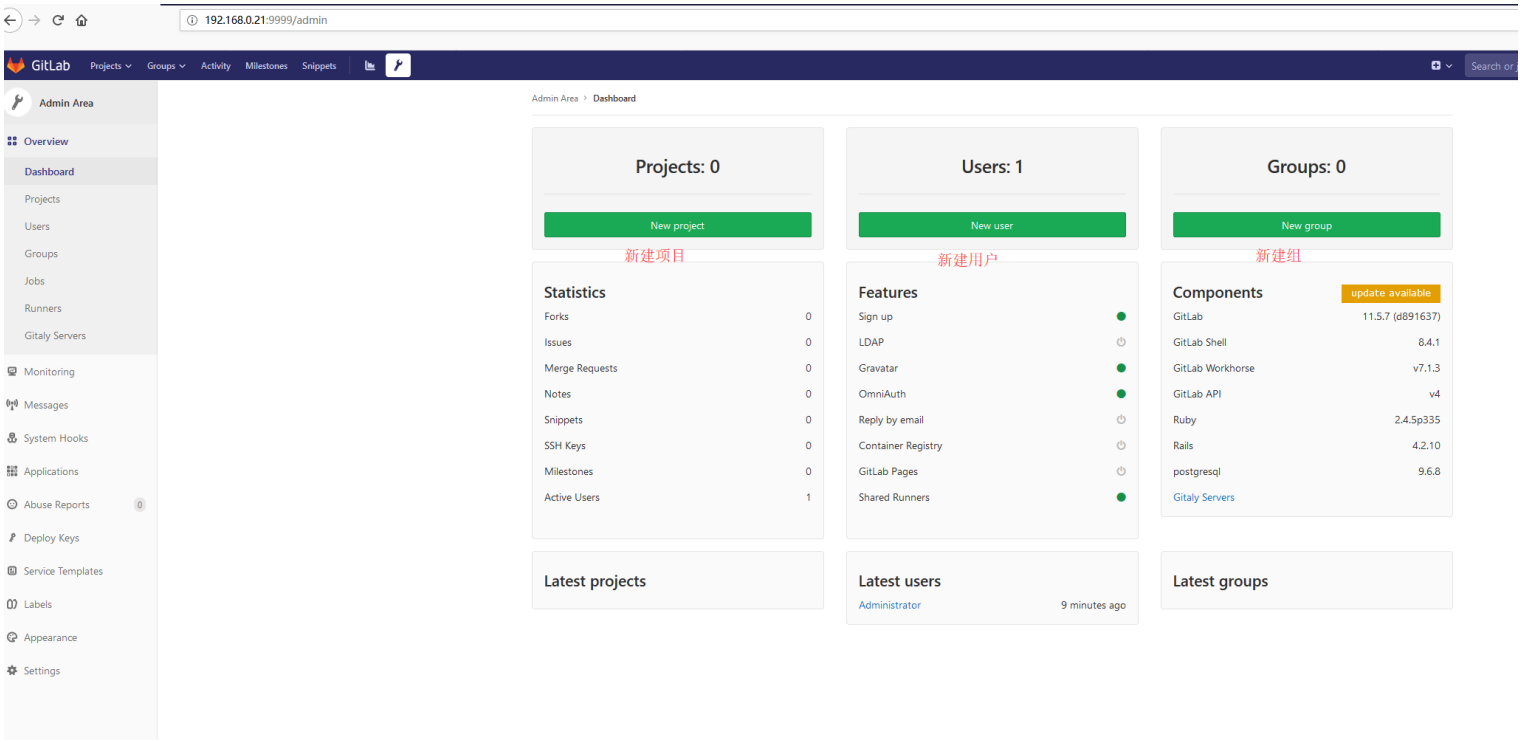
```
Swap:          12031           0       12031
// 安装、运行GitLab之后的内存
[root@localhost home]# free -m
              total        used        free      shared  buff/cache   available
Mem:          15828        5946        7082         130        2798        9267
Swap:         12031           0       12031
[root@localhost home]#
```

4、GitLab 的使用

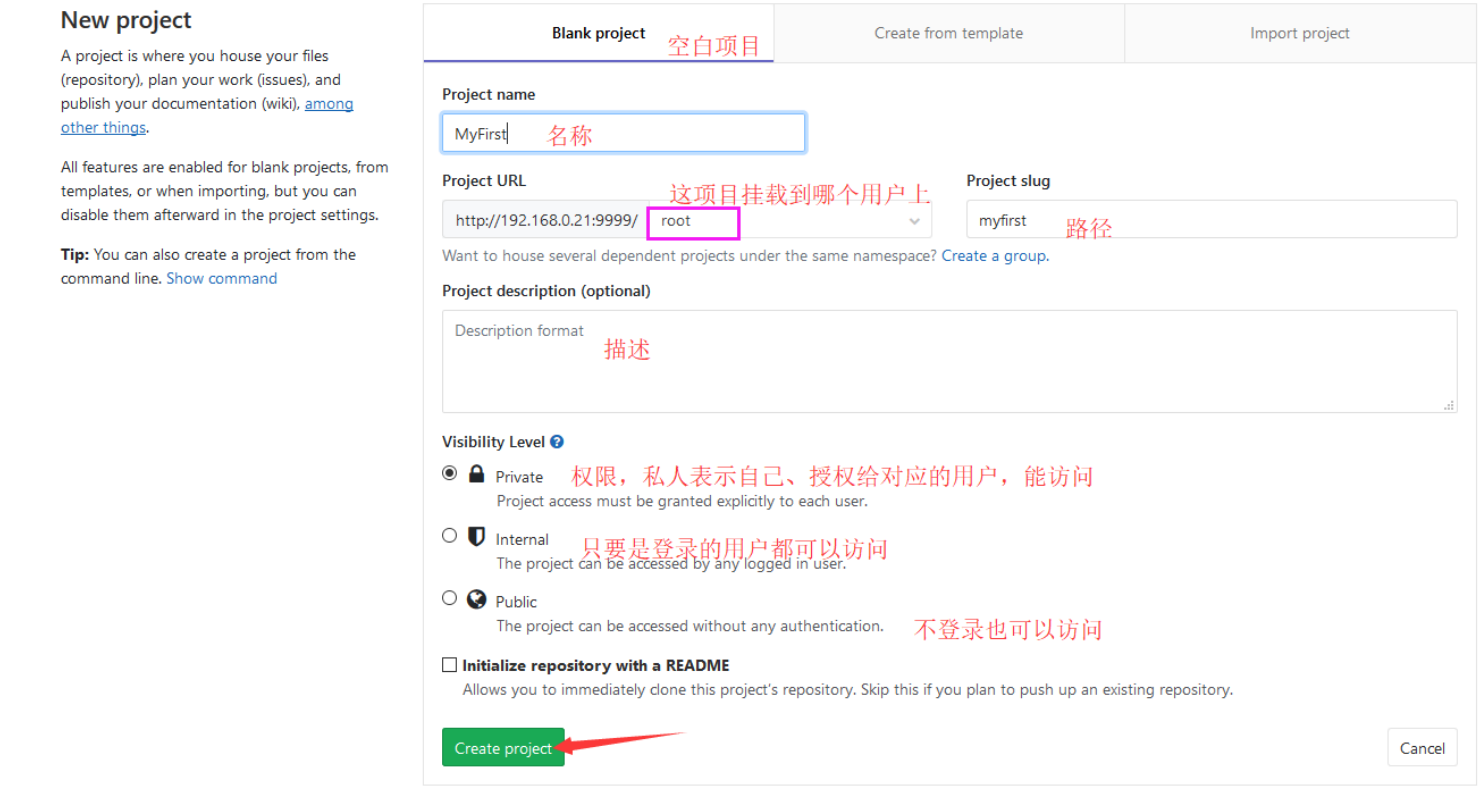
管理区域

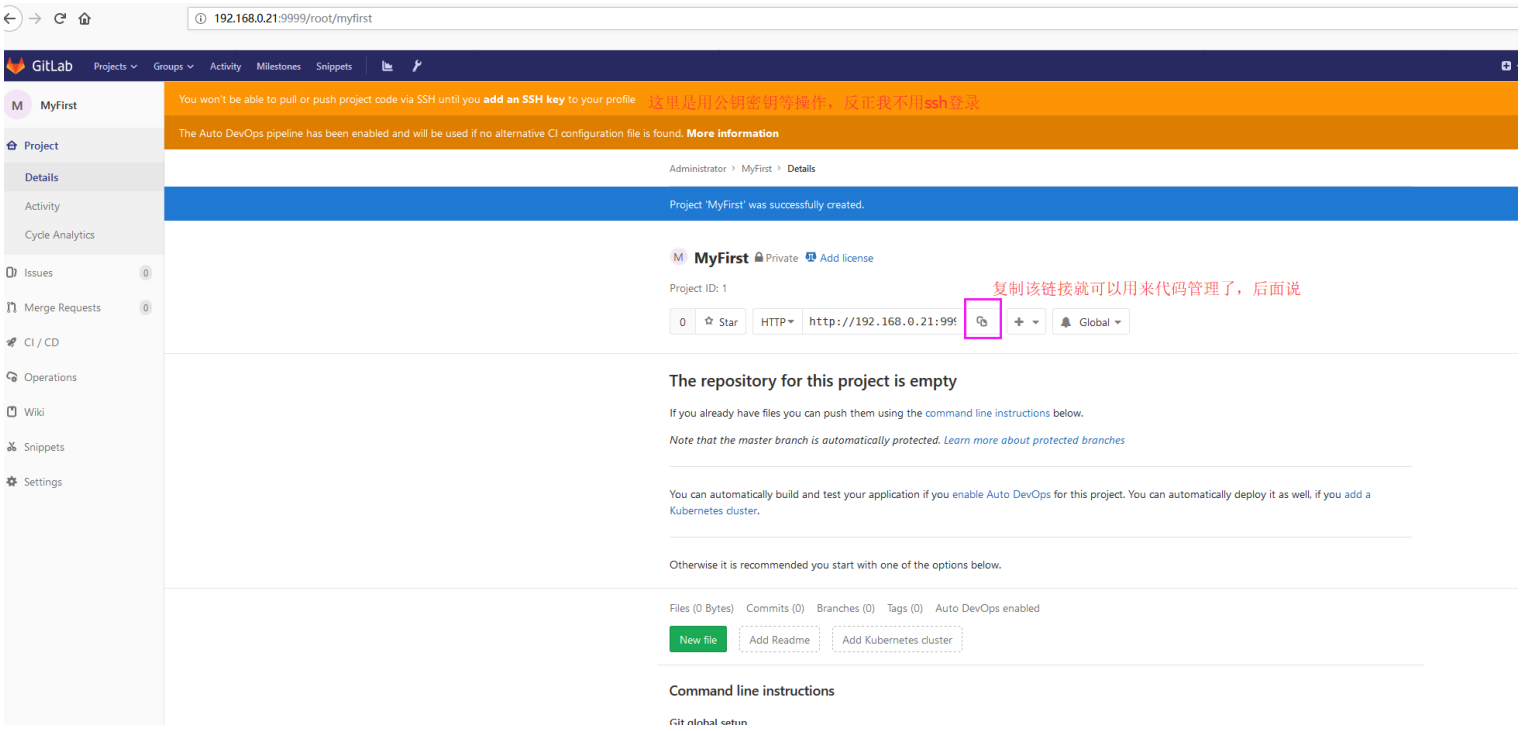


新建项目、新建用户、新建组 都在这个页面



新建项目



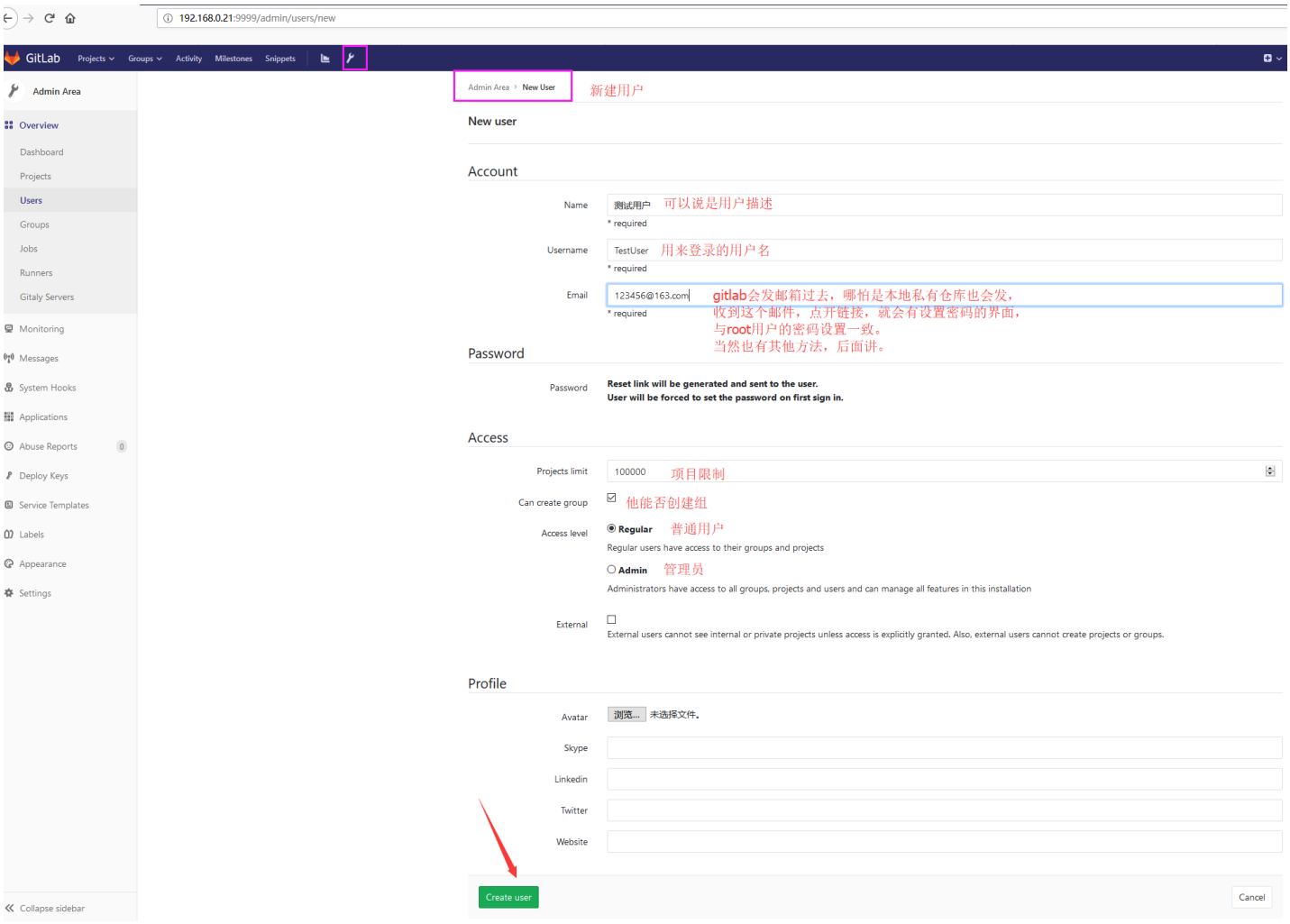


看下这个链接，myfirst 就是项目路径或者说是项目仓库名称

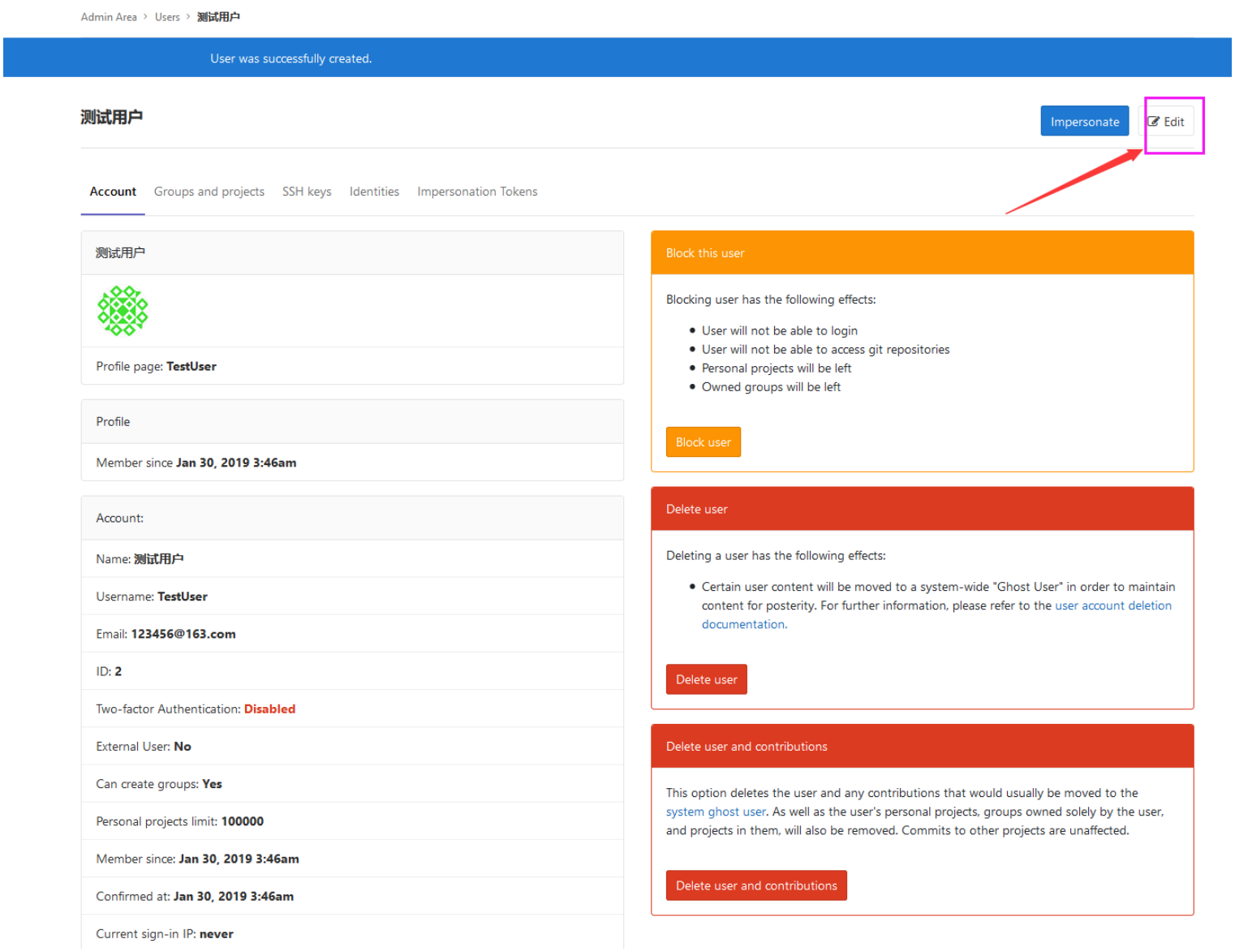
http://192.168.0.21:9999/root/myfirst.git

新建用户

新用户的密码设置有两种，下图是一种



创建完成，点击 Edit



第二种方法设置密码，上面点击完 Edit 之后，会有下图

Edit user: 测试用户

Account

Name: 测试用户
* required

Username: TestUser
* required

Email: 123456@163.com
* required

Password

Password: ••••••••

Password confirmation: ••••••••

这里就可以设置密码了，不需要经过邮箱也是可以的，
注意密码要8位以上

Access

Projects limit: 100000

Can create group: ☑

Access level:

◉ **Regular**
Regular users have access to their groups and projects

○ **Admin**
Administrators have access to all groups, projects and users and can manage all features in this installation

External: ☐
External users cannot see internal or private projects unless access is explicitly granted. Also, external users cannot create projects or groups.

Profile

Avatar: 浏览... 未选择文件。

Skype:

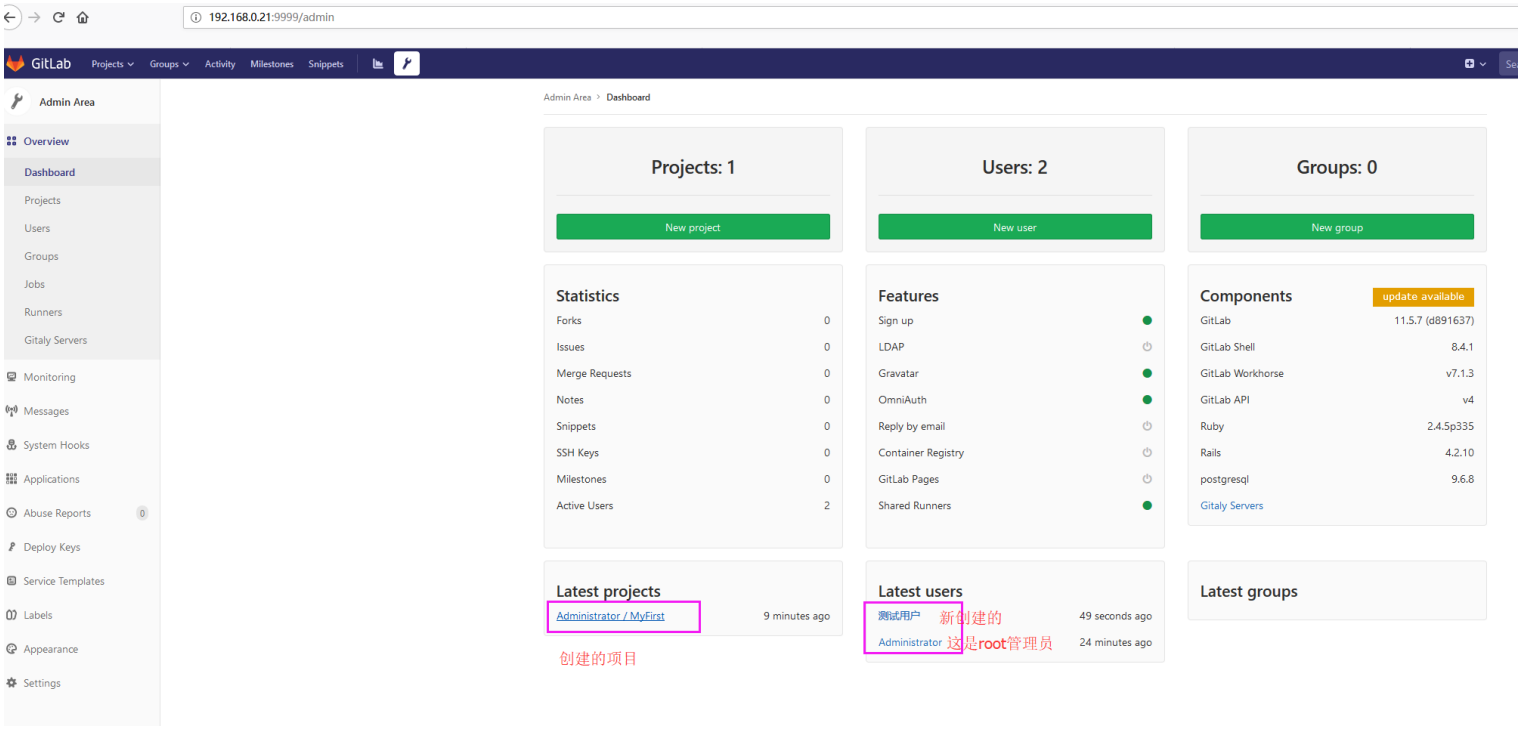
Linkedin:

Twitter:

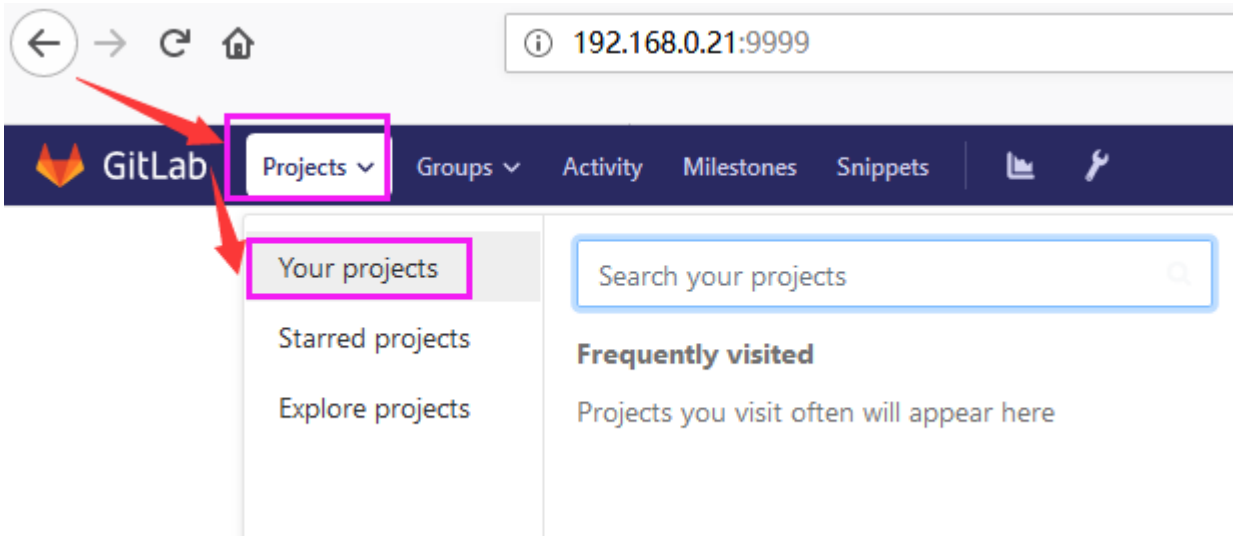
Website:

Save changes

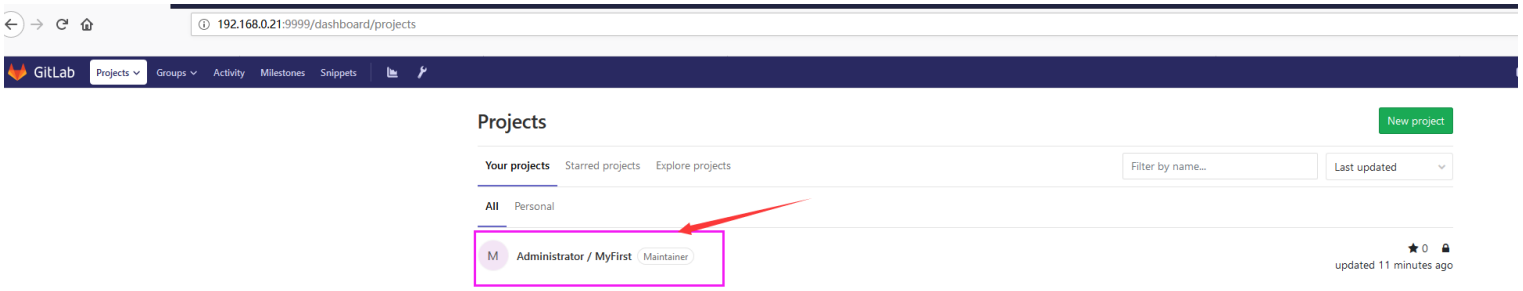
回到管理区域



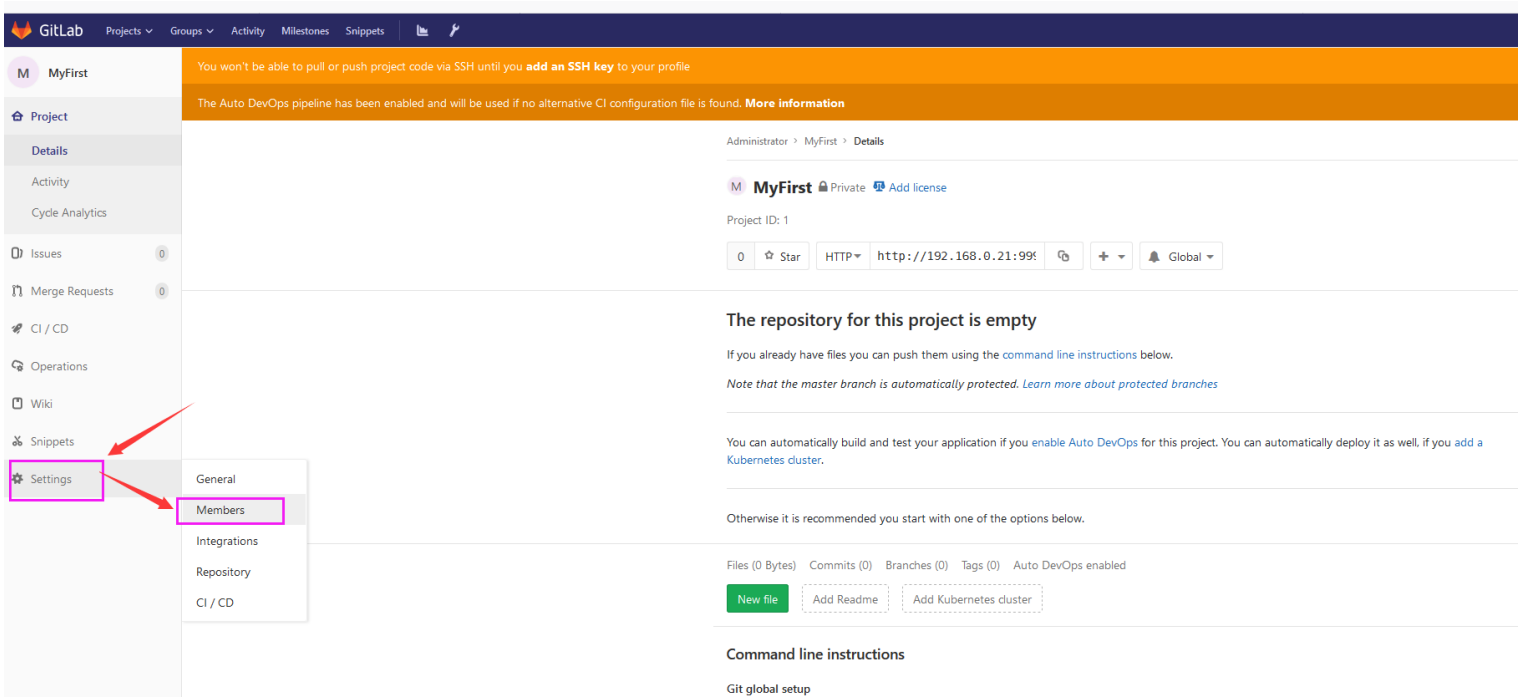
项目添加用户



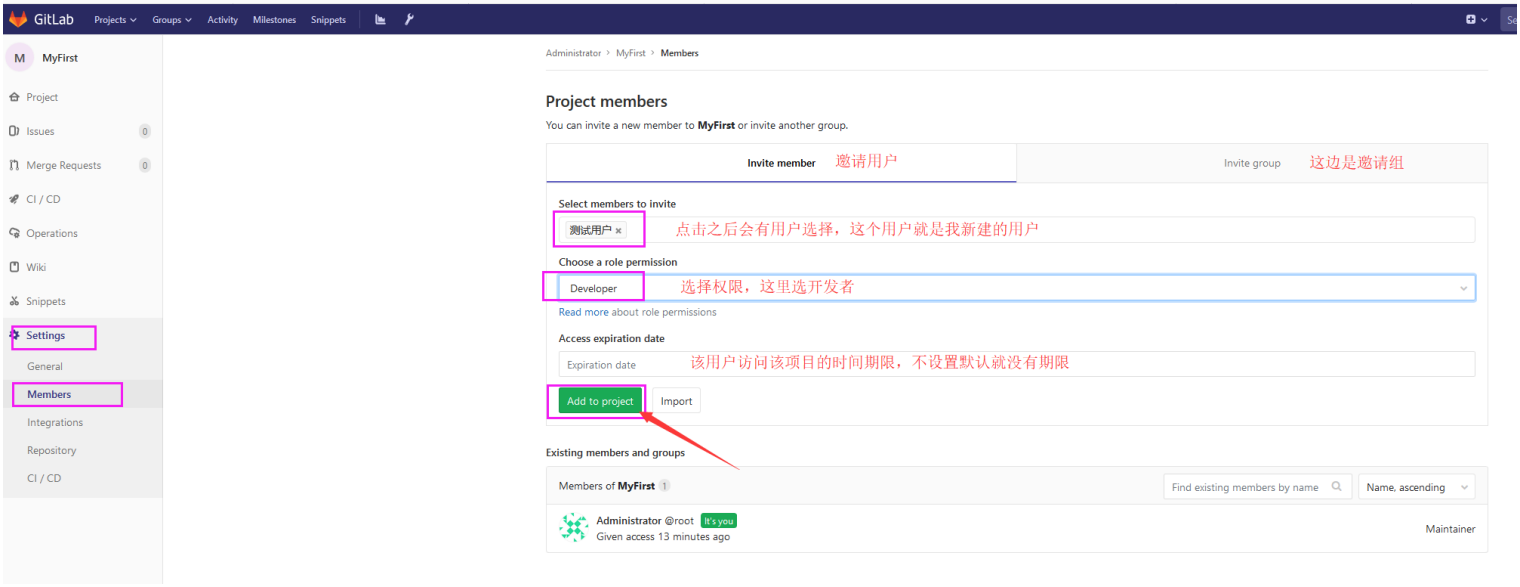
点击



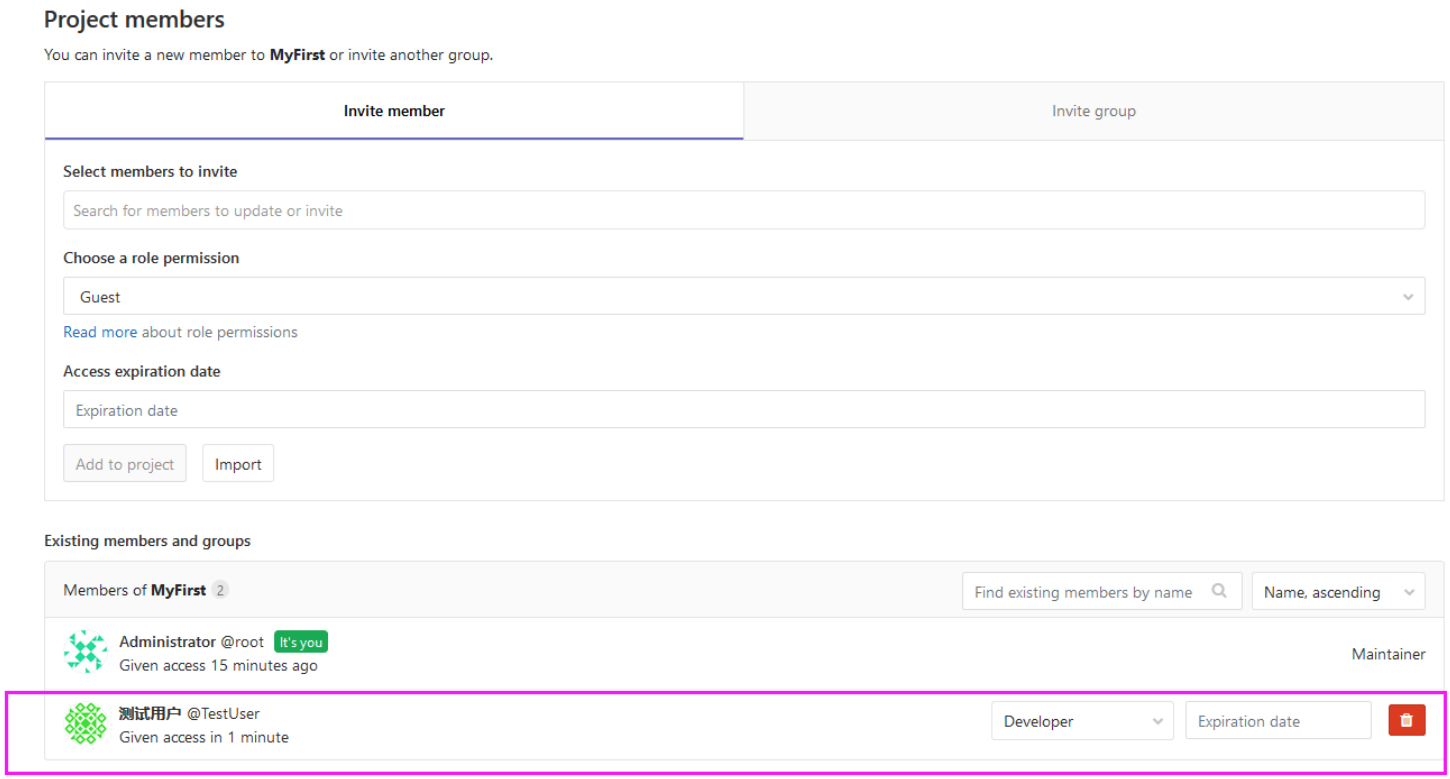
点击设置，然后成员



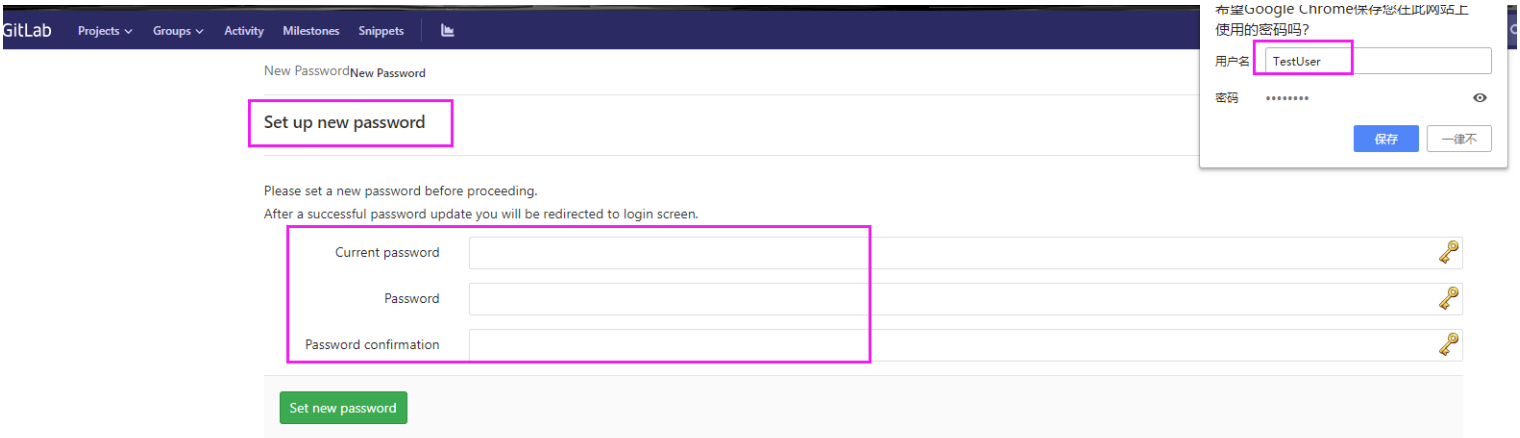
就可以邀请用户了，还可以邀请组，组我们还没有创建（需要注意的是 Developer 权限不能 pushmaster 主分支，需要更高的权限）



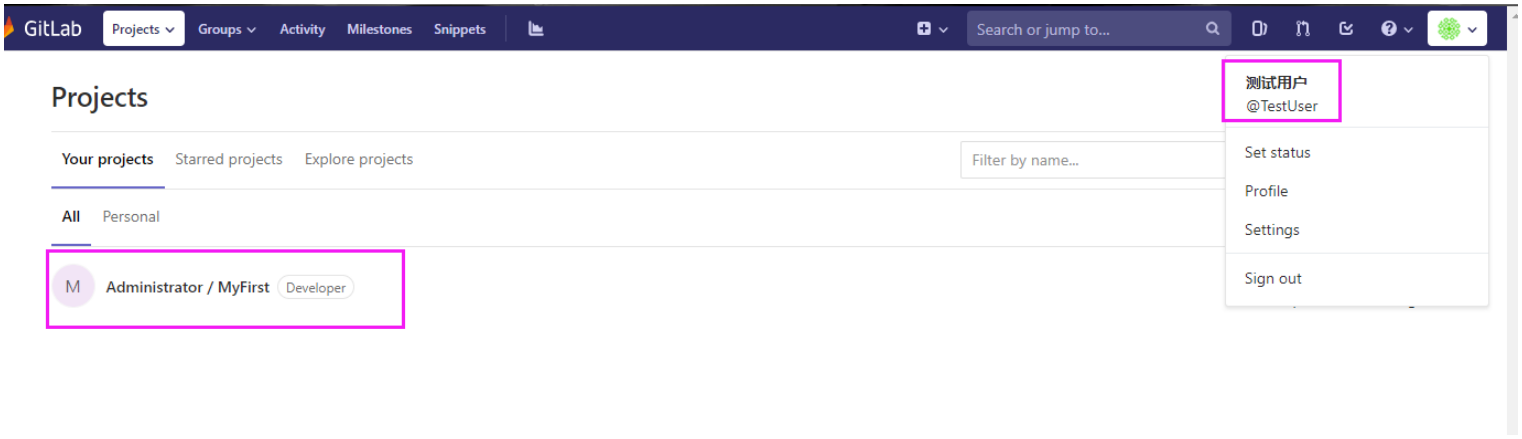
然后就能看到有新的成员的了，这样该项目 TestUser 用户就能管理了



登录 TestUser 测试用户（换一个浏览器登录，root 用户就不用退出了，后面 root 用户还会用到），会强制要求修改密码

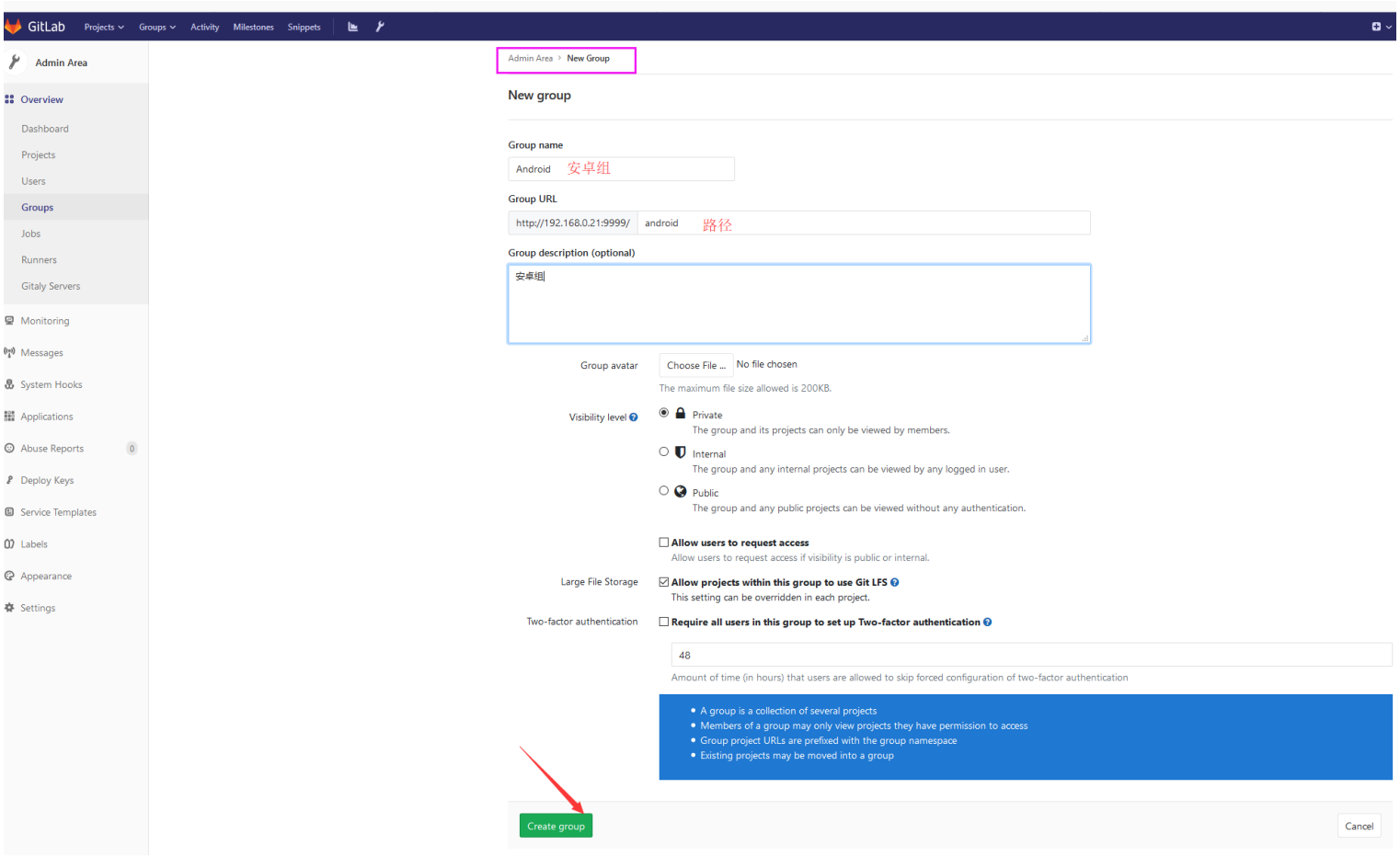


测试用户就有了新添加的项目

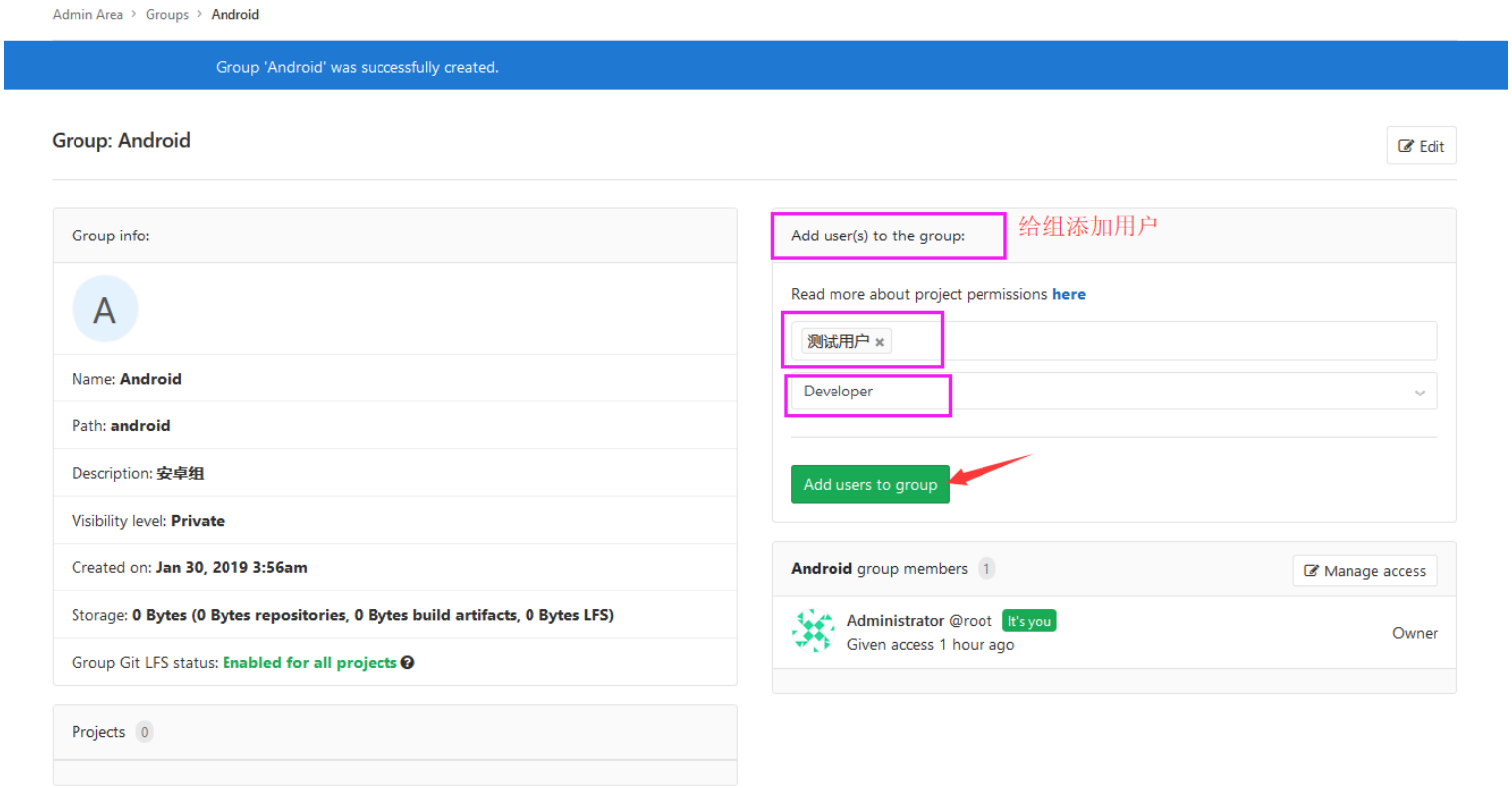


创建组

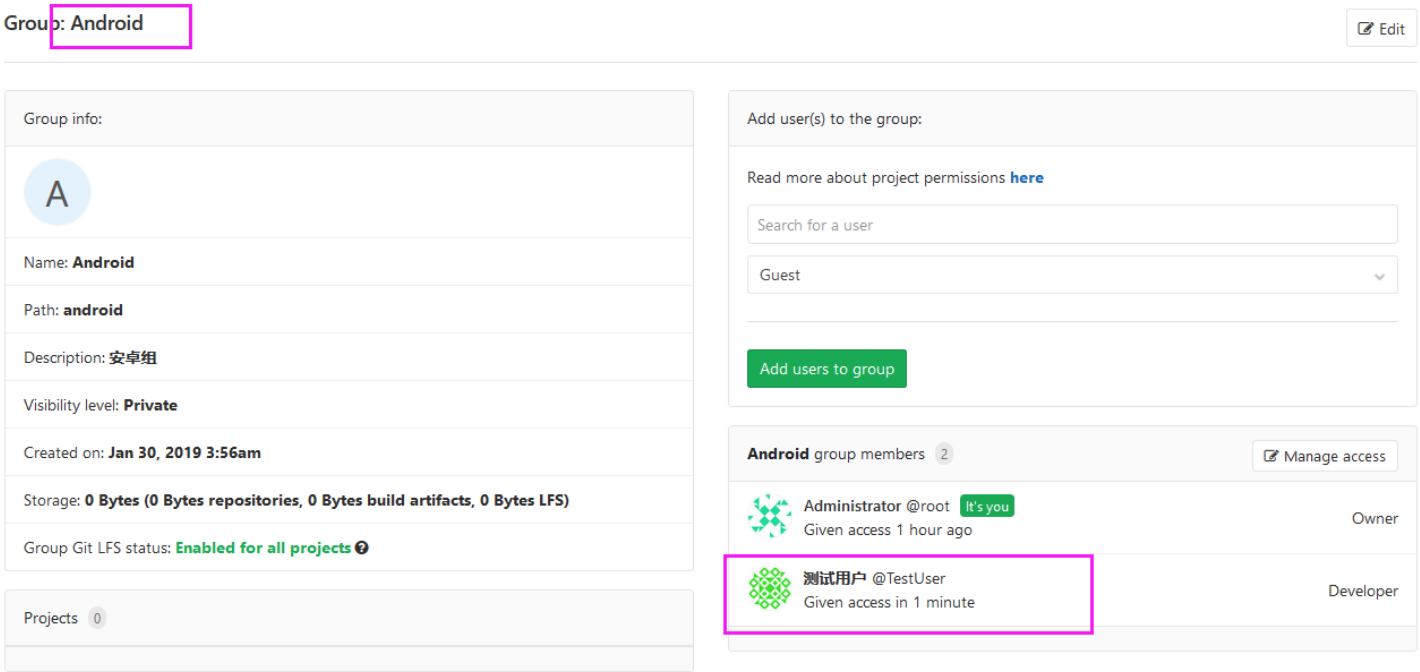
回到 root 用户，看图



创建组完成之后，该界面就能添加用户进 组 了

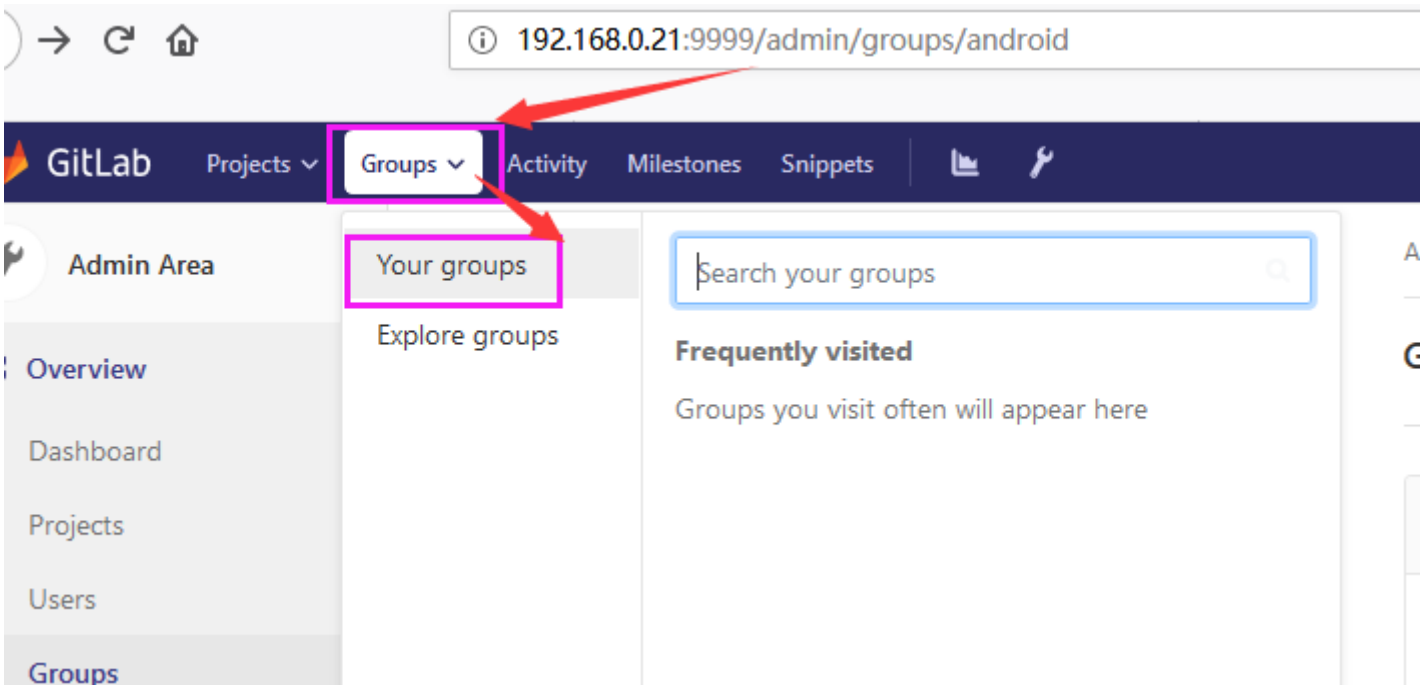


添加好了

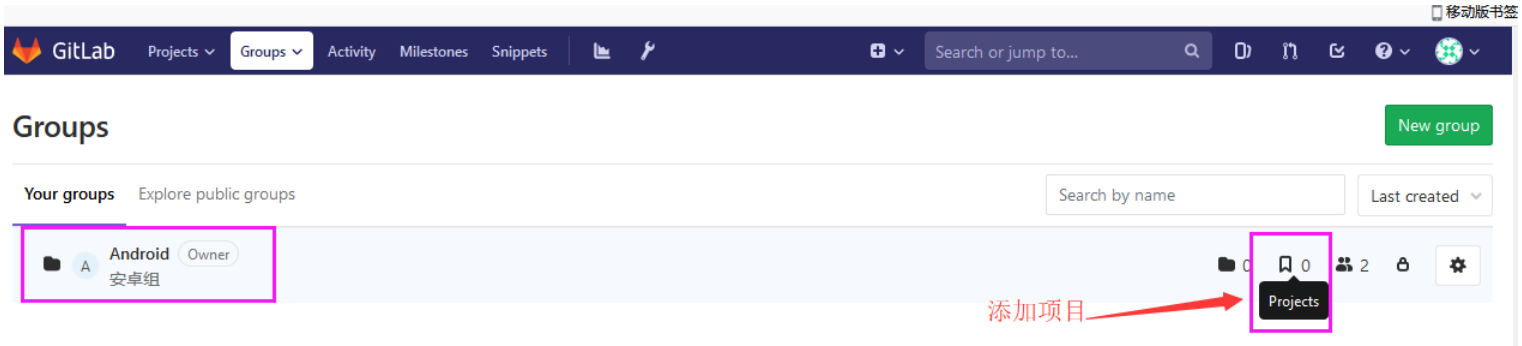


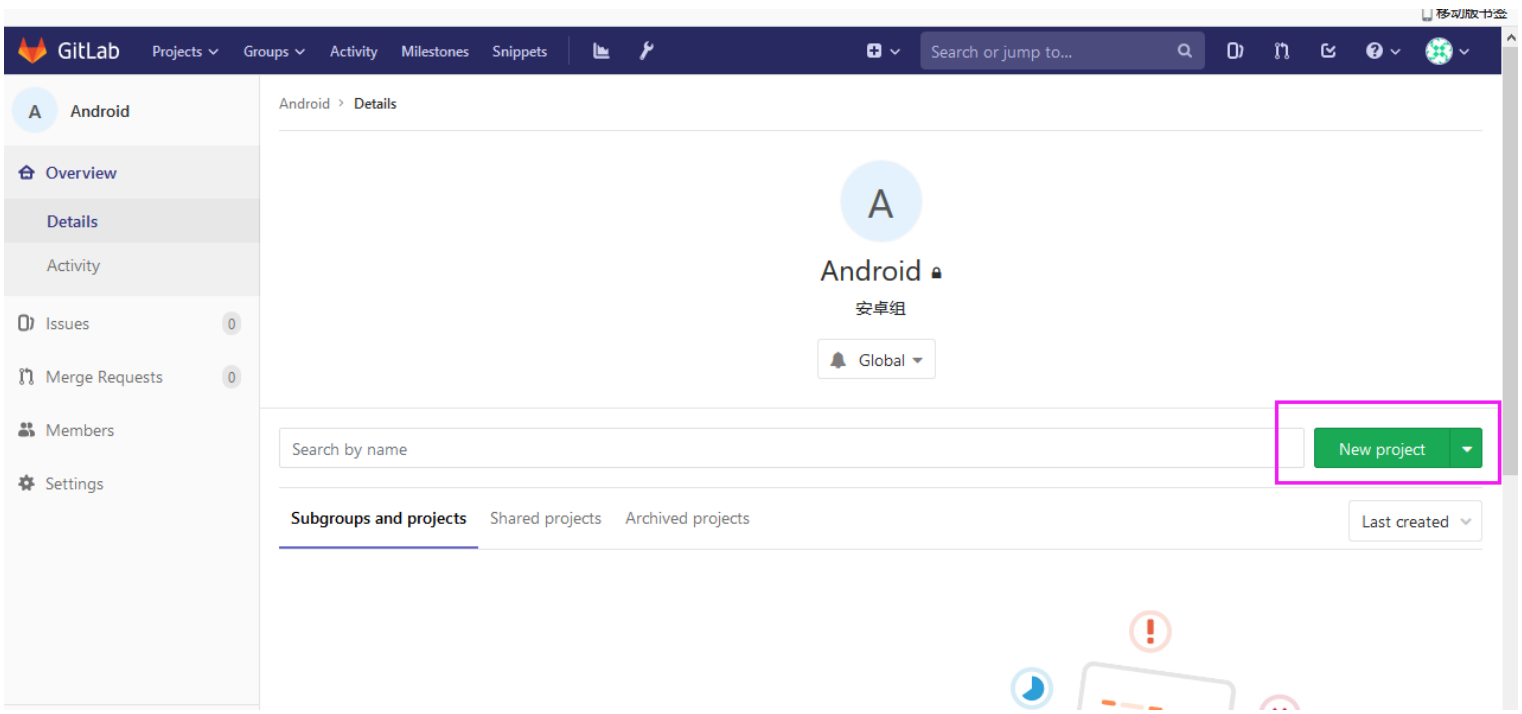
给组新建项目

给组添加项目，这样整一个组的成员都可以管理该项目

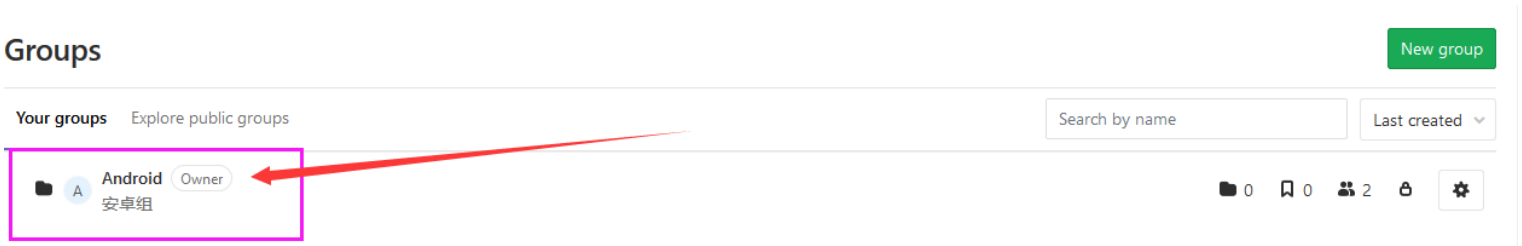


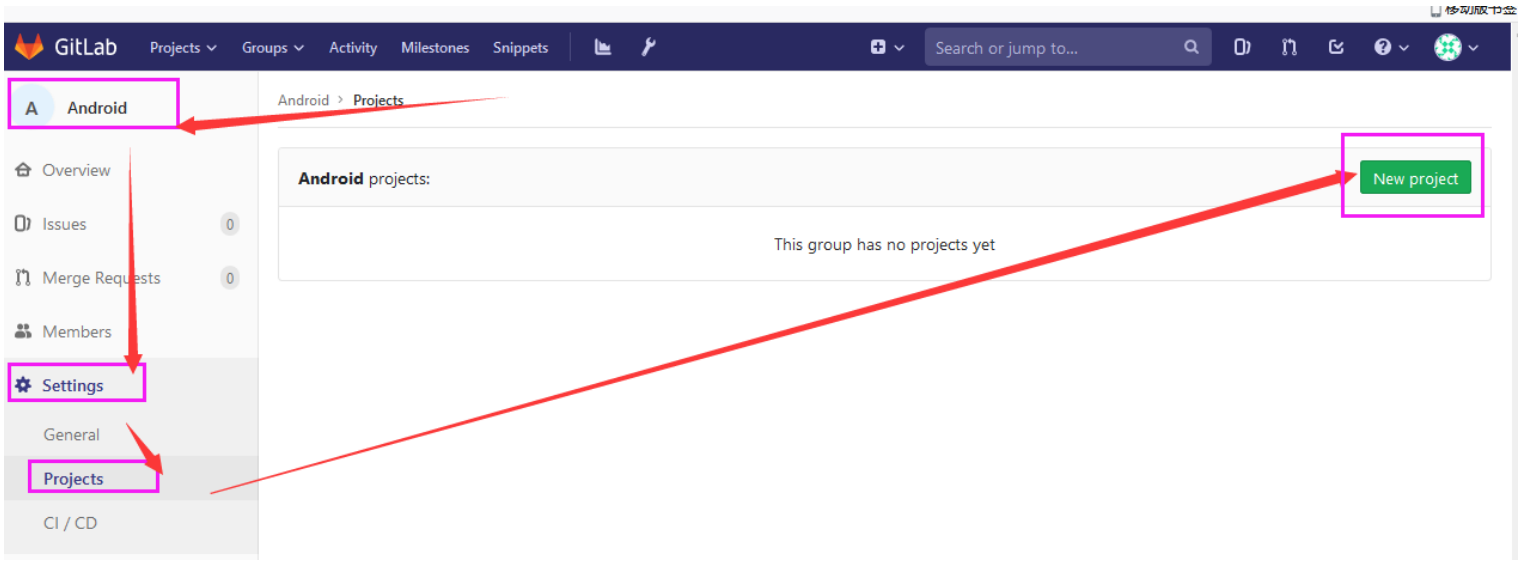
给组添加项目，可以从下图小图标进去

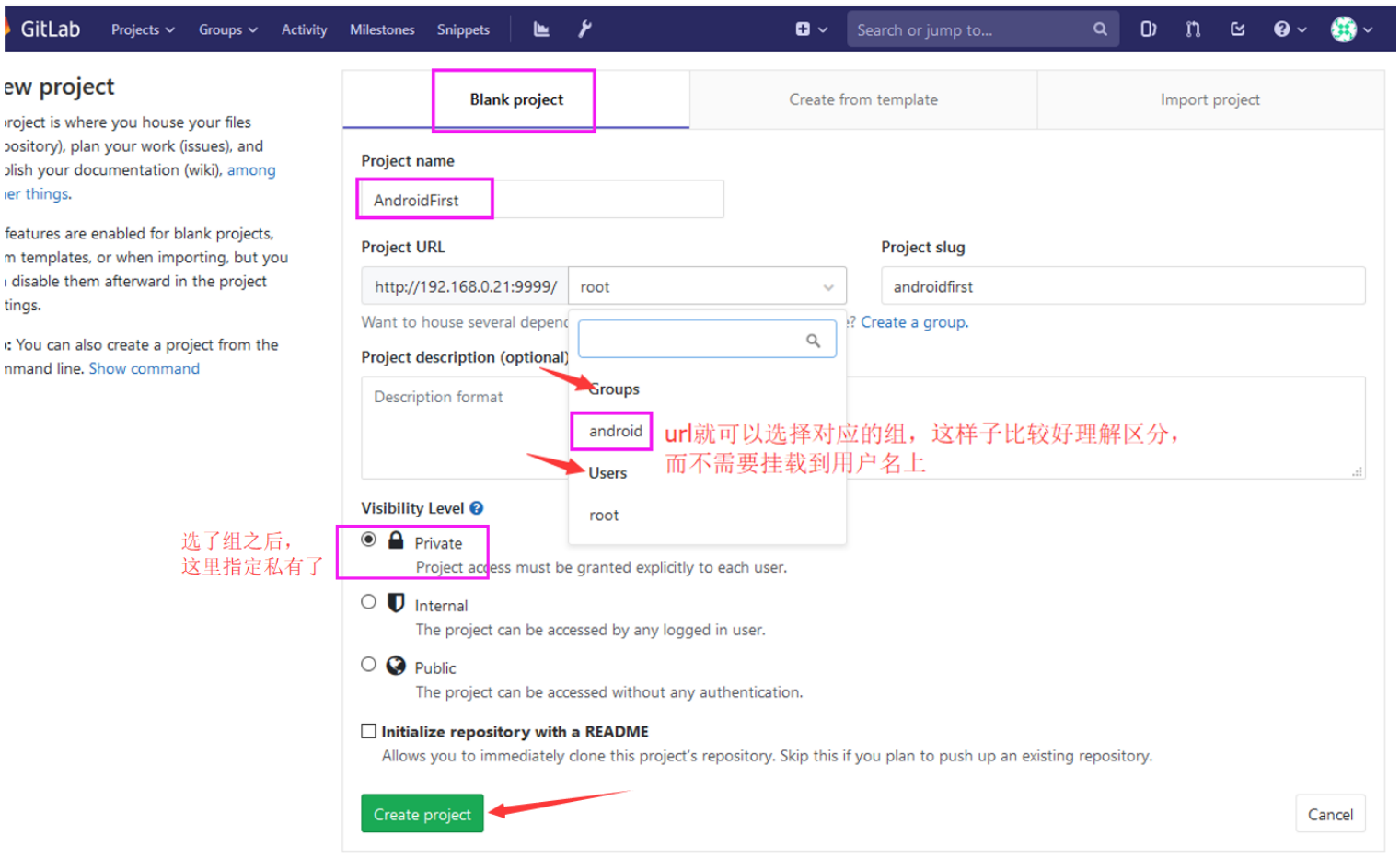




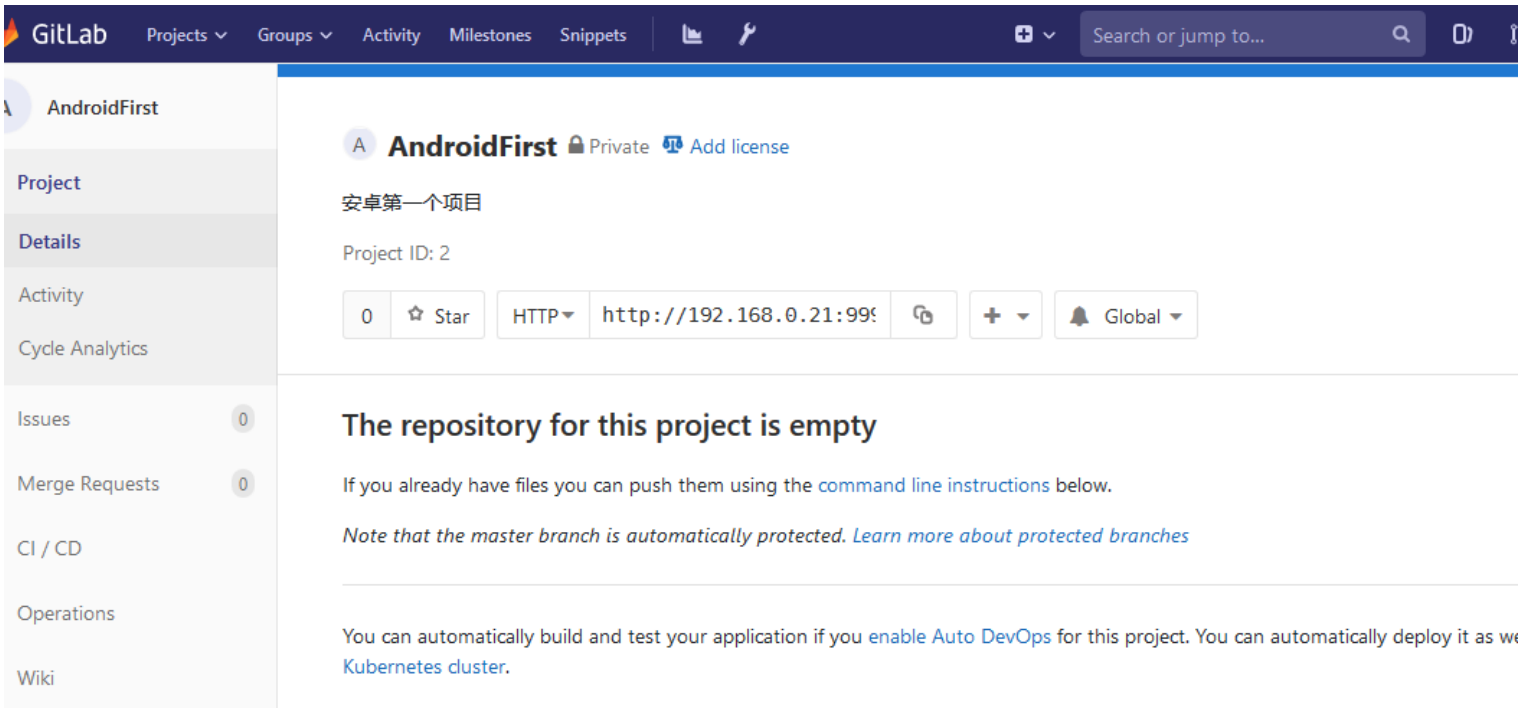
也可以，点击 组名称，然后选择 设置 进去添加项目


Groups
New group
Your groups
Explore public groups
Search by name
Last created
Android
Owner
安卓组


GitLab
Projects
Groups
Activity
Milestones
Snippets
Search or jump to...
Android
Android › Projects
Overview
Issues
Merge Requests
Members
Settings
General
Projects
CI / CD
Android projects:
New project
This group has no projects yet

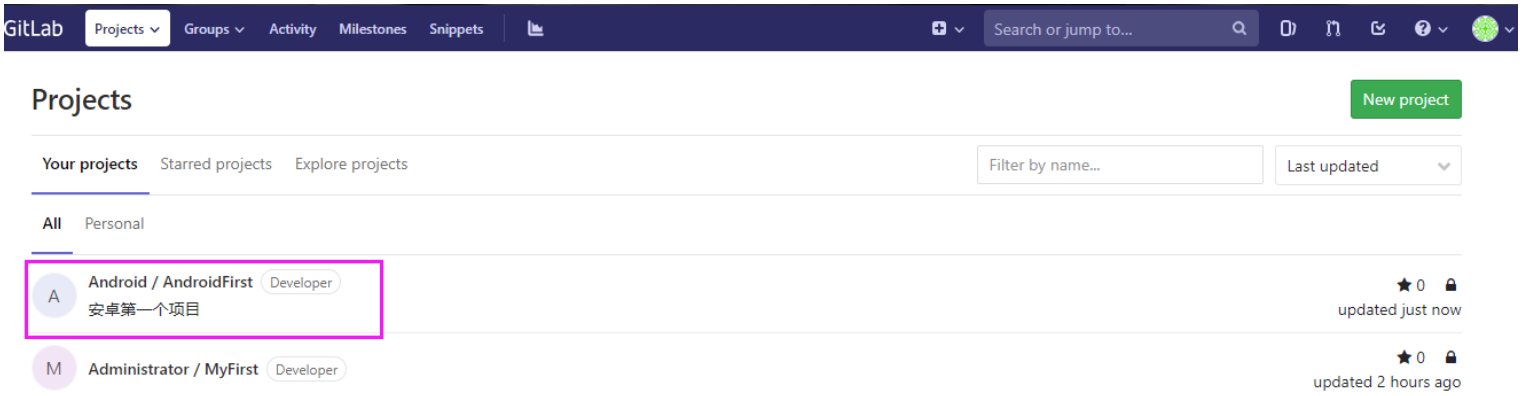

GitLab
Projects
Groups
Activity
Milestones
Snippets
Search or jump to...
ew project
roject is where you house your files
ository), plan your work (issues), and
olish your documentation (wiki), among
er things.
features are enabled for blank projects,
m templates, or when importing, but you
disable them afterward in the project
tings.
: You can also create a project from the
nmand line. Show command
Blank project
Create from template
Import project
Project name
AndroidFirst
Project URL
http://192.168.0.21:9999/
root
Project slug
androidfirst
Want to house several depend
? Create a group.
Project description (optional)
Description format
Groups
android
Users
root
url就可以选择对应的组，这样子比较好理解区分，
而不需要挂载到用户名上
Visibility Level
选了组之后，
这里指定私有了
Private
Project access must be granted explicitly to each user.
Internal
The project can be accessed by any logged in user.
Public
The project can be accessed without any authentication.
Initialize repository with a README
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.
Create project
Cancel

创建完成



看下仓库路径
http://192.168.0.21:9999/android/androidfirst.git

登录 TestUser 用户，可以看到，又多了一个项目



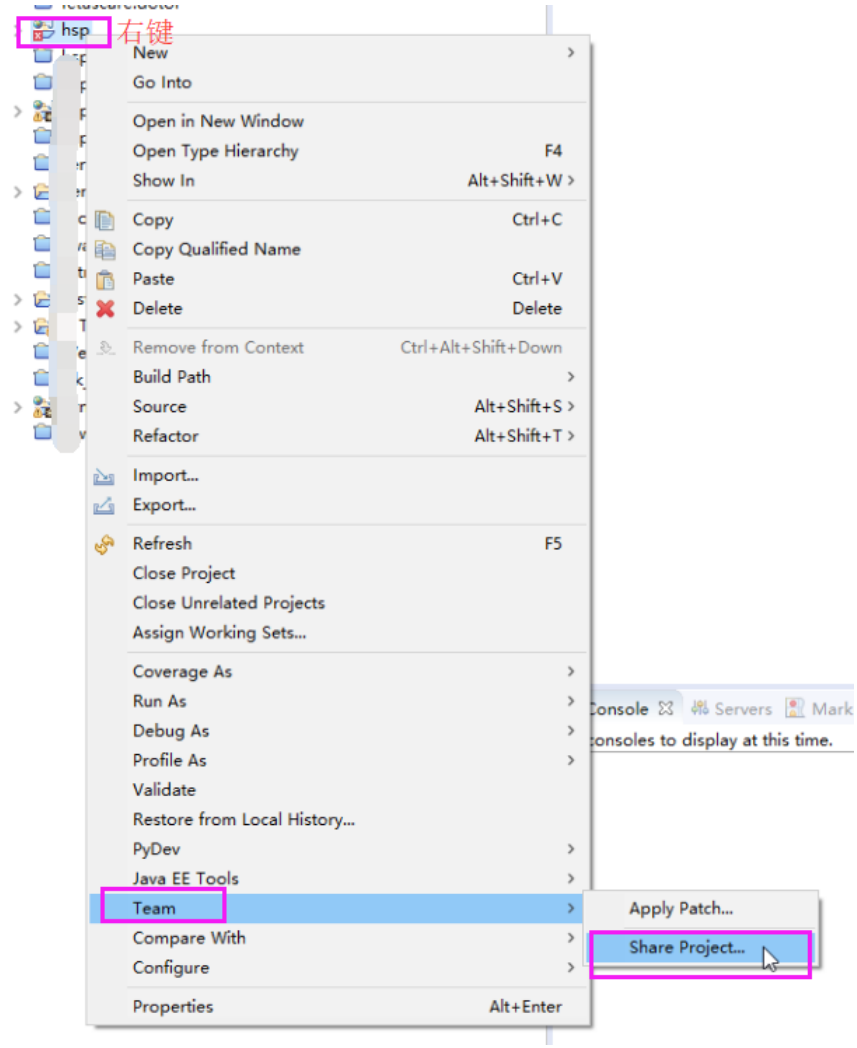
这样子项目、用户、组，就都说了，对于团队的项目，推荐采用分组，然后组添加人员，然后在组里面创建项目（注意，组里面不能添加用户下的已有项目，只能新建项目，或者导入，起码我没有找到能添加的方法）。

5、Eclipse 提交代码到本地仓库、私有远程 GitLab 仓库

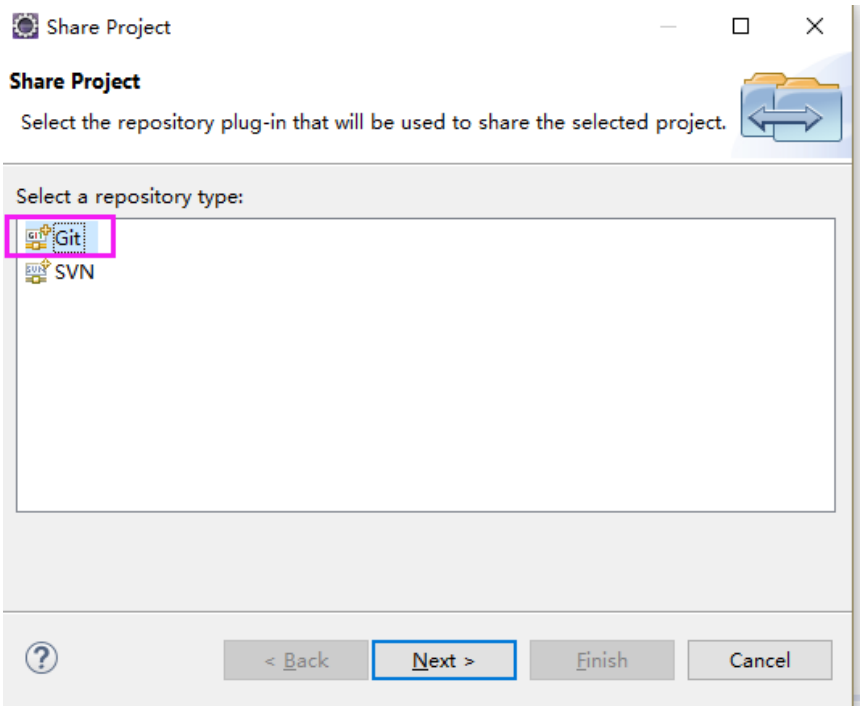
先在 GitLab 上创建项目 http://192.168.0.21:9999/backend/hsp.git ，我这里是分组，组里面创建项目。都没有关系。

创建本地仓库

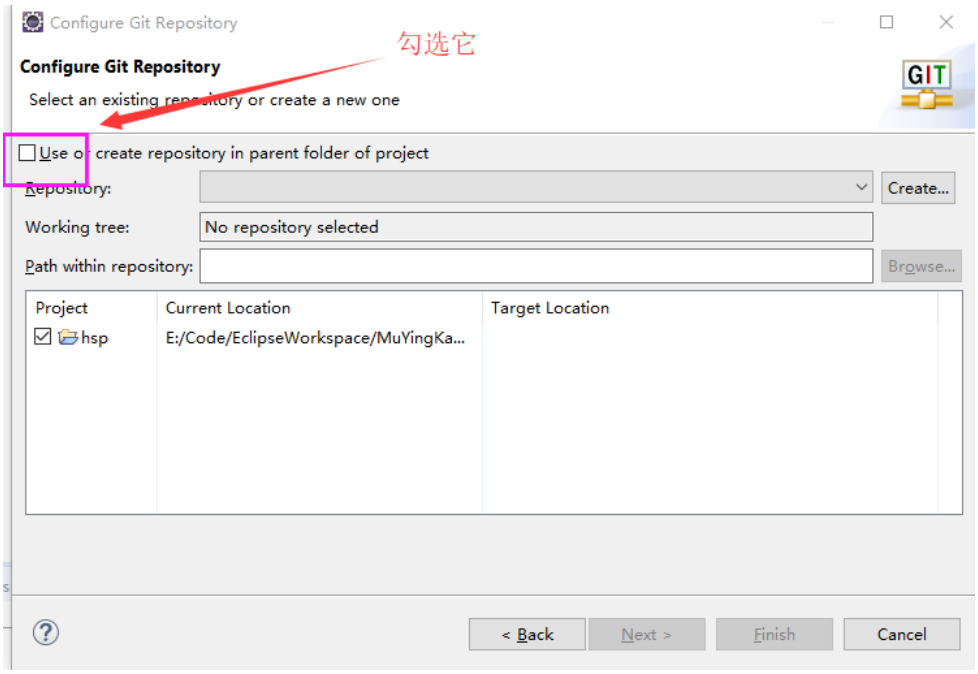
右键需要提交的项目

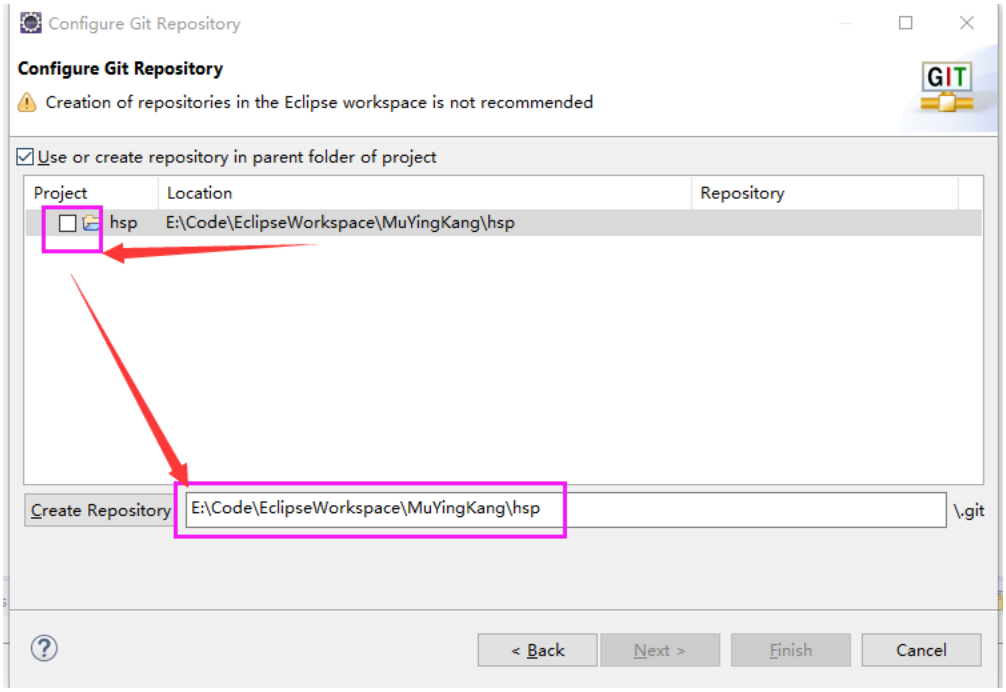


选 Git

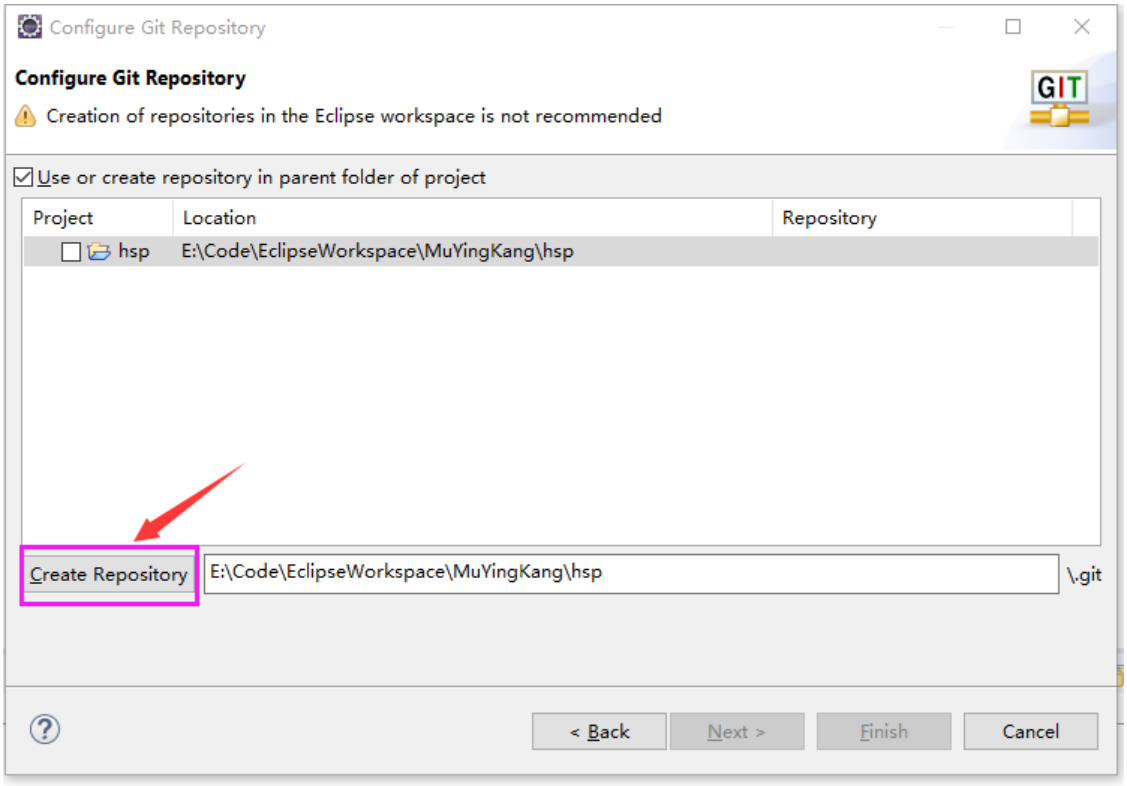


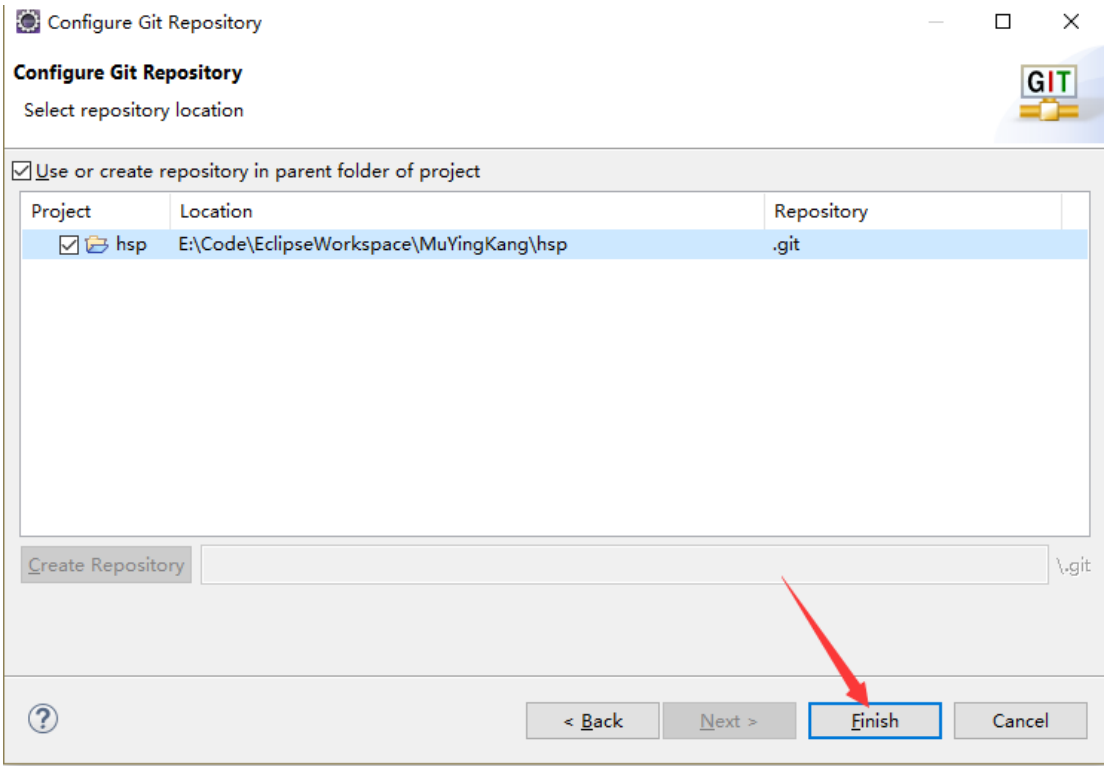
配置本地 Git 仓库





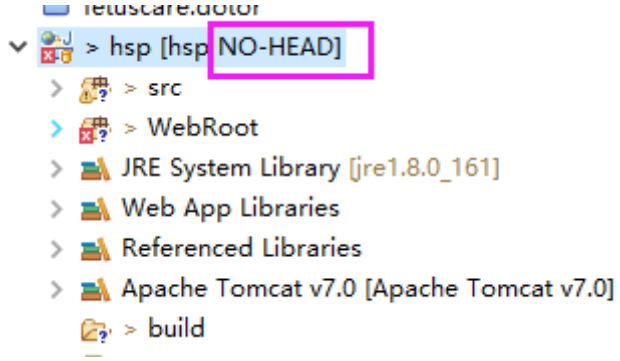
git 提交代码一般都是先提交到 git 本地的仓库，然后再提交到远程的仓库（即类似于 GitLab、GitHub 这样的），这里应该就是创建本地的仓库，这是我用大腿猜的



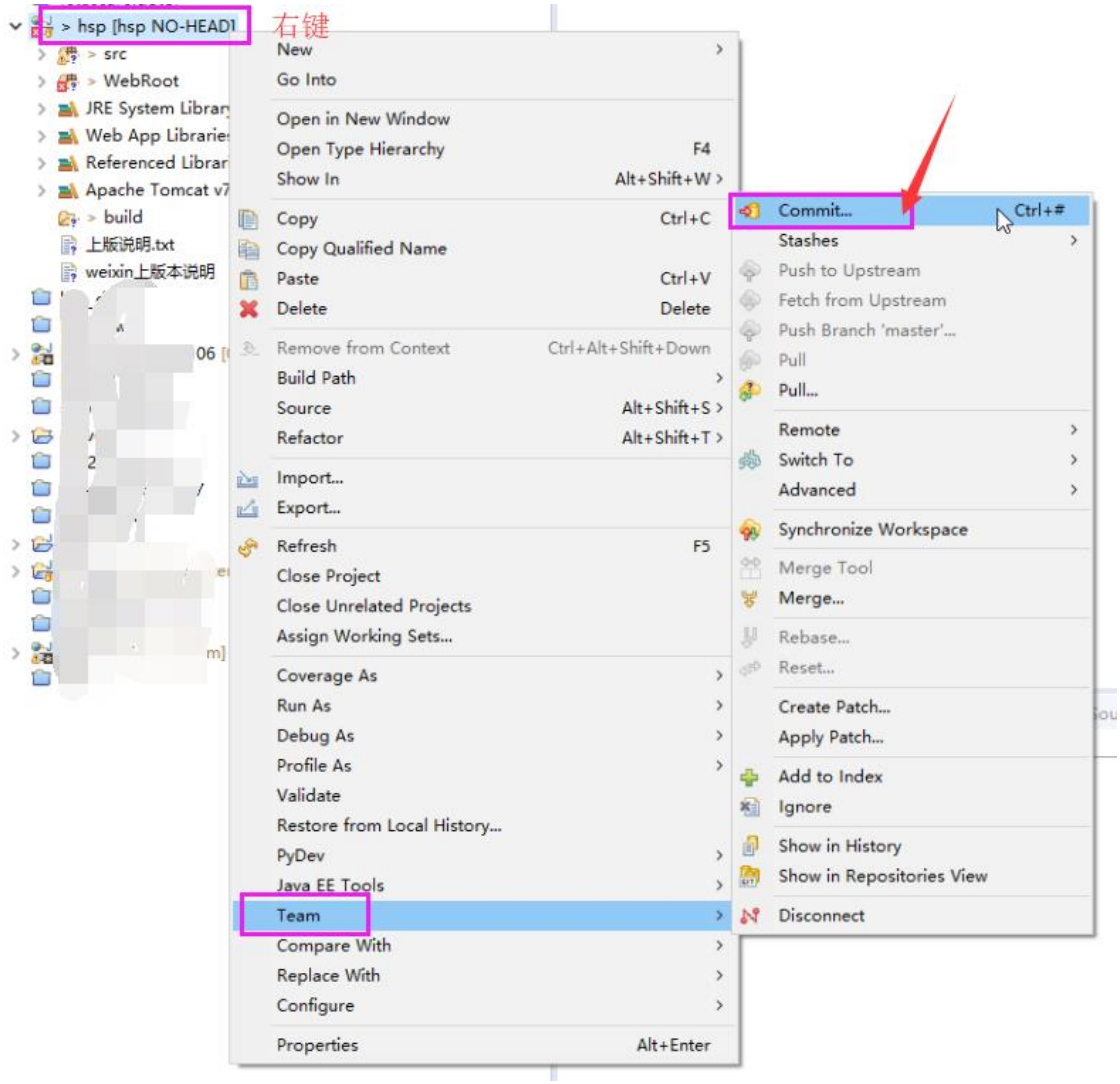


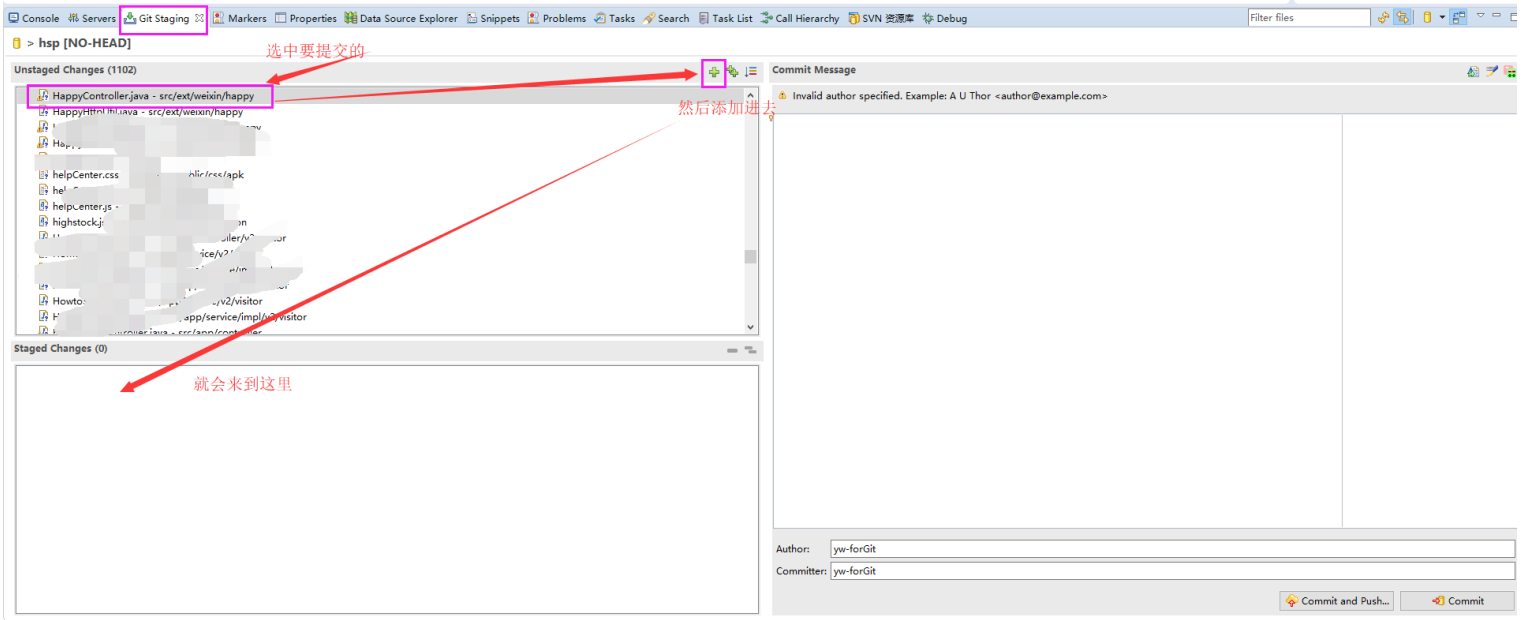
预 Commit

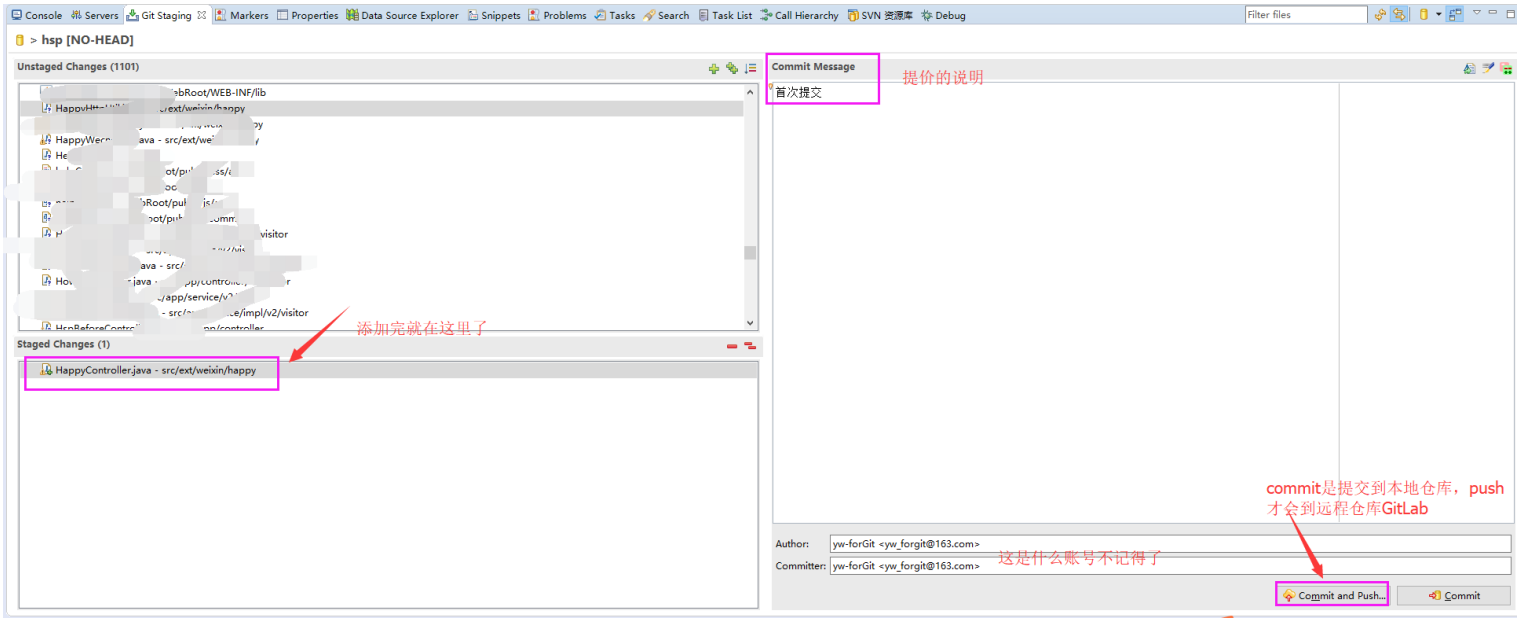
完成之后，什么弹框都没有，不要慌，看项目，多了 NO-HEAD 标志



右键项目，最后点击 Commit 提交，也不要慌，还不会提交的



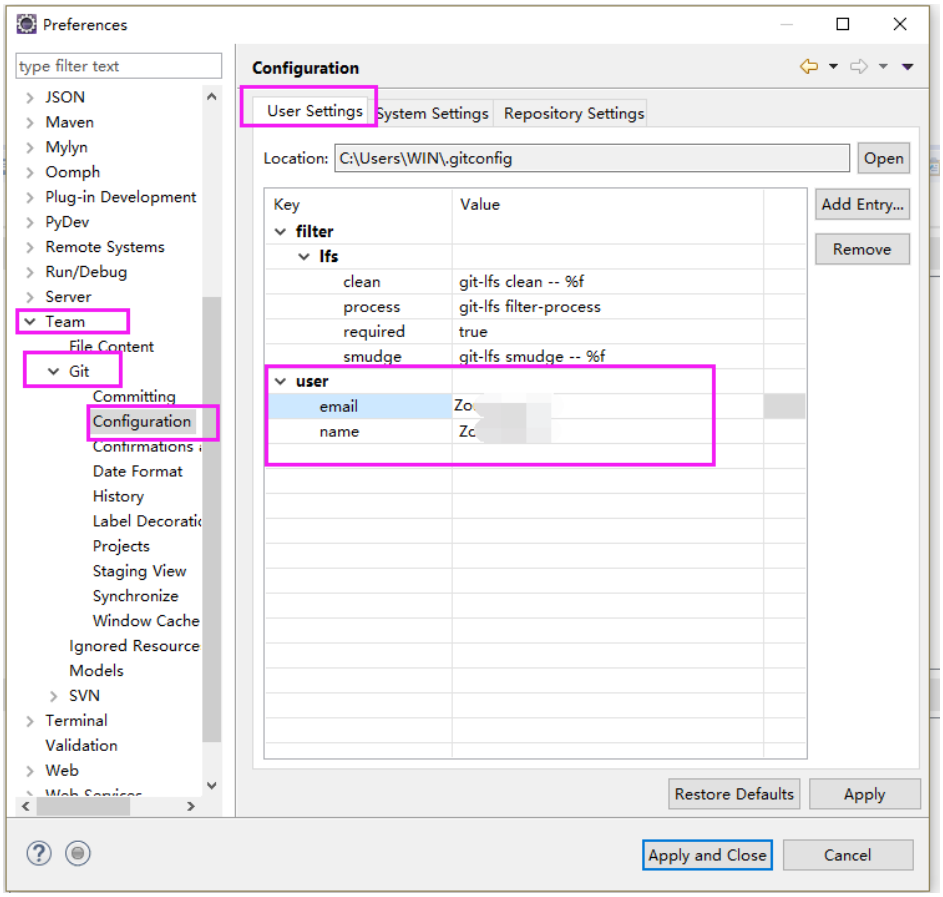




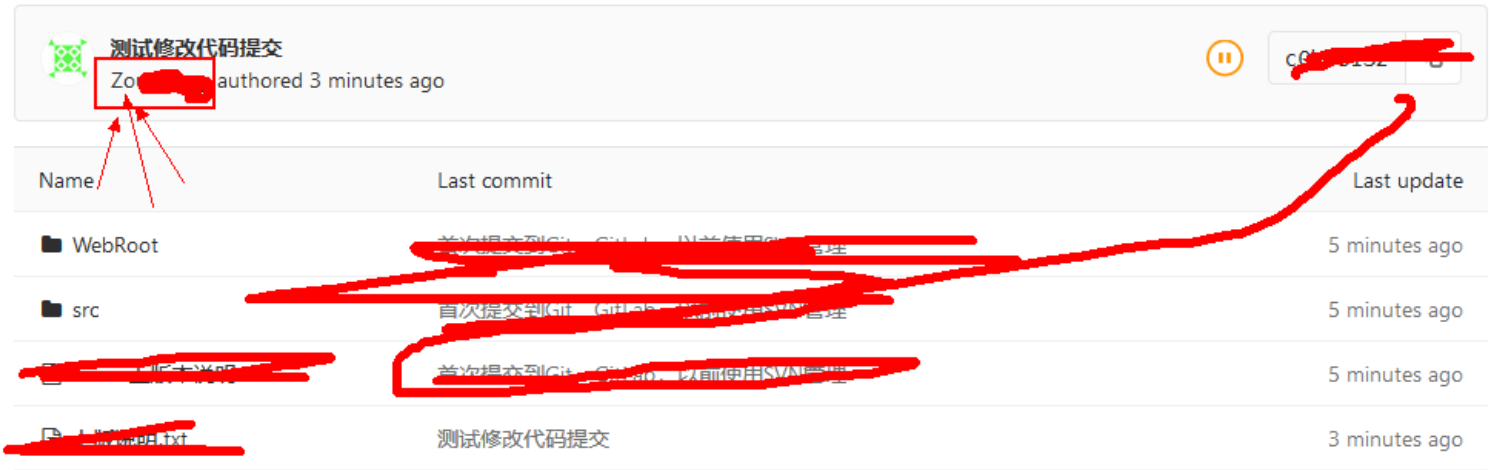
插播 修改提交者的名称

（eclipse 通过 Git 提交到 GitLab，提交代码的作者不是用 GitLab 的账号记录的）

（插播）去设置发现，上图那个 Author 是在这里设置的，要修改，不然提交的时候在 GitLab 那边是以这个名称显示提交者的，注意，修改之后如果不生效，可能要重启 eclipse。

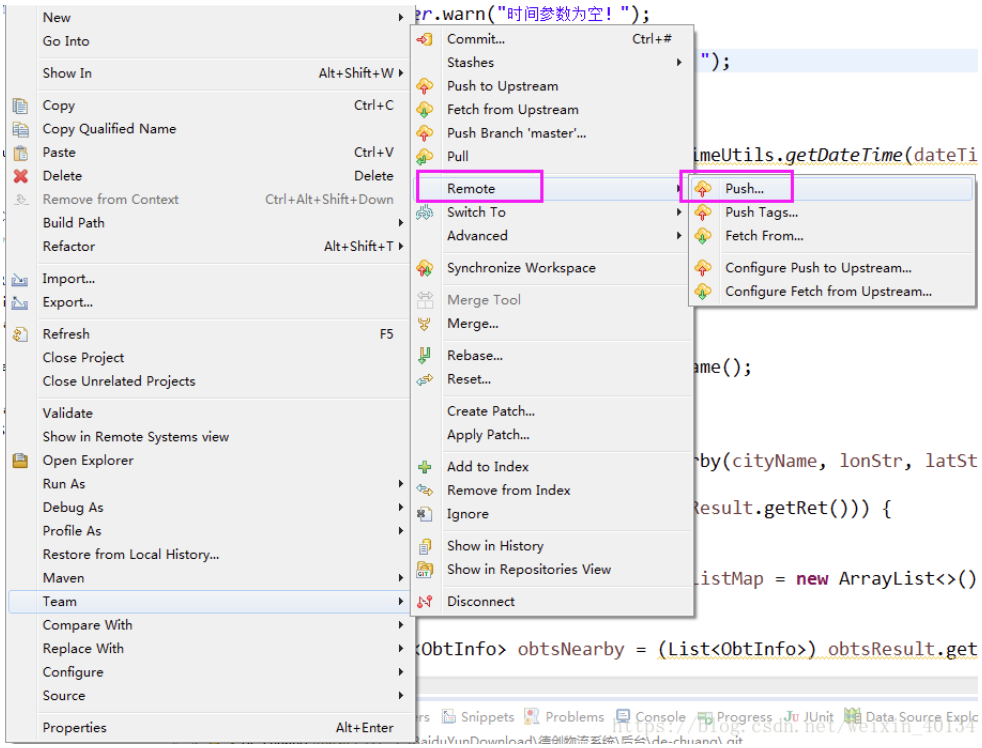


提交完成后，GitLab 仓库查看到



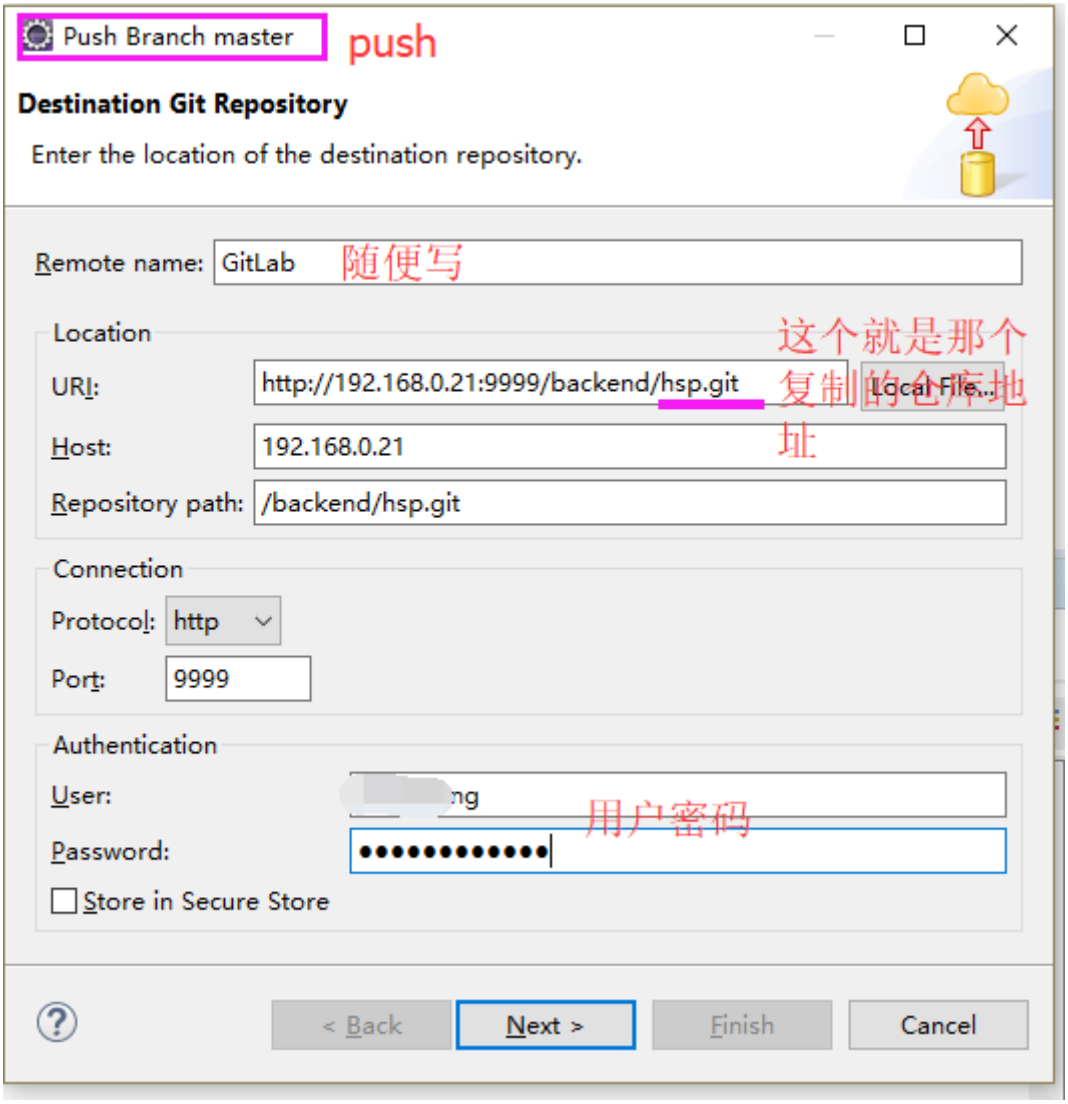
Commit 与 Push 分开

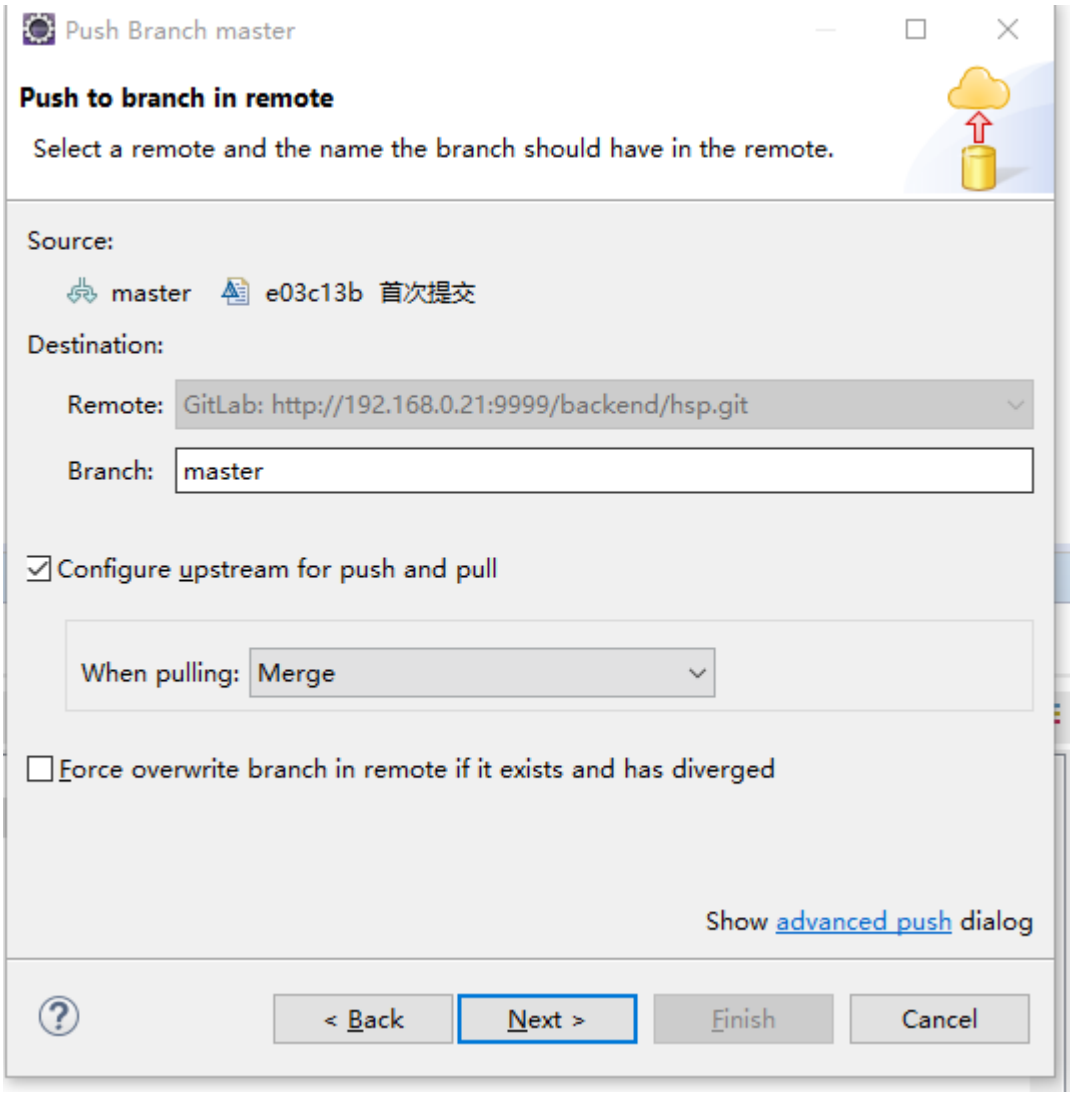
当然也可以先 commit 到本地仓库，最后再 push 到远程仓库（别人的截图）



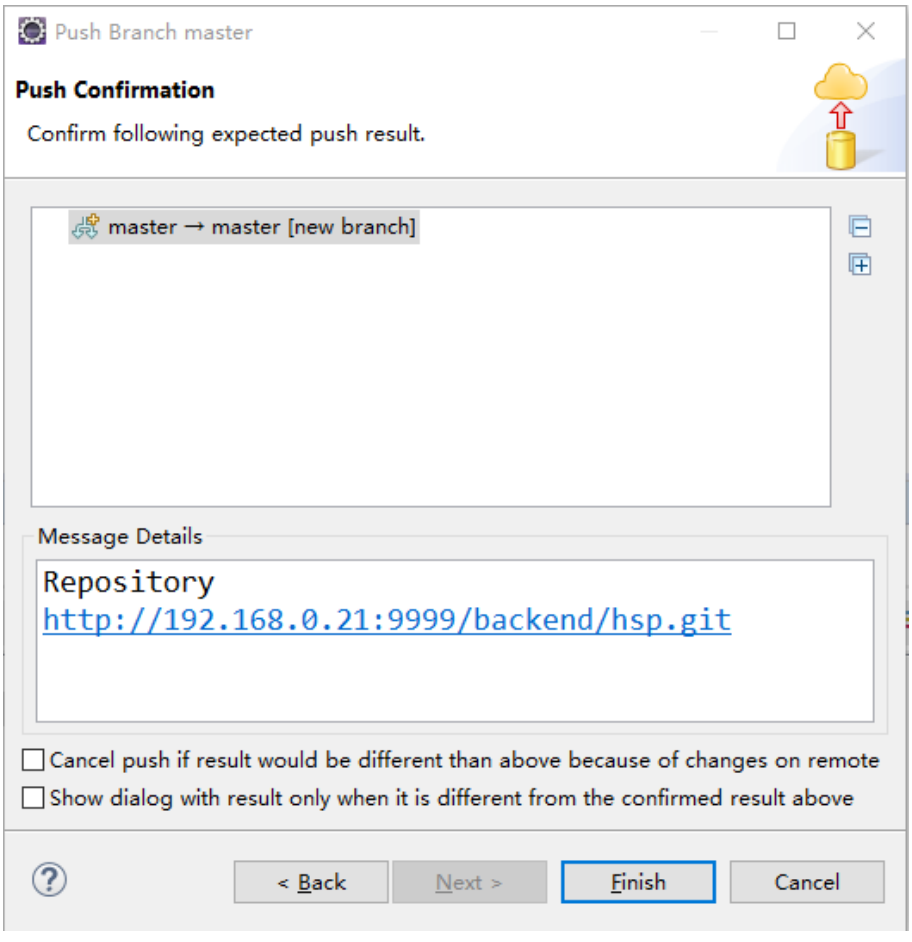
GitLab 仓库配置

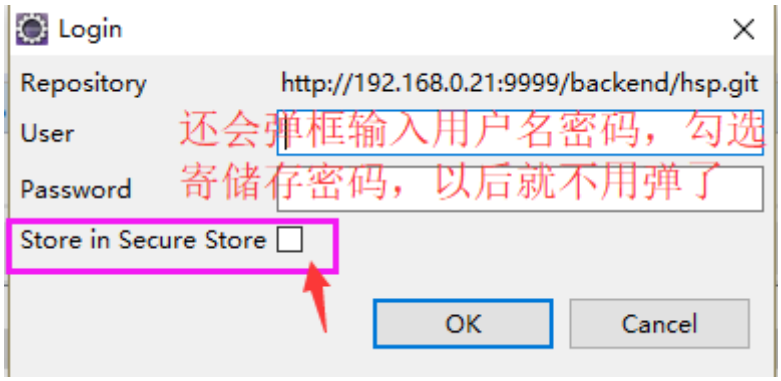
预 Commit、Commit 与 Push 分开 最后都会来到这里。这里的用户名、密码才是 GitLab 的密码。

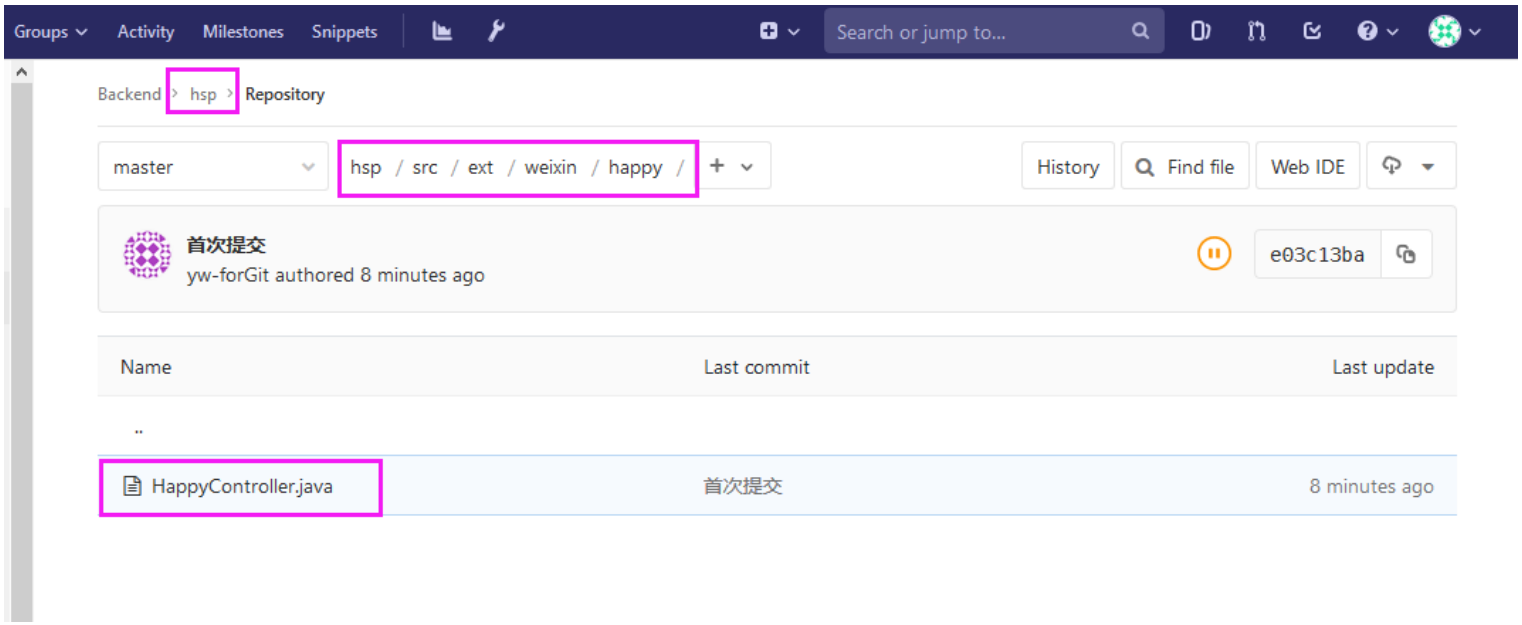




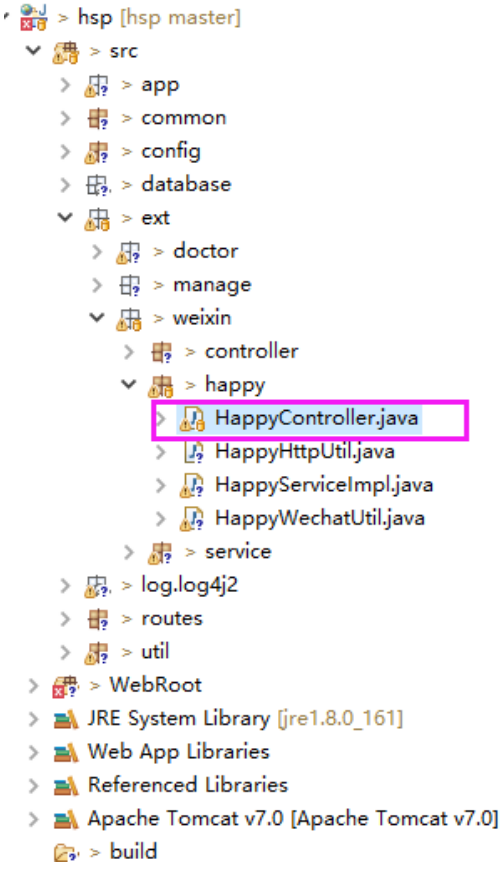
注意，如果提交失败，则会有 Rejected....







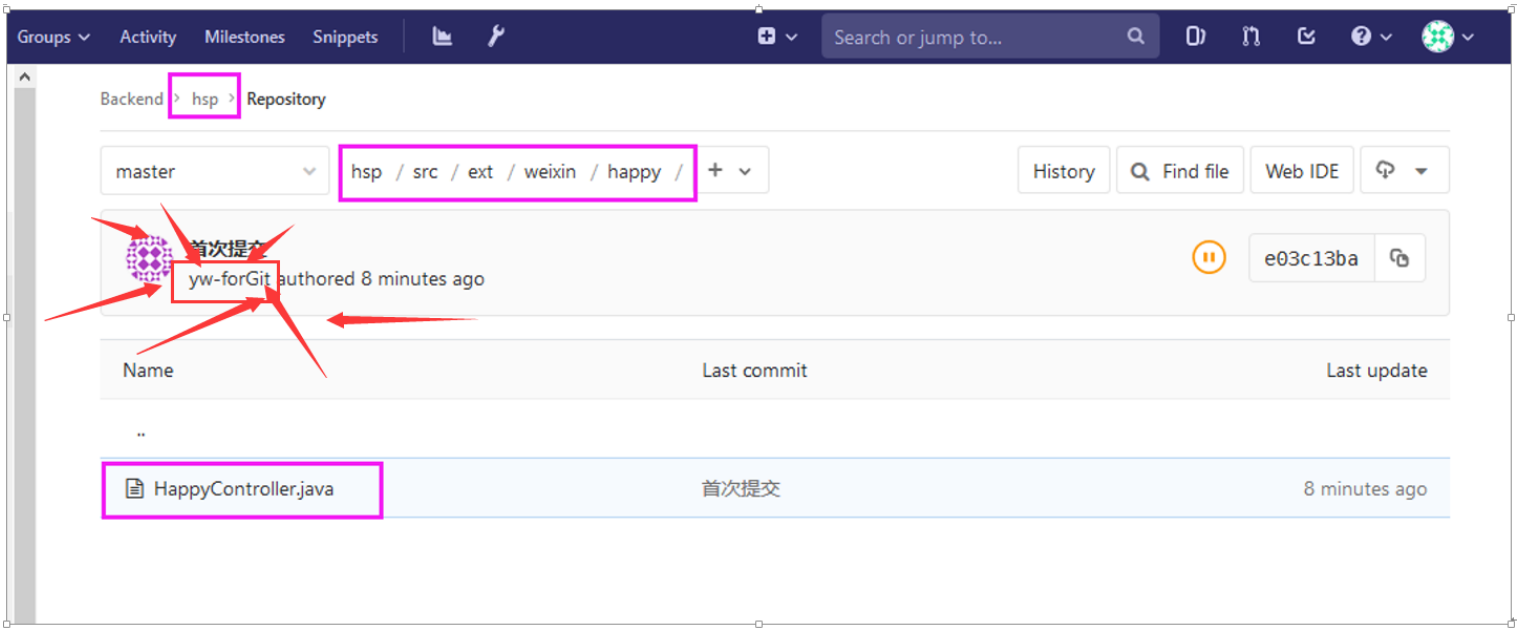
对应一下项目



注意，提交的项目最好要与 GitLab 创建的项目的名称一致，因为提交整个项目，其项目名称这个路径不会提交到 GitLab 上，

只会提交该项目目录下的东西到 GitLab。

注意点：
可以看到 yw-forGit 名称就是我 eclipse 设置的 user 的 name。



（好像 master 分支要一次性提交完成代码，如果第一次只提交了一个文件，后面想一次性提交整个项目，会被 rejected，好像是这样的，然后我删除整个本地 git 仓库提交记录、GitLab 私有仓库的对应项目，重新提交就好了）

结束日期：2019-02-01